

FflasFpack

Generated by Doxygen 1.9.5



|   |           |
|---|-----------|
| <b>1 FFLAS-FFPACK Documentation.</b>                          | <b>1</b>  |
| 1.1 Introduction . . . . .                                    | 1         |
| 1.2 Goals . . . . .   | 1         |
| 1.3 Design . . . . .  | 1         |
| 1.4 Using FFLAS-FFPACK. . . . .                               | 1         |
| 1.5 Contributing to fflas-ffpack, getting assistance. . . . . | 1         |
| <b>2 Configuring and Installing FFLAS-FFPACK</b>              | <b>3</b>  |
| <b>3 Copying and Licence</b>                                  | <b>5</b>  |
| <b>4 Tutorial</b>   | <b>7</b>  |
| <b>5 Architecture of the library.</b>                         | <b>9</b>  |
| <b>6 Bug List</b>   | <b>11</b> |
| <b>7 Bibliography</b>   | <b>15</b> |
| <b>8 Todo List</b>  | <b>17</b> |
| <b>9 Module Index</b>   | <b>21</b> |
| 9.1 Modules . . . . .   | 21        |
| <b>10 Namespace Index</b>                                     | <b>23</b> |
| 10.1 Namespace List . . . . .                                 | 23        |
| <b>11 Hierarchical Index</b>                                  | <b>25</b> |
| 11.1 Class Hierarchy . . . . .                                | 25        |
| <b>12 Data Structure Index</b>                                | <b>33</b> |
| 12.1 Data Structures . . . . .                                | 33        |
| <b>13 File Index</b>  | <b>41</b> |
| 13.1 File List . . . . .                                      | 41        |
| <b>14 Module Documentation</b>                                | <b>49</b> |
| 14.1 CHECKER . . . . .  | 49        |
| 14.2 FFLAS-FFPACK . . . . .                                   | 49        |
| 14.2.1 Detailed Description . . . . .                         | 49        |
| 14.3 FFLAS . . . . .  | 50        |
| 14.4 Matrix Multiplication Algorithms . . . . .               | 50        |
| 14.4.1 Detailed Description . . . . .                         | 50        |
| 14.5 SIMD wrapper . . . . .                                   | 50        |
| 14.6 FFPACK . . . . .   | 50        |
| 14.6.1 Detailed Description . . . . .                         | 51        |
| 14.7 FFLAS-FFPACK fields . . . . .                            | 51        |

|                                       |           |
|---------------------------------------|-----------|
| 14.7.1 Detailed Description           | 51        |
| 14.8 RNS                              | 51        |
| 14.9 Interfaces                       | 51        |
| <b>15 Namespace Documentation</b>     | <b>53</b> |
| 15.1 FFLAS Namespace Reference        | 53        |
| 15.1.1 Typedef Documentation          | 78        |
| 15.1.1.1 Checker_fgemm                | 78        |
| 15.1.1.2 Checker_ftsm                 | 78        |
| 15.1.1.3 ForceCheck_fgemm             | 78        |
| 15.1.1.4 ForceCheck_ftsm              | 79        |
| 15.1.1.5 ZOSparseMatrix               | 79        |
| 15.1.1.6 NotZOSparseMatrix            | 79        |
| 15.1.1.7 SimdSparseMatrix             | 79        |
| 15.1.1.8 NoSimdSparseMatrix           | 79        |
| 15.1.1.9 MKLSparseMatrixFormat        | 79        |
| 15.1.1.10 NotMKLSparseMatrixFormat    | 79        |
| 15.1.1.11 has_plus                    | 79        |
| 15.1.1.12 has_minus                   | 80        |
| 15.1.1.13 has_equal                   | 80        |
| 15.1.1.14 has_plus_eq                 | 80        |
| 15.1.1.15 has_minus_eq                | 80        |
| 15.1.1.16 has_mul                     | 80        |
| 15.1.1.17 has_mul_eq                  | 80        |
| 15.1.1.18 Timer                       | 80        |
| 15.1.1.19 BaseTimer                   | 81        |
| 15.1.1.20 UserTimer                   | 81        |
| 15.1.1.21 SysTimer                    | 81        |
| 15.1.2 Enumeration Type Documentation | 81        |
| 15.1.2.1 FFLAS_ORDER                  | 81        |
| 15.1.2.2 FFLAS_TRANSPOSE              | 81        |
| 15.1.2.3 FFLAS_UPLO                   | 82        |
| 15.1.2.4 FFLAS_DIAG                   | 82        |
| 15.1.2.5 FFLAS_SIDE                   | 82        |
| 15.1.2.6 FFLAS_BASE                   | 82        |
| 15.1.2.7 number_kind                  | 83        |
| 15.1.2.8 SparseMatrix_t               | 83        |
| 15.1.2.9 FFLAS_FORMAT                 | 83        |
| 15.1.3 Function Documentation         | 84        |
| 15.1.3.1 InfNorm()                    | 84        |
| 15.1.3.2 min3()                       | 84        |
| 15.1.3.3 max3()                       | 84        |

|                            |    |
|----------------------------|----|
| 15.1.3.4 min4()            | 84 |
| 15.1.3.5 max4()            | 85 |
| 15.1.3.6 fadd() [1/8]      | 85 |
| 15.1.3.7 faddin() [1/4]    | 85 |
| 15.1.3.8 fsub() [1/4]      | 85 |
| 15.1.3.9 fsubin() [1/3]    | 86 |
| 15.1.3.10 fadd() [2/8]     | 86 |
| 15.1.3.11 pfadd()          | 86 |
| 15.1.3.12 pfsub()          | 87 |
| 15.1.3.13 pfaddin()        | 87 |
| 15.1.3.14 pfsubin()        | 87 |
| 15.1.3.15 fadd() [3/8]     | 87 |
| 15.1.3.16 fsub() [2/4]     | 89 |
| 15.1.3.17 faddin() [2/4]   | 89 |
| 15.1.3.18 fsubin() [2/3]   | 90 |
| 15.1.3.19 fadd() [4/8]     | 90 |
| 15.1.3.20 fassign() [1/10] | 91 |
| 15.1.3.21 fassign() [2/10] | 91 |
| 15.1.3.22 fassign() [3/10] | 91 |
| 15.1.3.23 fassign() [4/10] | 92 |
| 15.1.3.24 fassign() [5/10] | 92 |
| 15.1.3.25 fassign() [6/10] | 92 |
| 15.1.3.26 fassign() [7/10] | 92 |
| 15.1.3.27 fassign() [8/10] | 93 |
| 15.1.3.28 faxpy() [1/6]    | 93 |
| 15.1.3.29 faxpy() [2/6]    | 94 |
| 15.1.3.30 faxpy() [3/6]    | 94 |
| 15.1.3.31 faxpy() [4/6]    | 94 |
| 15.1.3.32 fdot() [1/11]    | 95 |
| 15.1.3.33 fdot() [2/11]    | 95 |
| 15.1.3.34 fdot() [3/11]    | 95 |
| 15.1.3.35 fdot() [4/11]    | 95 |
| 15.1.3.36 fdot() [5/11]    | 96 |
| 15.1.3.37 fdot() [6/11]    | 96 |
| 15.1.3.38 fdot() [7/11]    | 96 |
| 15.1.3.39 fdot() [8/11]    | 96 |
| 15.1.3.40 fgemm() [1/23]   | 97 |
| 15.1.3.41 fgemm() [2/23]   | 97 |
| 15.1.3.42 fgemm() [3/23]   | 98 |
| 15.1.3.43 fgemm() [4/23]   | 98 |
| 15.1.3.44 fgemm() [5/23]   | 99 |
| 15.1.3.45 fgemm() [6/23]   | 99 |

|                           |     |
|---------------------------|-----|
| 15.1.3.46 fsquare() [1/6] | 100 |
| 15.1.3.47 fsquare() [2/6] | 100 |
| 15.1.3.48 fsquare() [3/6] | 101 |
| 15.1.3.49 fsquare() [4/6] | 101 |
| 15.1.3.50 fsquare() [5/6] | 101 |
| 15.1.3.51 fgemm() [7/23]  | 101 |
| 15.1.3.52 fgemm() [8/23]  | 102 |
| 15.1.3.53 fgemm() [9/23]  | 102 |
| 15.1.3.54 fgemm() [10/23] | 103 |
| 15.1.3.55 fgemm() [11/23] | 103 |
| 15.1.3.56 fgemm() [12/23] | 103 |
| 15.1.3.57 fgemm() [13/23] | 104 |
| 15.1.3.58 fgemm() [14/23] | 104 |
| 15.1.3.59 fgemm() [15/23] | 105 |
| 15.1.3.60 fgemm() [16/23] | 105 |
| 15.1.3.61 fgemm() [17/23] | 105 |
| 15.1.3.62 fgemm() [18/23] | 106 |
| 15.1.3.63 fgemv() [1/19]  | 106 |
| 15.1.3.64 fgemv() [2/19]  | 107 |
| 15.1.3.65 fgemv() [3/19]  | 107 |
| 15.1.3.66 fgemv() [4/19]  | 107 |
| 15.1.3.67 fgemv() [5/19]  | 108 |
| 15.1.3.68 fgemv() [6/19]  | 108 |
| 15.1.3.69 fgemv() [7/19]  | 109 |
| 15.1.3.70 fgemv() [8/19]  | 109 |
| 15.1.3.71 fgemv() [9/19]  | 109 |
| 15.1.3.72 fgemv() [10/19] | 110 |
| 15.1.3.73 fgemv() [11/19] | 110 |
| 15.1.3.74 fgemv() [12/19] | 111 |
| 15.1.3.75 fgemv() [13/19] | 111 |
| 15.1.3.76 fgemv() [14/19] | 111 |
| 15.1.3.77 fgemv() [15/19] | 112 |
| 15.1.3.78 fgemv() [16/19] | 112 |
| 15.1.3.79 fger() [1/12]   | 112 |
| 15.1.3.80 fger() [2/12]   | 113 |
| 15.1.3.81 fger() [3/12]   | 113 |
| 15.1.3.82 fger() [4/12]   | 114 |
| 15.1.3.83 fger() [5/12]   | 114 |
| 15.1.3.84 fger() [6/12]   | 114 |
| 15.1.3.85 fger() [7/12]   | 115 |
| 15.1.3.86 fger() [8/12]   | 115 |
| 15.1.3.87 fger() [9/12]   | 115 |

|                                   |         |     |
|-----------------------------------|---------|-----|
| 15.1.3.88 fger()                  | [10/12] | 116 |
| 15.1.3.89 fger()                  | [11/12] | 116 |
| 15.1.3.90 freduce()               | [1/10]  | 116 |
| 15.1.3.91 freduce()               | [2/10]  | 117 |
| 15.1.3.92 freduce_constoverride() | [1/2]   | 117 |
| 15.1.3.93 finit()                 | [1/8]   | 117 |
| 15.1.3.94 finit()                 | [2/8]   | 118 |
| 15.1.3.95 freduce()               | [3/10]  | 118 |
| 15.1.3.96 pfreduce()              |         | 118 |
| 15.1.3.97 freduce()               | [4/10]  | 119 |
| 15.1.3.98 freduce_constoverride() | [2/2]   | 119 |
| 15.1.3.99 finit()                 | [3/8]   | 119 |
| 15.1.3.100 finit()                | [4/8]   | 120 |
| 15.1.3.101 freduce()              | [5/10]  | 120 |
| 15.1.3.102 freduce()              | [6/10]  | 120 |
| 15.1.3.103 freivalds()            |         | 121 |
| 15.1.3.104 fscal()                | [1/10]  | 121 |
| 15.1.3.105 fscal()                | [1/10]  | 122 |
| 15.1.3.106 fscal()                | [2/10]  | 122 |
| 15.1.3.107 fscal()                | [3/10]  | 123 |
| 15.1.3.108 fscal()                | [2/10]  | 123 |
| 15.1.3.109 fscal()                | [3/10]  | 123 |
| 15.1.3.110 fscal()                | [4/10]  | 123 |
| 15.1.3.111 fscal()                | [4/10]  | 124 |
| 15.1.3.112 fscal()                | [5/10]  | 124 |
| 15.1.3.113 fscal()                | [5/10]  | 125 |
| 15.1.3.114 fscal()                | [6/10]  | 125 |
| 15.1.3.115 fscal()                | [6/10]  | 125 |
| 15.1.3.116 fscal()                | [7/10]  | 125 |
| 15.1.3.117 fscal()                | [7/10]  | 126 |
| 15.1.3.118 fscal()                | [8/10]  | 126 |
| 15.1.3.119 fscal()                | [8/10]  | 126 |
| 15.1.3.120 fsyr2k()               |         | 126 |
| 15.1.3.121 fsyrk()                | [1/5]   | 127 |
| 15.1.3.122 fsyrk()                | [2/5]   | 128 |
| 15.1.3.123 fsyrk()                | [3/5]   | 129 |
| 15.1.3.124 fsyrk()                | [4/5]   | 129 |
| 15.1.3.125 fsyrk()                | [5/5]   | 129 |
| 15.1.3.126 ftrmm()                | [1/3]   | 130 |
| 15.1.3.127 ftrmm()                | [2/3]   | 131 |
| 15.1.3.128 ftrsm()                | [1/9]   | 132 |
| 15.1.3.129 ftrsm()                | [2/9]   | 132 |

|  |     |
|--|-----|
| 15.1.3.130 ftrsm() [3/9]                                 | 133 |
| 15.1.3.131 ftrsm() [4/9]                                 | 133 |
| 15.1.3.132 ftrsm() [5/9]                                 | 133 |
| 15.1.3.133 cblas_imptrsm()                               | 134 |
| 15.1.3.134 ftrsv() [1/2]                                 | 134 |
| 15.1.3.135 igemm_()                                      | 134 |
| 15.1.3.136 finit() [5/8]                                 | 135 |
| 15.1.3.137 fconvert() [1/3]                              | 135 |
| 15.1.3.138 fnegin() [1/4]                                | 136 |
| 15.1.3.139 fneg() [1/4]                                  | 136 |
| 15.1.3.140 fzero() [1/4]                                 | 137 |
| 15.1.3.141 frand() [1/2]                                 | 137 |
| 15.1.3.142 fiszero() [1/4]                               | 138 |
| 15.1.3.143 fequal() [1/4]                                | 138 |
| 15.1.3.144 faxpby() [1/2]                                | 138 |
| 15.1.3.145 fdot() [9/11]                                 | 139 |
| 15.1.3.146 fswap() [1/2]                                 | 139 |
| 15.1.3.147 fzero() [2/4]                                 | 140 |
| 15.1.3.148 frand() [2/2]                                 | 140 |
| 15.1.3.149 fequal() [2/4]                                | 141 |
| 15.1.3.150 fiszero() [2/4]                               | 141 |
| 15.1.3.151 fidentity() [1/4]                             | 142 |
| 15.1.3.152 fidentity() [2/4]                             | 142 |
| 15.1.3.153 finit() [6/8]                                 | 142 |
| 15.1.3.154 fconvert() [2/3]                              | 143 |
| 15.1.3.155 fnegin() [2/4]                                | 143 |
| 15.1.3.156 fneg() [2/4]                                  | 144 |
| 15.1.3.157 faxpby() [2/2]                                | 144 |
| 15.1.3.158 fmove() [1/2]                                 | 145 |
| 15.1.3.159 bitsize()                                     | 145 |
| 15.1.3.160 bitsize< Givaro::ZRing< Givaro::Integer > >() | 146 |
| 15.1.3.161 ftrmv()                                       | 146 |
| 15.1.3.162 ftrsm() [6/9]                                 | 146 |
| 15.1.3.163 pfgemm() [1/7]                                | 147 |
| 15.1.3.164 pfgemm_1D_rec()                               | 148 |
| 15.1.3.165 pfgemm_2D_rec()                               | 148 |
| 15.1.3.166 pfgemm_3D_rec()                               | 148 |
| 15.1.3.167 pfgemm_3D_rec2()                              | 149 |
| 15.1.3.168 fgemm() [19/23]                               | 149 |
| 15.1.3.169 ftrsm() [7/9]                                 | 150 |
| 15.1.3.170 ftrsm() [8/9]                                 | 150 |
| 15.1.3.171 fspmv() [1/2]                                 | 150 |



|                                    |     |
|------------------------------------|-----|
| 15.1.3.172 fspmm()                 | 151 |
| 15.1.3.173 sparse_init() [1/16]    | 151 |
| 15.1.3.174 sparse_init() [2/16]    | 151 |
| 15.1.3.175 sparse_delete() [1/12]  | 151 |
| 15.1.3.176 sparse_delete() [2/12]  | 152 |
| 15.1.3.177 sparse_init() [3/16]    | 152 |
| 15.1.3.178 sparse_init() [4/16]    | 152 |
| 15.1.3.179 sparse_delete() [3/12]  | 152 |
| 15.1.3.180 sparse_delete() [4/12]  | 152 |
| 15.1.3.181 sparse_print() [1/3]    | 153 |
| 15.1.3.182 sparse_init() [5/16]    | 153 |
| 15.1.3.183 sparse_init() [6/16]    | 153 |
| 15.1.3.184 sparse_init() [7/16]    | 153 |
| 15.1.3.185 sparse_init() [8/16]    | 154 |
| 15.1.3.186 sparse_delete() [5/12]  | 154 |
| 15.1.3.187 sparse_init() [9/16]    | 154 |
| 15.1.3.188 sparse_init() [10/16]   | 154 |
| 15.1.3.189 sparse_init() [11/16]   | 155 |
| 15.1.3.190 sparse_delete() [6/12]  | 155 |
| 15.1.3.191 sparse_delete() [7/12]  | 155 |
| 15.1.3.192 sparse_init() [12/16]   | 155 |
| 15.1.3.193 sparse_init() [13/16]   | 155 |
| 15.1.3.194 sparse_delete() [8/12]  | 156 |
| 15.1.3.195 sparse_delete() [9/12]  | 156 |
| 15.1.3.196 sparse_print() [2/3]    | 156 |
| 15.1.3.197 sparse_delete() [10/12] | 156 |
| 15.1.3.198 sparse_init() [14/16]   | 156 |
| 15.1.3.199 operator<<()            | 156 |
| 15.1.3.200 readSmsFormat()         | 157 |
| 15.1.3.201 readSprFormat()         | 157 |
| 15.1.3.202 getDataType() [1/4]     | 157 |
| 15.1.3.203 getDataType() [2/4]     | 157 |
| 15.1.3.204 getDataType() [3/4]     | 157 |
| 15.1.3.205 getDataType() [4/4]     | 157 |
| 15.1.3.206 readMachineType()       | 158 |
| 15.1.3.207 readDnsFormat()         | 158 |
| 15.1.3.208 writeDnsFormat()        | 158 |
| 15.1.3.209 fspmv() [2/2]           | 158 |
| 15.1.3.210 sparse_delete() [11/12] | 158 |
| 15.1.3.211 sparse_delete() [12/12] | 159 |
| 15.1.3.212 sparse_print() [3/3]    | 159 |
| 15.1.3.213 sparse_init() [15/16]   | 159 |

|   |     |
|---|-----|
| 15.1.3.214 <code>sparse_init()</code> [16/16] | 159 |
| 15.1.3.215 <code>computeDeviation()</code>    | 159 |
| 15.1.3.216 <code>getStat()</code>             | 160 |
| 15.1.3.217 <code>fflas_delete()</code> [1/4]  | 160 |
| 15.1.3.218 <code>fflas_delete()</code> [2/4]  | 160 |
| 15.1.3.219 <code>fflas_new()</code> [1/7]     | 160 |
| 15.1.3.220 <code>fflas_new()</code> [2/7]     | 160 |
| 15.1.3.221 <code>finit_rns()</code> [1/2]     | 161 |
| 15.1.3.222 <code>finit_trans_rns()</code>     | 161 |
| 15.1.3.223 <code>fconvert_rns()</code> [1/2]  | 161 |
| 15.1.3.224 <code>fconvert_trans_rns()</code>  | 161 |
| 15.1.3.225 <code>fflas_new()</code> [3/7]     | 162 |
| 15.1.3.226 <code>fflas_new()</code> [4/7]     | 162 |
| 15.1.3.227 <code>finit_rns()</code> [2/2]     | 162 |
| 15.1.3.228 <code>fconvert_rns()</code> [2/2]  | 162 |
| 15.1.3.229 <code>freduce()</code> [7/10]      | 162 |
| 15.1.3.230 <code>freduce()</code> [8/10]      | 163 |
| 15.1.3.231 <code>finit()</code> [7/8]         | 163 |
| 15.1.3.232 <code>fconvert()</code> [3/3]      | 164 |
| 15.1.3.233 <code>fnegin()</code> [3/4]        | 164 |
| 15.1.3.234 <code>fneg()</code> [3/4]          | 165 |
| 15.1.3.235 <code>fzero()</code> [3/4]         | 165 |
| 15.1.3.236 <code>fiszero()</code> [3/4]       | 166 |
| 15.1.3.237 <code>fequal()</code> [3/4]        | 166 |
| 15.1.3.238 <code>fassign()</code> [9/10]      | 167 |
| 15.1.3.239 <code>fscalin()</code> [9/10]      | 167 |
| 15.1.3.240 <code>fscal()</code> [9/10]        | 168 |
| 15.1.3.241 <code>faxpy()</code> [5/6]         | 168 |
| 15.1.3.242 <code>fdot()</code> [10/11]        | 169 |
| 15.1.3.243 <code>fswap()</code> [2/2]         | 169 |
| 15.1.3.244 <code>fadd()</code> [5/8]          | 170 |
| 15.1.3.245 <code>fsub()</code> [3/4]          | 170 |
| 15.1.3.246 <code>faddin()</code> [3/4]        | 171 |
| 15.1.3.247 <code>fadd()</code> [6/8]          | 171 |
| 15.1.3.248 <code>fassign()</code> [10/10]     | 171 |
| 15.1.3.249 <code>fzero()</code> [4/4]         | 172 |
| 15.1.3.250 <code>fequal()</code> [4/4]        | 172 |
| 15.1.3.251 <code>fiszero()</code> [4/4]       | 173 |
| 15.1.3.252 <code>fidentity()</code> [3/4]     | 173 |
| 15.1.3.253 <code>fidentity()</code> [4/4]     | 173 |
| 15.1.3.254 <code>freduce()</code> [9/10]      | 173 |
| 15.1.3.255 <code>freduce()</code> [10/10]     | 174 |

|   |     |
|---|-----|
| 15.1.3.256 finit() [8/8]  | 174 |
| 15.1.3.257 fnegin() [4/4]   | 175 |
| 15.1.3.258 fneg() [4/4]   | 175 |
| 15.1.3.259 fscaln() [10/10]   | 176 |
| 15.1.3.260 fscal() [10/10]  | 176 |
| 15.1.3.261 faxpy() [6/6]  | 177 |
| 15.1.3.262 fmove() [2/2]  | 177 |
| 15.1.3.263 fadd() [7/8]   | 178 |
| 15.1.3.264 fsub() [4/4]   | 178 |
| 15.1.3.265 fsubin() [3/3]   | 179 |
| 15.1.3.266 fadd() [8/8]   | 179 |
| 15.1.3.267 faddin() [4/4]   | 180 |
| 15.1.3.268 fgemv() [17/19]  | 180 |
| 15.1.3.269 fger() [12/12]   | 182 |
| 15.1.3.270 ftrsv() [2/2]  | 183 |
| 15.1.3.271 ftrsm() [9/9]  | 183 |
| 15.1.3.272 ftrmm() [3/3]  | 184 |
| 15.1.3.273 fgemm() [20/23]  | 185 |
| 15.1.3.274 fgemm() [21/23]  | 185 |
| 15.1.3.275 fgemm() [22/23]  | 186 |
| 15.1.3.276 fgemm() [23/23]  | 186 |
| 15.1.3.277 fsquare() [6/6]  | 187 |
| 15.1.3.278 BlockCuts() [1/2]  | 187 |
| 15.1.3.279 BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >() | 188 |
| 15.1.3.280 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >()      | 188 |
| 15.1.3.281 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >()      | 188 |
| 15.1.3.282 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >()    | 188 |
| 15.1.3.283 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >()   | 188 |
| 15.1.3.284 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >()   | 189 |
| 15.1.3.285 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >()    | 189 |
| 15.1.3.286 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >()    | 189 |
| 15.1.3.287 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >() | 189 |
| 15.1.3.288 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >()  | 189 |
| 15.1.3.289 BlockCuts() [2/2]  | 190 |
| 15.1.3.290 pfzero()   | 190 |
| 15.1.3.291 pfrand()   | 190 |
| 15.1.3.292 fdot() [11/11]   | 190 |
| 15.1.3.293 pfgemm() [2/7]   | 191 |
| 15.1.3.294 pfgemm() [3/7]   | 191 |
| 15.1.3.295 pfgemm() [4/7]   | 191 |
| 15.1.3.296 pfgemm() [5/7]   | 192 |
| 15.1.3.297 pfgemm() [6/7]   | 192 |

|   |     |
|---|-----|
| 15.1.3.298 pfgemm() [7/7]                       | 193 |
| 15.1.3.299 fgemv() [18/19]                      | 193 |
| 15.1.3.300 fgemv() [19/19]                      | 193 |
| 15.1.3.301 parseArguments()                     | 194 |
| 15.1.3.302 writeCommandString()                 | 194 |
| 15.1.3.303 WriteMatrix() [1/2]                  | 194 |
| 15.1.3.304 preamble()                           | 195 |
| 15.1.3.305 ReadMatrix() [1/2]                   | 195 |
| 15.1.3.306 ReadMatrix() [2/2]                   | 195 |
| 15.1.3.307 WriteMatrix() [2/2]                  | 196 |
| 15.1.3.308 WritePermutation()                   | 196 |
| 15.1.3.309 alignable()                          | 196 |
| 15.1.3.310 alignable< Givaro::Integer * >()     | 197 |
| 15.1.3.311 fflas_new() [5/7]                    | 197 |
| 15.1.3.312 fflas_new() [6/7]                    | 197 |
| 15.1.3.313 fflas_new() [7/7]                    | 197 |
| 15.1.3.314 fflas_delete() [3/4]                 | 197 |
| 15.1.3.315 fflas_delete() [4/4]                 | 197 |
| 15.1.3.316 prefetch()                           | 198 |
| 15.1.3.317 getTLBSize()                         | 198 |
| 15.1.3.318 queryCacheSizes()                    | 198 |
| 15.1.3.319 queryL1CacheSize()                   | 198 |
| 15.1.3.320 queryTopLevelCacheSize()             | 198 |
| 15.1.3.321 getSeed()                            | 198 |
| 15.2 FFLAS::BLAS3 Namespace Reference           | 199 |
| 15.2.1 Function Documentation                   | 200 |
| 15.2.1.1 Bini()                                 | 200 |
| 15.2.1.2 WinoPar()                              | 201 |
| 15.2.1.3 Winograd()                             | 201 |
| 15.2.1.4 WinogradAcc_3_23()                     | 201 |
| 15.2.1.5 WinogradAcc_3_21()                     | 202 |
| 15.2.1.6 WinogradAcc_2_24()                     | 202 |
| 15.2.1.7 WinogradAcc_2_27()                     | 203 |
| 15.2.1.8 WinogradAcc_LR()                       | 203 |
| 15.2.1.9 WinogradAcc_R_S()                      | 203 |
| 15.2.1.10 WinogradAcc_L_S()                     | 204 |
| 15.2.1.11 Winograd_LR_S()                       | 204 |
| 15.2.1.12 Winograd_L_S()                        | 205 |
| 15.2.1.13 Winograd_R_S()                        | 205 |
| 15.3 FFLAS::csr_hyb_details Namespace Reference | 205 |
| 15.4 FFLAS::CuttingStrategy Namespace Reference | 205 |
| 15.4.1 Typedef Documentation                    | 206 |

|   |     |
|---|-----|
| 15.4.1.1 RNSModulus                               | 206 |
| 15.5 FFLAS::details Namespace Reference           | 206 |
| 15.5.1 Function Documentation                     | 207 |
| 15.5.1.1 fadd() [1/5]                             | 207 |
| 15.5.1.2 fadd() [2/5]                             | 208 |
| 15.5.1.3 fadd() [3/5]                             | 208 |
| 15.5.1.4 fadd() [4/5]                             | 208 |
| 15.5.1.5 fadd() [5/5]                             | 209 |
| 15.5.1.6 freduce() [1/4]                          | 209 |
| 15.5.1.7 freduce() [2/4]                          | 209 |
| 15.5.1.8 freduce() [3/4]                          | 209 |
| 15.5.1.9 freduce() [4/4]                          | 210 |
| 15.5.1.10 fscaln() [1/2]                          | 210 |
| 15.5.1.11 fscal() [1/2]                           | 210 |
| 15.5.1.12 fscaln() [2/2]                          | 210 |
| 15.5.1.13 fscal() [2/2]                           | 211 |
| 15.5.1.14 igebb44()                               | 211 |
| 15.5.1.15 igebb24()                               | 211 |
| 15.5.1.16 igebb14()                               | 211 |
| 15.5.1.17 igebb41()                               | 212 |
| 15.5.1.18 igebb21()                               | 212 |
| 15.5.1.19 igebb11()                               | 212 |
| 15.5.1.20 igebp()                                 | 213 |
| 15.5.1.21 pack_lhs()                              | 213 |
| 15.5.1.22 pack_rhs()                              | 213 |
| 15.5.1.23 gebp()                                  | 214 |
| 15.5.1.24 BlockingFactor()                        | 214 |
| 15.6 FFLAS::details_spmv Namespace Reference      | 214 |
| 15.7 FFLAS::ElementCategories Namespace Reference | 214 |
| 15.8 FFLAS::FieldCategories Namespace Reference   | 215 |
| 15.8.1 Detailed Description                       | 215 |
| 15.9 FFLAS::MMHelperAlgo Namespace Reference      | 215 |
| 15.10 FFLAS::ModeCategories Namespace Reference   | 215 |
| 15.10.1 Detailed Description                      | 216 |
| 15.11 FFLAS::ParSeqHelper Namespace Reference     | 216 |
| 15.11.1 Detailed Description                      | 216 |
| 15.12 FFLAS::Protected Namespace Reference        | 216 |
| 15.12.1 Function Documentation                    | 219 |
| 15.12.1.1 computeFactorClassic() [1/3]            | 219 |
| 15.12.1.2 computeFactorClassic() [2/3]            | 219 |
| 15.12.1.3 computeFactorClassic() [3/3]            | 220 |
| 15.12.1.4 DotProdBoundClassic()                   | 220 |

|   |     |
|---|-----|
| 15.12.1.5 TRSMBound() [1/3]                     | 220 |
| 15.12.1.6 TRSMBound() [2/3]                     | 220 |
| 15.12.1.7 TRSMBound() [3/3]                     | 220 |
| 15.12.1.8 fgemm_convert()                       | 221 |
| 15.12.1.9 NeedPreAddReduction() [1/2]           | 221 |
| 15.12.1.10 NeedPreAddReduction() [2/2]          | 221 |
| 15.12.1.11 NeedPreSubReduction() [1/2]          | 221 |
| 15.12.1.12 NeedPreSubReduction() [2/2]          | 222 |
| 15.12.1.13 NeedDoublePreAddReduction() [1/2]    | 222 |
| 15.12.1.14 NeedDoublePreAddReduction() [2/2]    | 222 |
| 15.12.1.15 ScalAndReduce() [1/2]                | 222 |
| 15.12.1.16 ScalAndReduce() [2/2]                | 223 |
| 15.12.1.17 fsquareCommon()                      | 223 |
| 15.12.1.18 WinogradThreshold() [1/4]            | 223 |
| 15.12.1.19 WinogradThreshold() [2/4]            | 223 |
| 15.12.1.20 WinogradThreshold() [3/4]            | 224 |
| 15.12.1.21 WinogradThreshold() [4/4]            | 224 |
| 15.12.1.22 WinogradSteps()                      | 224 |
| 15.12.1.23 DynamicPeeling()                     | 224 |
| 15.12.1.24 DynamicPeeling2()                    | 225 |
| 15.12.1.25 WinogradCalc()                       | 225 |
| 15.12.1.26 fgemv_convert()                      | 226 |
| 15.12.1.27 fger_convert()                       | 226 |
| 15.12.1.28 min_types() [1/7]                    | 226 |
| 15.12.1.29 min_types() [2/7]                    | 226 |
| 15.12.1.30 min_types() [3/7]                    | 226 |
| 15.12.1.31 min_types() [4/7]                    | 227 |
| 15.12.1.32 min_types() [5/7]                    | 227 |
| 15.12.1.33 min_types() [6/7]                    | 227 |
| 15.12.1.34 min_types() [7/7]                    | 227 |
| 15.12.1.35 unfit() [1/4]                        | 227 |
| 15.12.1.36 unfit() [2/4]                        | 227 |
| 15.12.1.37 unfit() [3/4]                        | 227 |
| 15.12.1.38 unfit() [4/4]                        | 228 |
| 15.12.1.39 igemm_colmajor() [1/2]               | 228 |
| 15.12.1.40 igemm_colmajor() [2/2]               | 228 |
| 15.12.1.41 igemm()                              | 228 |
| 15.12.1.42 MatF2MatD_Triangular()               | 229 |
| 15.12.1.43 MatF2MatFI_Triangular()              | 229 |
| 15.13 FFLAS::sell_details Namespace Reference   | 229 |
| 15.14 FFLAS::sparse_details Namespace Reference | 230 |
| 15.14.1 Function Documentation                  | 232 |

|   |     |
|---|-----|
| 15.14.1.1 <code>init_y()</code> [1/2]           | 233 |
| 15.14.1.2 <code>init_y()</code> [2/2]           | 233 |
| 15.14.1.3 <code>fspmv_dispatch()</code> [1/2]   | 233 |
| 15.14.1.4 <code>fspmv_dispatch()</code> [2/2]   | 233 |
| 15.14.1.5 <code>fspmv()</code> [1/12]           | 234 |
| 15.14.1.6 <code>fspmv()</code> [2/12]           | 234 |
| 15.14.1.7 <code>fspmv()</code> [3/12]           | 234 |
| 15.14.1.8 <code>fspmv()</code> [4/12]           | 234 |
| 15.14.1.9 <code>fspmv()</code> [5/12]           | 235 |
| 15.14.1.10 <code>fspmv()</code> [6/12]          | 235 |
| 15.14.1.11 <code>fspmv()</code> [7/12]          | 235 |
| 15.14.1.12 <code>fspmv()</code> [8/12]          | 235 |
| 15.14.1.13 <code>fspmv()</code> [9/12]          | 236 |
| 15.14.1.14 <code>fspmm_dispatch()</code> [1/2]  | 236 |
| 15.14.1.15 <code>fspmm_dispatch()</code> [2/2]  | 236 |
| 15.14.1.16 <code>fspmm()</code> [1/9]           | 237 |
| 15.14.1.17 <code>fspmm()</code> [2/9]           | 237 |
| 15.14.1.18 <code>fspmm()</code> [3/9]           | 237 |
| 15.14.1.19 <code>fspmm()</code> [4/9]           | 237 |
| 15.14.1.20 <code>fspmm()</code> [5/9]           | 238 |
| 15.14.1.21 <code>fspmm()</code> [6/9]           | 238 |
| 15.14.1.22 <code>fspmm()</code> [7/9]           | 238 |
| 15.14.1.23 <code>fspmm()</code> [8/9]           | 238 |
| 15.14.1.24 <code>fspmm()</code> [9/9]           | 239 |
| 15.14.1.25 <code>pfspmm_dispatch()</code> [1/2] | 239 |
| 15.14.1.26 <code>pfspmm_dispatch()</code> [2/2] | 239 |
| 15.14.1.27 <code>pfspmm()</code> [1/9]          | 240 |
| 15.14.1.28 <code>pfspmm()</code> [2/9]          | 240 |
| 15.14.1.29 <code>pfspmm()</code> [3/9]          | 240 |
| 15.14.1.30 <code>pfspmm()</code> [4/9]          | 240 |
| 15.14.1.31 <code>pfspmm()</code> [5/9]          | 241 |
| 15.14.1.32 <code>pfspmm()</code> [6/9]          | 241 |
| 15.14.1.33 <code>pfspmm()</code> [7/9]          | 241 |
| 15.14.1.34 <code>pfspmm()</code> [8/9]          | 241 |
| 15.14.1.35 <code>pfspmm()</code> [9/9]          | 242 |
| 15.14.1.36 <code>pfspmv()</code> [1/6]          | 242 |
| 15.14.1.37 <code>pfspmv()</code> [2/6]          | 242 |
| 15.14.1.38 <code>pfspmv()</code> [3/6]          | 242 |
| 15.14.1.39 <code>pfspmv()</code> [4/6]          | 243 |
| 15.14.1.40 <code>pfspmv()</code> [5/6]          | 243 |
| 15.14.1.41 <code>pfspmv()</code> [6/6]          | 243 |
| 15.14.1.42 <code>fspmv()</code> [10/12]         | 243 |

|  |     |
|--|-----|
| 15.14.1.43 fspmv() [11/12]                           | 244 |
| 15.14.1.44 fspmv() [12/12]                           | 244 |
| 15.15 FFLAS::sparse_details_impl Namespace Reference | 244 |
| 15.15.1 Function Documentation                       | 252 |
| 15.15.1.1 fspmm() [1/15]                             | 253 |
| 15.15.1.2 fspmm() [2/15]                             | 253 |
| 15.15.1.3 fspmm() [3/15]                             | 253 |
| 15.15.1.4 fspmm_simd_aligned() [1/2]                 | 253 |
| 15.15.1.5 fspmm_simd_unaligned() [1/2]               | 254 |
| 15.15.1.6 fspmm_one() [1/4]                          | 254 |
| 15.15.1.7 fspmm_mone() [1/4]                         | 254 |
| 15.15.1.8 fspmm_one_simd_aligned() [1/3]             | 254 |
| 15.15.1.9 fspmm_one_simd_unaligned() [1/3]           | 255 |
| 15.15.1.10 fspmm_mone_simd_aligned() [1/3]           | 255 |
| 15.15.1.11 fspmm_mone_simd_unaligned() [1/3]         | 255 |
| 15.15.1.12 fspmv() [1/21]                            | 255 |
| 15.15.1.13 fspmv() [2/21]                            | 256 |
| 15.15.1.14 fspmv() [3/21]                            | 256 |
| 15.15.1.15 fspmv_one() [1/10]                        | 256 |
| 15.15.1.16 fspmv_mone() [1/10]                       | 256 |
| 15.15.1.17 fspmv_one() [2/10]                        | 256 |
| 15.15.1.18 fspmv_mone() [2/10]                       | 257 |
| 15.15.1.19 pfspmm() [1/18]                           | 257 |
| 15.15.1.20 pfspmm() [2/18]                           | 257 |
| 15.15.1.21 pfspmm() [3/18]                           | 257 |
| 15.15.1.22 pfspmm_one() [1/2]                        | 258 |
| 15.15.1.23 pfspmm_mone() [1/2]                       | 258 |
| 15.15.1.24 pfspmm_one() [2/2]                        | 258 |
| 15.15.1.25 pfspmm_mone() [2/2]                       | 258 |
| 15.15.1.26 pfspmv() [1/18]                           | 259 |
| 15.15.1.27 pfspmv_task()                             | 259 |
| 15.15.1.28 pfspmv() [2/18]                           | 259 |
| 15.15.1.29 pfspmv() [3/18]                           | 259 |
| 15.15.1.30 pfspmv_one() [1/8]                        | 259 |
| 15.15.1.31 pfspmv_mone() [1/8]                       | 260 |
| 15.15.1.32 pfspmv_one() [2/8]                        | 260 |
| 15.15.1.33 pfspmv_mone() [2/8]                       | 260 |
| 15.15.1.34 fspmm() [4/15]                            | 260 |
| 15.15.1.35 fspmm() [5/15]                            | 261 |
| 15.15.1.36 fspmm_simd_aligned() [2/2]                | 261 |
| 15.15.1.37 fspmm_simd_unaligned() [2/2]              | 261 |
| 15.15.1.38 fspmm() [6/15]                            | 261 |



|  |     |
|--|-----|
| 15.15.1.39 fspmm_one() [2/4]                 | 262 |
| 15.15.1.40 fspmm_mone() [2/4]                | 262 |
| 15.15.1.41 fspmm_one_simd_aligned() [2/3]    | 262 |
| 15.15.1.42 fspmm_one_simd_unaligned() [2/3]  | 262 |
| 15.15.1.43 fspmm_mone_simd_aligned() [2/3]   | 263 |
| 15.15.1.44 fspmm_mone_simd_unaligned() [2/3] | 263 |
| 15.15.1.45 fspmv() [4/21]                    | 263 |
| 15.15.1.46 fspmv() [5/21]                    | 263 |
| 15.15.1.47 fspmv() [6/21]                    | 264 |
| 15.15.1.48 fspmv_one() [3/10]                | 264 |
| 15.15.1.49 fspmv_mone() [3/10]               | 264 |
| 15.15.1.50 fspmv_one() [4/10]                | 264 |
| 15.15.1.51 fspmv_mone() [4/10]               | 264 |
| 15.15.1.52 pfspmm() [4/18]                   | 265 |
| 15.15.1.53 pfspmm() [5/18]                   | 265 |
| 15.15.1.54 pfspmm() [6/18]                   | 265 |
| 15.15.1.55 pfspmm() [7/18]                   | 265 |
| 15.15.1.56 pfspmm() [8/18]                   | 266 |
| 15.15.1.57 pfspmm() [9/18]                   | 266 |
| 15.15.1.58 pfspmv() [4/18]                   | 266 |
| 15.15.1.59 pfspmv() [5/18]                   | 266 |
| 15.15.1.60 pfspmv() [6/18]                   | 267 |
| 15.15.1.61 fspmm() [7/15]                    | 267 |
| 15.15.1.62 fspmm() [8/15]                    | 267 |
| 15.15.1.63 fspmm() [9/15]                    | 267 |
| 15.15.1.64 fspmv() [7/21]                    | 268 |
| 15.15.1.65 fspmv() [8/21]                    | 268 |
| 15.15.1.66 fspmv() [9/21]                    | 268 |
| 15.15.1.67 pfspmm() [10/18]                  | 268 |
| 15.15.1.68 pfspmm() [11/18]                  | 268 |
| 15.15.1.69 pfspmm() [12/18]                  | 269 |
| 15.15.1.70 pfspmm() [13/18]                  | 269 |
| 15.15.1.71 pfspmm() [14/18]                  | 269 |
| 15.15.1.72 pfspmm() [15/18]                  | 269 |
| 15.15.1.73 pfspmm_zo() [1/2]                 | 270 |
| 15.15.1.74 pfspmm_zo() [2/2]                 | 270 |
| 15.15.1.75 pfspmv() [7/18]                   | 270 |
| 15.15.1.76 pfspmv() [8/18]                   | 270 |
| 15.15.1.77 pfspmv() [9/18]                   | 271 |
| 15.15.1.78 pfspmv_one() [3/8]                | 271 |
| 15.15.1.79 pfspmv_mone() [3/8]               | 271 |
| 15.15.1.80 pfspmv_one() [4/8]                | 271 |

|  |         |     |
|--|---------|-----|
| 15.15.1.81 pfspmv_mone()               | [4/8]   | 271 |
| 15.15.1.82 fspmm()                     | [10/15] | 272 |
| 15.15.1.83 fspmm()                     | [11/15] | 272 |
| 15.15.1.84 fspmm()                     | [12/15] | 272 |
| 15.15.1.85 fspmm_mone()                | [3/4]   | 272 |
| 15.15.1.86 fspmm_one()                 | [3/4]   | 273 |
| 15.15.1.87 fspmm_mone()                | [4/4]   | 273 |
| 15.15.1.88 fspmm_one()                 | [4/4]   | 273 |
| 15.15.1.89 fspmm_one_simd_aligned()    | [3/3]   | 273 |
| 15.15.1.90 fspmm_one_simd_unaligned()  | [3/3]   | 274 |
| 15.15.1.91 fspmm_mone_simd_aligned()   | [3/3]   | 274 |
| 15.15.1.92 fspmm_mone_simd_unaligned() | [3/3]   | 274 |
| 15.15.1.93 fspmv()                     | [10/21] | 274 |
| 15.15.1.94 fspmv()                     | [11/21] | 275 |
| 15.15.1.95 fspmv()                     | [12/21] | 275 |
| 15.15.1.96 fspmv_one()                 | [5/10]  | 275 |
| 15.15.1.97 fspmv_mone()                | [5/10]  | 275 |
| 15.15.1.98 fspmv_one()                 | [6/10]  | 275 |
| 15.15.1.99 fspmv_mone()                | [6/10]  | 276 |
| 15.15.1.100 pfspmv()                   | [10/18] | 276 |
| 15.15.1.101 pfspmv()                   | [11/18] | 276 |
| 15.15.1.102 pfspmv()                   | [12/18] | 276 |
| 15.15.1.103 pfspmv_one()               | [5/8]   | 276 |
| 15.15.1.104 pfspmv_mone()              | [5/8]   | 277 |
| 15.15.1.105 pfspmv_one()               | [6/8]   | 277 |
| 15.15.1.106 pfspmv_mone()              | [6/8]   | 277 |
| 15.15.1.107 fspmv()                    | [13/21] | 277 |
| 15.15.1.108 fspmv_simd()               | [1/4]   | 277 |
| 15.15.1.109 fspmv()                    | [14/21] | 278 |
| 15.15.1.110 fspmv_simd()               | [2/4]   | 278 |
| 15.15.1.111 fspmv()                    | [15/21] | 278 |
| 15.15.1.112 fspmv_one()                | [7/10]  | 278 |
| 15.15.1.113 fspmv_mone()               | [7/10]  | 278 |
| 15.15.1.114 fspmv_one()                | [8/10]  | 279 |
| 15.15.1.115 fspmv_mone()               | [8/10]  | 279 |
| 15.15.1.116 fspmv_one_simd()           | [1/2]   | 279 |
| 15.15.1.117 fspmv_mone_simd()          | [1/2]   | 279 |
| 15.15.1.118 pfspmmm()                  | [16/18] | 279 |
| 15.15.1.119 pfspmmm()                  | [17/18] | 280 |
| 15.15.1.120 pfspmmm()                  | [18/18] | 280 |
| 15.15.1.121 pfspmv()                   | [13/18] | 280 |
| 15.15.1.122 pfspmv()                   | [14/18] | 280 |

|  |         |     |
|--|---------|-----|
| 15.15.1.123 pfspmv()                               | [15/18] | 281 |
| 15.15.1.124 fspm()                                 | [13/15] | 281 |
| 15.15.1.125 fspm()                                 | [14/15] | 281 |
| 15.15.1.126 fspm()                                 | [15/15] | 281 |
| 15.15.1.127 fspm()                                 | [16/21] | 282 |
| 15.15.1.128 fspm()                                 | [17/21] | 282 |
| 15.15.1.129 fspm()                                 | [18/21] | 282 |
| 15.15.1.130 pfspmv()                               | [16/18] | 282 |
| 15.15.1.131 pfspmv()                               | [17/18] | 282 |
| 15.15.1.132 pfspmv()                               | [18/18] | 283 |
| 15.15.1.133 pfspmv_one()                           | [7/8]   | 283 |
| 15.15.1.134 pfspmv_mone()                          | [7/8]   | 283 |
| 15.15.1.135 pfspmv_one()                           | [8/8]   | 283 |
| 15.15.1.136 pfspmv_mone()                          | [8/8]   | 283 |
| 15.15.1.137 fspm()                                 | [19/21] | 284 |
| 15.15.1.138 fspm_simd()                            | [3/4]   | 284 |
| 15.15.1.139 fspm()                                 | [20/21] | 284 |
| 15.15.1.140 fspm_simd()                            | [4/4]   | 284 |
| 15.15.1.141 fspm()                                 | [21/21] | 284 |
| 15.15.1.142 fspm_one()                             | [9/10]  | 285 |
| 15.15.1.143 fspm_mone()                            | [9/10]  | 285 |
| 15.15.1.144 fspm_one_simd()                        | [2/2]   | 285 |
| 15.15.1.145 fspm_mone_simd()                       | [2/2]   | 285 |
| 15.15.1.146 fspm_one()                             | [10/10] | 285 |
| 15.15.1.147 fspm_mone()                            | [10/10] | 286 |
| 15.16 FFLAS::StrategyParameter Namespace Reference |         | 286 |
| 15.17 FFLAS::StructureHelper Namespace Reference   |         | 286 |
| 15.17.1 Detailed Description                       |         | 286 |
| 15.18 FFLAS::vectorised Namespace Reference        |         | 286 |
| 15.18.1 Function Documentation                     |         | 288 |
| 15.18.1.1 VEC_ADD()                                |         | 288 |
| 15.18.1.2 addp()                                   |         | 288 |
| 15.18.1.3 VEC_SUB()                                |         | 288 |
| 15.18.1.4 subp()                                   |         | 289 |
| 15.18.1.5 add()                                    |         | 289 |
| 15.18.1.6 sub()                                    |         | 289 |
| 15.18.1.7 reduce()                                 | [1/9]   | 289 |
| 15.18.1.8 reduce()                                 | [2/9]   | 289 |
| 15.18.1.9 reduce()                                 | [3/9]   | 290 |
| 15.18.1.10 reduce()                                | [4/9]   | 290 |
| 15.18.1.11 reduce()                                | [5/9]   | 290 |
| 15.18.1.12 reduce()                                | [6/9]   | 290 |

|   |     |
|---|-----|
| 15.18.1.13 <code>reduce()</code> [7/9]                | 290 |
| 15.18.1.14 <code>reduce()</code> [8/9]                | 291 |
| 15.18.1.15 <code>reduce()</code> [9/9]                | 291 |
| 15.18.1.16 <code>modp()</code> [1/2]                  | 291 |
| 15.18.1.17 <code>modp()</code> [2/2]                  | 291 |
| 15.18.1.18 <code>scalp()</code> [1/2]                 | 291 |
| 15.18.1.19 <code>scalp()</code> [2/2]                 | 292 |
| 15.19 FFLAS::vectorised::unswitch Namespace Reference | 292 |
| 15.19.1 Function Documentation                        | 292 |
| 15.19.1.1 <code>modp()</code> [1/2]                   | 292 |
| 15.19.1.2 <code>modp()</code> [2/2]                   | 293 |
| 15.19.1.3 <code>scalp()</code> [1/2]                  | 293 |
| 15.19.1.4 <code>scalp()</code> [2/2]                  | 293 |
| 15.20 FFPACK Namespace Reference                      | 293 |
| 15.20.1 Detailed Description                          | 309 |
| 15.20.2 Typedef Documentation                         | 309 |
| 15.20.2.1 <code>Checker_PLUQ</code>                   | 309 |
| 15.20.2.2 <code>Checker_Det</code>                    | 309 |
| 15.20.2.3 <code>Checker_invert</code>                 | 309 |
| 15.20.2.4 <code>Checker_charpoly</code>               | 309 |
| 15.20.2.5 <code>ForceCheck_PLUQ</code>                | 309 |
| 15.20.2.6 <code>ForceCheck_Det</code>                 | 309 |
| 15.20.2.7 <code>ForceCheck_invert</code>              | 310 |
| 15.20.2.8 <code>ForceCheck_charpoly</code>            | 310 |
| 15.20.3 Function Documentation                        | 310 |
| 15.20.3.1 <code>LAPACKPerm2MathPerm()</code>          | 310 |
| 15.20.3.2 <code>MathPerm2LAPACKPerm()</code>          | 310 |
| 15.20.3.3 <code>applyP()</code> [1/4]                 | 310 |
| 15.20.3.4 <code>applyP()</code> [2/4]                 | 311 |
| 15.20.3.5 <code>applyP()</code> [3/4]                 | 312 |
| 15.20.3.6 <code>MonotonicApplyP()</code>              | 312 |
| 15.20.3.7 <code>fgetrs()</code> [1/4]                 | 313 |
| 15.20.3.8 <code>fgetrs()</code> [2/4]                 | 313 |
| 15.20.3.9 <code>fgesv()</code> [1/4]                  | 314 |
| 15.20.3.10 <code>fgesv()</code> [2/4]                 | 315 |
| 15.20.3.11 <code>fttrtri()</code> [1/2]               | 316 |
| 15.20.3.12 <code>trinv_left()</code> [1/2]            | 316 |
| 15.20.3.13 <code>fttrtm()</code> [1/2]                | 316 |
| 15.20.3.14 <code>fttrstr()</code>                     | 317 |
| 15.20.3.15 <code>ftrssyr2k()</code>                   | 318 |
| 15.20.3.16 <code>fsytrf()</code> [1/3]                | 318 |
| 15.20.3.17 <code>fsytrf()</code> [2/3]                | 319 |

|   |     |
|---|-----|
| 15.20.3.18 fsytrf() [3/3]                   | 319 |
| 15.20.3.19 fsytrf_nonunit() [1/3]           | 319 |
| 15.20.3.20 PLUQ() [1/6]                     | 320 |
| 15.20.3.21 pPLUQ()                          | 320 |
| 15.20.3.22 PLUQ() [2/6]                     | 321 |
| 15.20.3.23 PLUQ() [3/6]                     | 321 |
| 15.20.3.24 LUdivine() [1/4]                 | 321 |
| 15.20.3.25 ColumnEchelonForm() [1/3]        | 322 |
| 15.20.3.26 pColumnEchelonForm()             | 323 |
| 15.20.3.27 ColumnEchelonForm() [2/3]        | 323 |
| 15.20.3.28 RowEchelonForm() [1/3]           | 323 |
| 15.20.3.29 pRowEchelonForm()                | 324 |
| 15.20.3.30 RowEchelonForm() [2/3]           | 324 |
| 15.20.3.31 ReducedColumnEchelonForm() [1/3] | 325 |
| 15.20.3.32 pReducedColumnEchelonForm()      | 325 |
| 15.20.3.33 ReducedColumnEchelonForm() [2/3] | 326 |
| 15.20.3.34 ReducedRowEchelonForm() [1/3]    | 326 |
| 15.20.3.35 pReducedRowEchelonForm()         | 326 |
| 15.20.3.36 ReducedRowEchelonForm() [2/3]    | 327 |
| 15.20.3.37 Invert() [1/4]                   | 327 |
| 15.20.3.38 Invert() [2/4]                   | 328 |
| 15.20.3.39 Invert2() [1/2]                  | 328 |
| 15.20.3.40 CharPoly() [1/8]                 | 329 |
| 15.20.3.41 CharPoly() [2/8]                 | 330 |
| 15.20.3.42 CharPoly() [3/8]                 | 330 |
| 15.20.3.43 MinPoly() [1/4]                  | 331 |
| 15.20.3.44 MinPoly() [2/4]                  | 331 |
| 15.20.3.45 MatVecMinPoly() [1/2]            | 332 |
| 15.20.3.46 Rank() [1/3]                     | 332 |
| 15.20.3.47 pRank()                          | 333 |
| 15.20.3.48 Rank() [2/3]                     | 333 |
| 15.20.3.49 IsSingular() [1/2]               | 333 |
| 15.20.3.50 Det() [1/6]                      | 334 |
| 15.20.3.51 pDet()                           | 334 |
| 15.20.3.52 Det() [2/6]                      | 335 |
| 15.20.3.53 Solve() [1/3]                    | 335 |
| 15.20.3.54 Solve() [2/3]                    | 335 |
| 15.20.3.55 pSolve()                         | 336 |
| 15.20.3.56 RandomNullSpaceVector() [1/3]    | 336 |
| 15.20.3.57 NullSpaceBasis() [1/2]           | 337 |
| 15.20.3.58 RowRankProfile() [1/3]           | 337 |
| 15.20.3.59 pRowRankProfile()                | 338 |

|   |         |     |
|---|---------|-----|
| 15.20.3.60 RowRankProfile()                 | [ 2/3 ] | 338 |
| 15.20.3.61 ColumnRankProfile()              | [ 1/3 ] | 338 |
| 15.20.3.62 pColumnRankProfile()             |         | 339 |
| 15.20.3.63 ColumnRankProfile()              | [ 2/3 ] | 339 |
| 15.20.3.64 RankProfileFromLU()              |         | 339 |
| 15.20.3.65 LeadingSubmatrixRankProfiles()   |         | 340 |
| 15.20.3.66 RowRankProfileSubmatrixIndices() | [ 1/2 ] | 341 |
| 15.20.3.67 ColRankProfileSubmatrixIndices() | [ 1/2 ] | 341 |
| 15.20.3.68 RowRankProfileSubmatrix()        | [ 1/2 ] | 342 |
| 15.20.3.69 ColRankProfileSubmatrix()        | [ 1/2 ] | 343 |
| 15.20.3.70 getTriangular()                  | [ 1/2 ] | 343 |
| 15.20.3.71 getTriangular()                  | [ 2/2 ] | 344 |
| 15.20.3.72 getEchelonForm()                 | [ 1/2 ] | 345 |
| 15.20.3.73 getEchelonForm()                 | [ 2/2 ] | 345 |
| 15.20.3.74 getEchelonTransform()            |         | 346 |
| 15.20.3.75 getReducedEchelonForm()          | [ 1/2 ] | 347 |
| 15.20.3.76 getReducedEchelonForm()          | [ 2/2 ] | 348 |
| 15.20.3.77 getReducedEchelonTransform()     |         | 348 |
| 15.20.3.78 PLUQtoEchelonPermutation()       |         | 349 |
| 15.20.3.79 LQUPtoInverseOfFullRankMinor()   | [ 1/2 ] | 349 |
| 15.20.3.80 RandomNullSpaceVector()          | [ 2/3 ] | 350 |
| 15.20.3.81 solveLB()                        | [ 1/2 ] | 351 |
| 15.20.3.82 solveLB2()                       | [ 1/2 ] | 351 |
| 15.20.3.83 Danilevski()                     |         | 351 |
| 15.20.3.84 buildMatrix()                    |         | 352 |
| 15.20.3.85 CharPoly()                       | [ 4/8 ] | 352 |
| 15.20.3.86 CharPoly()                       | [ 5/8 ] | 352 |
| 15.20.3.87 Det()                            | [ 3/6 ] | 352 |
| 15.20.3.88 Det()                            | [ 4/6 ] | 353 |
| 15.20.3.89 fsytrf_BC_Crout()                |         | 353 |
| 15.20.3.90 fsytrf_BC_RL()                   |         | 353 |
| 15.20.3.91 fsytrf_UP_RPM_BC_RL()            |         | 353 |
| 15.20.3.92 fsytrf_LOW_RPM_BC_Crout()        |         | 354 |
| 15.20.3.93 fsytrf_UP_RPM_BC_Crout()         |         | 354 |
| 15.20.3.94 fsytrf_UP_RPM()                  |         | 354 |
| 15.20.3.95 fsytrf_nonunit()                 | [ 2/3 ] | 354 |
| 15.20.3.96 fsytrf_nonunit()                 | [ 3/3 ] | 355 |
| 15.20.3.97 fsytrf_RPM()                     |         | 355 |
| 15.20.3.98 getTridiagonal()                 |         | 355 |
| 15.20.3.99 LUdivine_gauss()                 | [ 1/2 ] | 355 |
| 15.20.3.100 LUdivine_small()                | [ 1/2 ] | 356 |
| 15.20.3.101 LUdivine()                      | [ 2/4 ] | 356 |

|   |     |
|---|-----|
| 15.20.3.102 LUdivine() [3/4]              | 356 |
| 15.20.3.103 MonotonicCompress()           | 357 |
| 15.20.3.104 MonotonicCompressMorePivots() | 357 |
| 15.20.3.105 MonotonicCompressCycles()     | 357 |
| 15.20.3.106 MonotonicExpand()             | 358 |
| 15.20.3.107 applyP_block()                | 358 |
| 15.20.3.108 doApplyS()                    | 358 |
| 15.20.3.109 MatrixApplyS() [1/3]          | 359 |
| 15.20.3.110 MatrixApplyS() [2/3]          | 359 |
| 15.20.3.111 MatrixApplyS() [3/3]          | 359 |
| 15.20.3.112 PermApplyS()                  | 360 |
| 15.20.3.113 doApplyT()                    | 360 |
| 15.20.3.114 MatrixApplyT() [1/3]          | 360 |
| 15.20.3.115 MatrixApplyT() [2/3]          | 360 |
| 15.20.3.116 MatrixApplyT() [3/3]          | 361 |
| 15.20.3.117 PermApplyT()                  | 361 |
| 15.20.3.118 composePermutationsLLL()      | 361 |
| 15.20.3.119 composePermutationsLLM()      | 362 |
| 15.20.3.120 composePermutationsMLM()      | 362 |
| 15.20.3.121 cyclic_shift_mathPerm()       | 362 |
| 15.20.3.122 cyclic_shift_row_col() [1/2]  | 363 |
| 15.20.3.123 cyclic_shift_row() [1/3]      | 363 |
| 15.20.3.124 cyclic_shift_row() [2/3]      | 363 |
| 15.20.3.125 cyclic_shift_col() [1/3]      | 363 |
| 15.20.3.126 cyclic_shift_col() [2/3]      | 364 |
| 15.20.3.127 PLUQ_basecaseV3()             | 364 |
| 15.20.3.128 PLUQ_basecaseV2()             | 364 |
| 15.20.3.129 PLUQ_basecaseCrout()          | 364 |
| 15.20.3.130 _PLUQ()                       | 365 |
| 15.20.3.131 PLUQ() [4/6]                  | 365 |
| 15.20.3.132 threads_fgemm()               | 365 |
| 15.20.3.133 threads_frsm()                | 365 |
| 15.20.3.134 PLUQ() [5/6]                  | 366 |
| 15.20.3.135 fflas_const_cast() [1/3]      | 366 |
| 15.20.3.136 fflas_const_cast() [2/3]      | 366 |
| 15.20.3.137 cyclic_shift_row_col() [2/2]  | 366 |
| 15.20.3.138 cyclic_shift_row() [3/3]      | 366 |
| 15.20.3.139 cyclic_shift_col() [3/3]      | 367 |
| 15.20.3.140 applyP() [4/4]                | 367 |
| 15.20.3.141 fgetrs() [3/4]                | 367 |
| 15.20.3.142 fgetrs() [4/4]                | 367 |
| 15.20.3.143 fgesv() [3/4]                 | 368 |

|  |     |
|--|-----|
| 15.20.3.144 fgesv() [4/4]                                      | 368 |
| 15.20.3.145 ftrtri() [2/2]                                     | 368 |
| 15.20.3.146 trinv_left() [2/2]                                 | 369 |
| 15.20.3.147 ftrtrm() [2/2]                                     | 369 |
| 15.20.3.148 PLUQ() [6/6]                                       | 369 |
| 15.20.3.149 LUdivine() [4/4]                                   | 369 |
| 15.20.3.150 LUdivine_small() [2/2]                             | 370 |
| 15.20.3.151 LUdivine_gauss() [2/2]                             | 370 |
| 15.20.3.152 RowEchelonForm() [3/3]                             | 370 |
| 15.20.3.153 ReducedRowEchelonForm() [3/3]                      | 371 |
| 15.20.3.154 ColumnEchelonForm() [3/3]                          | 371 |
| 15.20.3.155 ReducedColumnEchelonForm() [3/3]                   | 371 |
| 15.20.3.156 Invert() [3/4]                                     | 371 |
| 15.20.3.157 Invert() [4/4]                                     | 372 |
| 15.20.3.158 Invert2() [2/2]                                    | 372 |
| 15.20.3.159 CharPoly() [6/8]                                   | 372 |
| 15.20.3.160 CharPoly() [7/8]                                   | 372 |
| 15.20.3.161 CharPoly() [8/8]                                   | 373 |
| 15.20.3.162 MinPoly() [3/4]                                    | 373 |
| 15.20.3.163 MinPoly() [4/4]                                    | 373 |
| 15.20.3.164 MatVecMinPoly() [2/2]                              | 373 |
| 15.20.3.165 KrylovElim()                                       | 374 |
| 15.20.3.166 SpecRankProfile()                                  | 374 |
| 15.20.3.167 Rank() [3/3]                                       | 374 |
| 15.20.3.168 IsSingular() [2/2]                                 | 374 |
| 15.20.3.169 Det() [5/6]  | 375 |
| 15.20.3.170 Det() [6/6]  | 375 |
| 15.20.3.171 Solve() [3/3]                                      | 375 |
| 15.20.3.172 solveLB() [2/2]                                    | 375 |
| 15.20.3.173 solveLB2() [2/2]                                   | 376 |
| 15.20.3.174 RandomNullSpaceVector() [3/3]                      | 376 |
| 15.20.3.175 NullSpaceBasis() [2/2]                             | 376 |
| 15.20.3.176 RowRankProfile() [3/3]                             | 376 |
| 15.20.3.177 ColumnRankProfile() [3/3]                          | 377 |
| 15.20.3.178 RowRankProfileSubmatrixIndices() [2/2]             | 377 |
| 15.20.3.179 ColRankProfileSubmatrixIndices() [2/2]             | 377 |
| 15.20.3.180 RowRankProfileSubmatrix() [2/2]                    | 377 |
| 15.20.3.181 ColRankProfileSubmatrix() [2/2]                    | 378 |
| 15.20.3.182 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [1/2]  | 378 |
| 15.20.3.183 getTriangular< FFLAS_FIELD< FFLAS_ELT > >() [2/2]  | 378 |
| 15.20.3.184 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2] | 378 |
| 15.20.3.185 getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [2/2] | 379 |



|   |     |
|---|-----|
| 15.20.3.186 getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >()         | 379 |
| 15.20.3.187 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [1/2] | 379 |
| 15.20.3.188 getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >() [2/2] | 380 |
| 15.20.3.189 getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >()  | 380 |
| 15.20.3.190 LQUPtoInverseOfFullRankMinor() [2/2]                      | 380 |
| 15.20.3.191 fflas_const_cast() [3/3]                                  | 381 |
| 15.20.3.192 failure()   | 381 |
| 15.20.3.193 isOdd() [1/3]   | 381 |
| 15.20.3.194 isOdd() [2/3]   | 381 |
| 15.20.3.195 isOdd() [3/3]   | 381 |
| 15.20.3.196 NonZeroRandomMatrix() [1/2]                               | 381 |
| 15.20.3.197 NonZeroRandomMatrix() [2/2]                               | 382 |
| 15.20.3.198 RandomMatrix() [1/2]                                      | 382 |
| 15.20.3.199 RandomMatrix() [2/2]                                      | 383 |
| 15.20.3.200 RandomTriangularMatrix() [1/2]                            | 384 |
| 15.20.3.201 RandomTriangularMatrix() [2/2]                            | 384 |
| 15.20.3.202 RandInt()   | 385 |
| 15.20.3.203 RandomSymmetricMatrix()                                   | 385 |
| 15.20.3.204 RandomMatrixWithRank() [1/2]                              | 386 |
| 15.20.3.205 RandomMatrixWithRank() [2/2]                              | 386 |
| 15.20.3.206 RandomIndexSubset()                                       | 387 |
| 15.20.3.207 RandomPermutation()                                       | 387 |
| 15.20.3.208 RandomRankProfileMatrix()                                 | 387 |
| 15.20.3.209 swapval()   | 388 |
| 15.20.3.210 RandomSymmetricRankProfileMatrix()                        | 388 |
| 15.20.3.211 RandomMatrixWithRankandRPM() [1/2]                        | 388 |
| 15.20.3.212 RandomMatrixWithRankandRPM() [2/2]                        | 389 |
| 15.20.3.213 RandomSymmetricMatrixWithRankandRPM() [1/2]               | 390 |
| 15.20.3.214 RandomSymmetricMatrixWithRankandRPM() [2/2]               | 390 |
| 15.20.3.215 RandomMatrixWithRankandRandomRPM() [1/2]                  | 391 |
| 15.20.3.216 RandomMatrixWithRankandRandomRPM() [2/2]                  | 391 |
| 15.20.3.217 RandomSymmetricMatrixWithRankandRandomRPM() [1/2]         | 392 |
| 15.20.3.218 RandomSymmetricMatrixWithRankandRandomRPM() [2/2]         | 393 |
| 15.20.3.219 RandomMatrixWithDet() [1/2]                               | 393 |
| 15.20.3.220 RandomMatrixWithDet() [2/2]                               | 394 |
| 15.20.3.221 maxFieldElt()   | 394 |
| 15.20.3.222 maxFieldElt< Givaro::ZRing< Givaro::Integer > >()         | 394 |
| 15.20.3.223 chooseField()   | 394 |
| 15.20.3.224 chooseField< Givaro::ZRing< int32_t > >()                 | 395 |
| 15.20.3.225 chooseField< Givaro::ZRing< int64_t > >()                 | 395 |
| 15.20.3.226 chooseField< Givaro::ZRing< float > >()                   | 395 |
| 15.20.3.227 chooseField< Givaro::ZRing< double > >()                  | 395 |

|   |            |
|---|------------|
| 15.21 FFPACK::Protected Namespace Reference   | 395        |
| 15.21.1 Function Documentation  | 397        |
| 15.21.1.1 LUdivine_construct() [1/2]  | 397        |
| 15.21.1.2 GaussJordan()   | 397        |
| 15.21.1.3 KellerGehrig()  | 398        |
| 15.21.1.4 KGFast()  | 398        |
| 15.21.1.5 KGFast_generalized()  | 398        |
| 15.21.1.6 fgemv_kgf()   | 398        |
| 15.21.1.7 LUKrylov()  | 399        |
| 15.21.1.8 Danilevski()  | 399        |
| 15.21.1.9 RandomKrylovPrecond()   | 399        |
| 15.21.1.10 ArithProg()  | 400        |
| 15.21.1.11 LUKrylov_KGFast()  | 400        |
| 15.21.1.12 MatVecMinPoly()  | 400        |
| 15.21.1.13 Hybrid_KGF_LUK_MinPoly()   | 400        |
| 15.21.1.14 updateD()  | 400        |
| 15.21.1.15 newD()   | 401        |
| 15.21.1.16 CompressRows()   | 401        |
| 15.21.1.17 CompressRowsQK()   | 401        |
| 15.21.1.18 DeCompressRows()   | 401        |
| 15.21.1.19 DeCompressRowsQK()   | 402        |
| 15.21.1.20 CompressRowsQA()   | 402        |
| 15.21.1.21 DeCompressRowsQA()   | 402        |
| 15.21.1.22 LUdivine_construct() [2/2]   | 402        |
| 15.22 Givaro Namespace Reference  | 403        |
| 15.23 MKL_CONFIG Namespace Reference  | 403        |
| 15.24 Reclnt Namespace Reference  | 403        |
| <b>16 Data Structure Documentation</b>  | <b>405</b> |
| 16.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference   | 405        |
| 16.1.1 Member Typedef Documentation   | 405        |
| 16.1.1.1 value  | 405        |
| 16.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference | 405        |
| 16.2.1 Member Typedef Documentation   | 405        |
| 16.2.1.1 value  | 405        |
| 16.3 ArbitraryPrecIntTag Struct Reference   | 405        |
| 16.3.1 Detailed Description   | 406        |
| 16.4 AreEqual< X, Y > Class Template Reference  | 406        |
| 16.4.1 Field Documentation  | 406        |
| 16.4.1.1 value  | 406        |
| 16.5 AreEqual< X, X > Class Template Reference  | 406        |
| 16.5.1 Field Documentation  | 406        |

|  |     |
|--|-----|
| 16.5.1.1 value . . . . .   | 406 |
| 16.6 Argument Struct Reference . . . . .   | 406 |
| 16.6.1 Field Documentation . . . . .   | 407 |
| 16.6.1.1 c . . . . .   | 407 |
| 16.6.1.2 example . . . . .   | 407 |
| 16.6.1.3 helpString . . . . .  | 407 |
| 16.6.1.4 type . . . . .  | 407 |
| 16.6.1.5 data . . . . .  | 407 |
| 16.7 associatedDelayedField< Field > Struct Template Reference . . . . .                               | 407 |
| 16.7.1 Member Typedef Documentation . . . . .  | 407 |
| 16.7.1.1 field . . . . .   | 407 |
| 16.7.1.2 type . . . . .  | 407 |
| 16.8 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference . . . . .  | 407 |
| 16.8.1 Member Typedef Documentation . . . . .  | 408 |
| 16.8.1.1 field . . . . .   | 408 |
| 16.8.1.2 type . . . . .  | 408 |
| 16.9 associatedDelayedField< const Givaro::Modular< T, X > > Struct Template Reference . . . . .       | 408 |
| 16.9.1 Member Typedef Documentation . . . . .  | 408 |
| 16.9.1.1 field . . . . .   | 408 |
| 16.9.1.2 type . . . . .  | 408 |
| 16.10 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference . . . . . | 408 |
| 16.10.1 Member Typedef Documentation . . . . .   | 409 |
| 16.10.1.1 field . . . . .  | 409 |
| 16.10.1.2 type . . . . .   | 409 |
| 16.11 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference . . . . .           | 409 |
| 16.11.1 Member Typedef Documentation . . . . .   | 409 |
| 16.11.1.1 field . . . . .  | 409 |
| 16.11.1.2 type . . . . .   | 409 |
| 16.12 Auto Struct Reference . . . . .  | 409 |
| 16.13 Bini Struct Reference . . . . .  | 409 |
| 16.14 Block Struct Reference . . . . .   | 409 |
| 16.15 callLUdivine_small< Element > Class Template Reference . . . . .                                 | 410 |
| 16.15.1 Member Function Documentation . . . . .  | 410 |
| 16.15.1.1 operator()() . . . . .   | 410 |
| 16.16 callLUdivine_small< double > Class Reference . . . . .   | 410 |
| 16.16.1 Member Function Documentation . . . . .  | 410 |
| 16.16.1.1 operator()() . . . . .   | 410 |
| 16.17 callLUdivine_small< float > Class Reference . . . . .  | 411 |
| 16.17.1 Member Function Documentation . . . . .  | 411 |
| 16.17.1.1 operator()() . . . . .   | 411 |
| 16.18 CharpolyFailed Class Reference . . . . .   | 411 |
| 16.19 Checker_Empty< Field > Struct Template Reference . . . . .                                       | 411 |

|  |     |
|--|-----|
| 16.19.1 Constructor & Destructor Documentation                             | 411 |
| 16.19.1.1 Checker_Empty()  | 411 |
| 16.19.2 Member Function Documentation                                      | 412 |
| 16.19.2.1 check()  | 412 |
| 16.20 CheckerImplem_charpoly< Field, Polynomial > Class Template Reference | 412 |
| 16.20.1 Constructor & Destructor Documentation                             | 412 |
| 16.20.1.1 CheckerImplem_charpoly() [1/2]                                   | 412 |
| 16.20.1.2 CheckerImplem_charpoly() [2/2]                                   | 412 |
| 16.20.1.3 ~CheckerImplem_charpoly()  | 412 |
| 16.20.2 Member Function Documentation                                      | 412 |
| 16.20.2.1 check()  | 412 |
| 16.21 CheckerImplem_Det< Field > Class Template Reference                  | 413 |
| 16.21.1 Constructor & Destructor Documentation                             | 413 |
| 16.21.1.1 CheckerImplem_Det() [1/2]  | 413 |
| 16.21.1.2 CheckerImplem_Det() [2/2]  | 413 |
| 16.21.1.3 ~CheckerImplem_Det()   | 413 |
| 16.21.2 Member Function Documentation                                      | 413 |
| 16.21.2.1 check()  | 413 |
| 16.22 CheckerImplem_fgemm< Field > Class Template Reference                | 414 |
| 16.22.1 Constructor & Destructor Documentation                             | 414 |
| 16.22.1.1 CheckerImplem_fgemm() [1/2]                                      | 414 |
| 16.22.1.2 CheckerImplem_fgemm() [2/2]                                      | 414 |
| 16.22.1.3 ~CheckerImplem_fgemm()   | 414 |
| 16.22.2 Member Function Documentation                                      | 414 |
| 16.22.2.1 check()  | 415 |
| 16.23 CheckerImplem_ftdsm< Field > Class Template Reference                | 415 |
| 16.23.1 Constructor & Destructor Documentation                             | 415 |
| 16.23.1.1 CheckerImplem_ftdsm() [1/2]                                      | 415 |
| 16.23.1.2 CheckerImplem_ftdsm() [2/2]                                      | 415 |
| 16.23.1.3 ~CheckerImplem_ftdsm()   | 415 |
| 16.23.2 Member Function Documentation                                      | 416 |
| 16.23.2.1 check()  | 416 |
| 16.24 CheckerImplem_invert< Field > Class Template Reference               | 416 |
| 16.24.1 Constructor & Destructor Documentation                             | 416 |
| 16.24.1.1 CheckerImplem_invert() [1/2]                                     | 416 |
| 16.24.1.2 CheckerImplem_invert() [2/2]                                     | 416 |
| 16.24.1.3 ~CheckerImplem_invert()  | 416 |
| 16.24.2 Member Function Documentation                                      | 417 |
| 16.24.2.1 check()  | 417 |
| 16.25 CheckerImplem_PLUQ< Field > Class Template Reference                 | 417 |
| 16.25.1 Constructor & Destructor Documentation                             | 417 |
| 16.25.1.1 CheckerImplem_PLUQ() [1/2]                                       | 417 |

|  |     |
|--|-----|
| 16.25.1.2 CheckerImplem_PLUQ() [2/2]                                   | 417 |
| 16.25.1.3 ~CheckerImplem_PLUQ()  | 417 |
| 16.25.2 Member Function Documentation                                  | 417 |
| 16.25.2.1 check()  | 418 |
| 16.26 Classic Struct Reference   | 418 |
| 16.27 Column Struct Reference  | 418 |
| 16.28 CompactElement< Element > Struct Template Reference              | 418 |
| 16.28.1 Member Typedef Documentation                                   | 418 |
| 16.28.1.1 type   | 418 |
| 16.29 CompactElement< double > Struct Reference                        | 418 |
| 16.29.1 Member Typedef Documentation                                   | 419 |
| 16.29.1.1 type   | 419 |
| 16.30 CompactElement< float > Struct Reference                         | 419 |
| 16.30.1 Member Typedef Documentation                                   | 419 |
| 16.30.1.1 type   | 419 |
| 16.31 CompactElement< int16_t > Struct Reference                       | 419 |
| 16.31.1 Member Typedef Documentation                                   | 419 |
| 16.31.1.1 type   | 419 |
| 16.32 CompactElement< int32_t > Struct Reference                       | 419 |
| 16.32.1 Member Typedef Documentation                                   | 419 |
| 16.32.1.1 type   | 419 |
| 16.33 CompactElement< int64_t > Struct Reference                       | 420 |
| 16.33.1 Member Typedef Documentation                                   | 420 |
| 16.33.1.1 type   | 420 |
| 16.34 compatible_data_type< Field > Struct Template Reference          | 420 |
| 16.34.1 Field Documentation  | 420 |
| 16.34.1.1 value  | 420 |
| 16.35 compatible_data_type< Givaro::ZRing< double > > Struct Reference | 420 |
| 16.35.1 Field Documentation  | 420 |
| 16.35.1.1 value  | 420 |
| 16.36 compatible_data_type< Givaro::ZRing< float > > Struct Reference  | 420 |
| 16.36.1 Field Documentation  | 421 |
| 16.36.1.1 value  | 421 |
| 16.37 Compose< H1, H2 > Struct Template Reference                      | 421 |
| 16.37.1 Constructor & Destructor Documentation                         | 421 |
| 16.37.1.1 Compose() [1/5]  | 421 |
| 16.37.1.2 Compose() [2/5]  | 421 |
| 16.37.1.3 Compose() [3/5]  | 421 |
| 16.37.1.4 Compose() [4/5]  | 421 |
| 16.37.1.5 Compose() [5/5]  | 421 |
| 16.37.2 Member Function Documentation                                  | 422 |
| 16.37.2.1 first_component()  | 422 |

|   |     |
|---|-----|
| 16.37.2.2 second_component()  | 422 |
| 16.37.3 Friends And Related Function Documentation                    | 422 |
| 16.37.3.1 operator<<  | 422 |
| 16.38 Const_int_t< n > Class Template Reference                       | 422 |
| 16.39 Const_uint_t< n > Class Template Reference                      | 422 |
| 16.40 Simd128_impl< true, true, false, 2 >::Converter Union Reference | 422 |
| 16.40.1 Field Documentation   | 422 |
| 16.40.1.1 v   | 422 |
| 16.40.1.2 t   | 423 |
| 16.41 Simd128_impl< true, true, false, 4 >::Converter Union Reference | 423 |
| 16.41.1 Field Documentation   | 423 |
| 16.41.1.1 v   | 423 |
| 16.41.1.2 t   | 423 |
| 16.42 Simd128_impl< true, true, false, 8 >::Converter Union Reference | 423 |
| 16.42.1 Field Documentation   | 423 |
| 16.42.1.1 v   | 423 |
| 16.42.1.2 t   | 423 |
| 16.43 Simd128_impl< true, true, true, 2 >::Converter Union Reference  | 423 |
| 16.43.1 Field Documentation   | 424 |
| 16.43.1.1 v   | 424 |
| 16.43.1.2 t   | 424 |
| 16.44 Simd128_impl< true, true, true, 4 >::Converter Union Reference  | 424 |
| 16.44.1 Field Documentation   | 424 |
| 16.44.1.1 v   | 424 |
| 16.44.1.2 t   | 424 |
| 16.45 Simd128_impl< true, true, true, 8 >::Converter Union Reference  | 424 |
| 16.45.1 Field Documentation   | 424 |
| 16.45.1.1 v   | 424 |
| 16.45.1.2 t   | 424 |
| 16.46 Simd256_impl< true, true, false, 2 >::Converter Union Reference | 425 |
| 16.46.1 Field Documentation   | 425 |
| 16.46.1.1 v   | 425 |
| 16.46.1.2 t   | 425 |
| 16.47 Simd256_impl< true, true, false, 4 >::Converter Union Reference | 425 |
| 16.47.1 Field Documentation   | 425 |
| 16.47.1.1 v   | 425 |
| 16.47.1.2 t   | 425 |
| 16.48 Simd256_impl< true, true, false, 8 >::Converter Union Reference | 425 |
| 16.48.1 Field Documentation   | 425 |
| 16.48.1.1 v   | 426 |
| 16.48.1.2 t   | 426 |
| 16.49 Simd256_impl< true, true, true, 2 >::Converter Union Reference  | 426 |

|   |     |
|---|-----|
| 16.49.1 Field Documentation   | 426 |
| 16.49.1.1 v   | 426 |
| 16.49.1.2 t   | 426 |
| 16.50 Simd256_impl< true, true, true, 4 >::Converter Union Reference  | 426 |
| 16.50.1 Field Documentation   | 426 |
| 16.50.1.1 v   | 426 |
| 16.50.1.2 t   | 426 |
| 16.51 Simd256_impl< true, true, true, 8 >::Converter Union Reference  | 427 |
| 16.51.1 Field Documentation   | 427 |
| 16.51.1.1 v   | 427 |
| 16.51.1.2 t   | 427 |
| 16.52 Simd512_impl< true, true, false, 8 >::Converter Union Reference | 427 |
| 16.52.1 Field Documentation   | 427 |
| 16.52.1.1 v   | 427 |
| 16.52.1.2 t   | 427 |
| 16.53 Simd512_impl< true, true, true, 8 >::Converter Union Reference  | 427 |
| 16.53.1 Field Documentation   | 427 |
| 16.53.1.1 v   | 428 |
| 16.53.1.2 t   | 428 |
| 16.54 ConvertTo< T > Struct Template Reference                        | 428 |
| 16.54.1 Detailed Description  | 428 |
| 16.55 Coo< ValT, IdxT > Struct Template Reference                     | 428 |
| 16.55.1 Member Typedef Documentation                                  | 428 |
| 16.55.1.1 Self  | 429 |
| 16.55.2 Constructor & Destructor Documentation                        | 429 |
| 16.55.2.1 Coo() [1/4]   | 429 |
| 16.55.2.2 Coo() [2/4]   | 429 |
| 16.55.2.3 Coo() [3/4]   | 429 |
| 16.55.2.4 Coo() [4/4]   | 429 |
| 16.55.3 Member Function Documentation                                 | 429 |
| 16.55.3.1 operator=() [1/2]   | 429 |
| 16.55.3.2 operator=() [2/2]   | 429 |
| 16.55.4 Field Documentation   | 429 |
| 16.55.4.1 val   | 429 |
| 16.55.4.2 row   | 429 |
| 16.55.4.3 col   | 430 |
| 16.56 Coo< Field > Struct Template Reference                          | 430 |
| 16.56.1 Constructor & Destructor Documentation                        | 430 |
| 16.56.1.1 Coo() [1/4]   | 430 |
| 16.56.1.2 Coo() [2/4]   | 430 |
| 16.56.1.3 Coo() [3/4]   | 430 |
| 16.56.1.4 Coo() [4/4]   | 430 |

|   |     |
|---|-----|
| 16.56.2 Member Function Documentation             | 430 |
| 16.56.2.1 operator=() [1/2]                       | 431 |
| 16.56.2.2 operator=() [2/2]                       | 431 |
| 16.56.3 Field Documentation                       | 431 |
| 16.56.3.1 val                                     | 431 |
| 16.56.3.2 col                                     | 431 |
| 16.56.3.3 row                                     | 431 |
| 16.56.3.4 deleted                                 | 431 |
| 16.57 Coo< ValT, IdxT > Struct Template Reference | 431 |
| 16.57.1 Member Typedef Documentation              | 432 |
| 16.57.1.1 Self                                    | 432 |
| 16.57.2 Constructor & Destructor Documentation    | 432 |
| 16.57.2.1 Coo() [1/4]                             | 432 |
| 16.57.2.2 Coo() [2/4]                             | 432 |
| 16.57.2.3 Coo() [3/4]                             | 432 |
| 16.57.2.4 Coo() [4/4]                             | 432 |
| 16.57.3 Member Function Documentation             | 432 |
| 16.57.3.1 operator=() [1/2]                       | 432 |
| 16.57.3.2 operator=() [2/2]                       | 432 |
| 16.57.4 Field Documentation                       | 432 |
| 16.57.4.1 val                                     | 432 |
| 16.57.4.2 row                                     | 433 |
| 16.57.4.3 col                                     | 433 |
| 16.58 CooMat< Field > Struct Template Reference   | 433 |
| 16.58.1 Field Documentation                       | 433 |
| 16.58.1.1 _coo16                                  | 433 |
| 16.58.1.2 _coo32                                  | 433 |
| 16.58.1.3 _coo64                                  | 433 |
| 16.58.1.4 _coo16_zo                               | 433 |
| 16.58.1.5 _coo32_zo                               | 433 |
| 16.58.1.6 _coo64_zo                               | 433 |
| 16.59 CsrMat< Field > Struct Template Reference   | 434 |
| 16.59.1 Field Documentation                       | 434 |
| 16.59.1.1 _csr16                                  | 434 |
| 16.59.1.2 _csr32                                  | 434 |
| 16.59.1.3 _csr64                                  | 434 |
| 16.59.1.4 _csr16_zo                               | 434 |
| 16.59.1.5 _csr32_zo                               | 434 |
| 16.59.1.6 _csr64_zo                               | 434 |
| 16.60 DefaultBoundedTag Struct Reference          | 434 |
| 16.60.1 Detailed Description                      | 434 |
| 16.61 DefaultTag Struct Reference                 | 435 |



|   |     |
|---|-----|
| 16.61.1 Detailed Description  | 435 |
| 16.62 DelayedTag Struct Reference                                       | 435 |
| 16.62.1 Detailed Description  | 435 |
| 16.63 ElementTraits< Element > Struct Template Reference                | 435 |
| 16.63.1 Detailed Description  | 435 |
| 16.63.2 Member Typedef Documentation                                    | 435 |
| 16.63.2.1 value   | 435 |
| 16.64 ElementTraits< double > Struct Reference                          | 435 |
| 16.64.1 Member Typedef Documentation                                    | 436 |
| 16.64.1.1 value   | 436 |
| 16.65 ElementTraits< FFPACK::rns_double_elt > Struct Reference          | 436 |
| 16.65.1 Member Typedef Documentation                                    | 436 |
| 16.65.1.1 value   | 436 |
| 16.66 ElementTraits< float > Struct Reference                           | 436 |
| 16.66.1 Member Typedef Documentation                                    | 436 |
| 16.66.1.1 value   | 436 |
| 16.67 ElementTraits< Givaro::Integer > Struct Reference                 | 436 |
| 16.67.1 Member Typedef Documentation                                    | 437 |
| 16.67.1.1 value   | 437 |
| 16.68 ElementTraits< int16_t > Struct Reference                         | 437 |
| 16.68.1 Member Typedef Documentation                                    | 437 |
| 16.68.1.1 value   | 437 |
| 16.69 ElementTraits< int32_t > Struct Reference                         | 437 |
| 16.69.1 Member Typedef Documentation                                    | 437 |
| 16.69.1.1 value   | 437 |
| 16.70 ElementTraits< int64_t > Struct Reference                         | 437 |
| 16.70.1 Member Typedef Documentation                                    | 438 |
| 16.70.1.1 value   | 438 |
| 16.71 ElementTraits< int8_t > Struct Reference                          | 438 |
| 16.71.1 Member Typedef Documentation                                    | 438 |
| 16.71.1.1 value   | 438 |
| 16.72 ElementTraits< RecInt::rint< K > > Struct Template Reference      | 438 |
| 16.72.1 Member Typedef Documentation                                    | 438 |
| 16.72.1.1 value   | 438 |
| 16.73 ElementTraits< RecInt::rmint< K, MG > > Struct Template Reference | 438 |
| 16.73.1 Member Typedef Documentation                                    | 439 |
| 16.73.1.1 value   | 439 |
| 16.74 ElementTraits< RecInt::ruint< K > > Struct Template Reference     | 439 |
| 16.74.1 Member Typedef Documentation                                    | 439 |
| 16.74.1.1 value   | 439 |
| 16.75 ElementTraits< uint16_t > Struct Reference                        | 439 |
| 16.75.1 Member Typedef Documentation                                    | 439 |

|  |     |
|--|-----|
| 16.75.1.1 value                                    | 439 |
| 16.76 ElementTraits< uint32_t > Struct Reference   | 439 |
| 16.76.1 Member Typedef Documentation               | 440 |
| 16.76.1.1 value                                    | 440 |
| 16.77 ElementTraits< uint64_t > Struct Reference   | 440 |
| 16.77.1 Member Typedef Documentation               | 440 |
| 16.77.1.1 value                                    | 440 |
| 16.78 ElementTraits< uint8_t > Struct Reference    | 440 |
| 16.78.1 Member Typedef Documentation               | 440 |
| 16.78.1.1 value                                    | 440 |
| 16.79 EllMat< Field > Struct Template Reference    | 440 |
| 16.79.1 Field Documentation                        | 441 |
| 16.79.1.1 _ell16                                   | 441 |
| 16.79.1.2 _ell32                                   | 441 |
| 16.79.1.3 _ell64                                   | 441 |
| 16.79.1.4 _ell16_zo                                | 441 |
| 16.79.1.5 _ell32_zo                                | 441 |
| 16.79.1.6 _ell64_zo                                | 441 |
| 16.80 Failure Class Reference                      | 441 |
| 16.80.1 Detailed Description                       | 442 |
| 16.80.2 Constructor & Destructor Documentation     | 442 |
| 16.80.2.1 Failure()                                | 442 |
| 16.80.3 Member Function Documentation              | 442 |
| 16.80.3.1 operator>() [1/2]                        | 442 |
| 16.80.3.2 operator>() [2/2]                        | 442 |
| 16.80.3.3 setErrorMessage()                        | 443 |
| 16.80.3.4 print()                                  | 443 |
| 16.80.4 Field Documentation                        | 443 |
| 16.80.4.1 _errorMessage                            | 443 |
| 16.81 FailureCharnpolyCheck Class Reference        | 443 |
| 16.82 FailureDetCheck Class Reference              | 443 |
| 16.83 FailureFgemmCheck Class Reference            | 443 |
| 16.84 FailureInvertCheck Class Reference           | 443 |
| 16.85 FailurePLUQCheck Class Reference             | 444 |
| 16.86 FailureTrsmCheck Class Reference             | 444 |
| 16.87 FieldSimd< _Field > Class Template Reference | 444 |
| 16.87.1 Member Typedef Documentation               | 445 |
| 16.87.1.1 Field                                    | 445 |
| 16.87.1.2 Element                                  | 445 |
| 16.87.1.3 simd                                     | 445 |
| 16.87.1.4 vect_t                                   | 445 |
| 16.87.1.5 scalar_t                                 | 445 |

|  |     |
|--|-----|
| 16.87.2 Constructor & Destructor Documentation       | 445 |
| 16.87.2.1 FieldSimd() [1/3]                          | 445 |
| 16.87.2.2 FieldSimd() [2/3]                          | 445 |
| 16.87.2.3 FieldSimd() [3/3]                          | 445 |
| 16.87.3 Member Function Documentation                | 446 |
| 16.87.3.1 operator=() [1/2]                          | 446 |
| 16.87.3.2 operator=() [2/2]                          | 446 |
| 16.87.3.3 init() [1/2]                               | 446 |
| 16.87.3.4 init() [2/2]                               | 446 |
| 16.87.3.5 add() [1/2]                                | 446 |
| 16.87.3.6 add() [2/2]                                | 446 |
| 16.87.3.7 addin()                                    | 446 |
| 16.87.3.8 add_r() [1/2]                              | 446 |
| 16.87.3.9 add_r() [2/2]                              | 447 |
| 16.87.3.10 addin_r()                                 | 447 |
| 16.87.3.11 sub() [1/2]                               | 447 |
| 16.87.3.12 sub() [2/2]                               | 447 |
| 16.87.3.13 subin()                                   | 447 |
| 16.87.3.14 sub_r() [1/2]                             | 447 |
| 16.87.3.15 sub_r() [2/2]                             | 447 |
| 16.87.3.16 subin_r()                                 | 447 |
| 16.87.3.17 zero() [1/2]                              | 448 |
| 16.87.3.18 zero() [2/2]                              | 448 |
| 16.87.3.19 mod()                                     | 448 |
| 16.87.3.20 mul() [1/2]                               | 448 |
| 16.87.3.21 mul() [2/2]                               | 448 |
| 16.87.3.22 mulin()                                   | 448 |
| 16.87.3.23 mul_r() [1/2]                             | 448 |
| 16.87.3.24 mul_r() [2/2]                             | 448 |
| 16.87.3.25 axpy() [1/2]                              | 448 |
| 16.87.3.26 axpy() [2/2]                              | 449 |
| 16.87.3.27 axpyin()                                  | 449 |
| 16.87.3.28 axpy_r() [1/2]                            | 449 |
| 16.87.3.29 axpy_r() [2/2]                            | 449 |
| 16.87.3.30 axpyin_r()                                | 449 |
| 16.87.3.31 maxpy() [1/2]                             | 449 |
| 16.87.3.32 maxpy() [2/2]                             | 449 |
| 16.87.3.33 maxpyin()                                 | 450 |
| 16.87.4 Field Documentation                          | 450 |
| 16.87.4.1 vect_size                                  | 450 |
| 16.87.4.2 alignment                                  | 450 |
| 16.88 FieldTraits< Field > Struct Template Reference | 450 |

|   |     |
|---|-----|
| 16.88.1 Detailed Description  | 450 |
| 16.88.2 Member Typedef Documentation  | 450 |
| 16.88.2.1 category  | 450 |
| 16.88.3 Field Documentation   | 450 |
| 16.88.3.1 balanced  | 451 |
| 16.89 FieldTraits< FFPACK::RNSInteger< T > > Struct Template Reference            | 451 |
| 16.89.1 Member Typedef Documentation  | 451 |
| 16.89.1.1 category  | 451 |
| 16.89.2 Field Documentation   | 451 |
| 16.89.2.1 balanced  | 451 |
| 16.90 FieldTraits< FFPACK::RNSIntegerMod< T > > Struct Template Reference         | 451 |
| 16.90.1 Member Typedef Documentation  | 451 |
| 16.90.1.1 category  | 452 |
| 16.90.2 Field Documentation   | 452 |
| 16.90.2.1 balanced  | 452 |
| 16.91 FieldTraits< Givaro::Modular< Element > > Struct Template Reference         | 452 |
| 16.91.1 Member Typedef Documentation  | 452 |
| 16.91.1.1 category  | 452 |
| 16.91.2 Field Documentation   | 452 |
| 16.91.2.1 balanced  | 452 |
| 16.92 FieldTraits< Givaro::ModularBalanced< Element > > Struct Template Reference | 452 |
| 16.92.1 Member Typedef Documentation  | 453 |
| 16.92.1.1 category  | 453 |
| 16.92.2 Field Documentation   | 453 |
| 16.92.2.1 balanced  | 453 |
| 16.93 FieldTraits< Givaro::ZRing< double > > Struct Reference                     | 453 |
| 16.93.1 Member Typedef Documentation  | 453 |
| 16.93.1.1 category  | 453 |
| 16.93.2 Field Documentation   | 453 |
| 16.93.2.1 balanced  | 453 |
| 16.94 FieldTraits< Givaro::ZRing< float > > Struct Reference                      | 453 |
| 16.94.1 Member Typedef Documentation  | 454 |
| 16.94.1.1 category  | 454 |
| 16.94.2 Field Documentation   | 454 |
| 16.94.2.1 balanced  | 454 |
| 16.95 FieldTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference            | 454 |
| 16.95.1 Member Typedef Documentation  | 454 |
| 16.95.1.1 category  | 454 |
| 16.95.2 Field Documentation   | 454 |
| 16.95.2.1 balanced  | 454 |
| 16.96 FieldTraits< Givaro::ZRing< int16_t > > Struct Reference                    | 455 |
| 16.96.1 Member Typedef Documentation  | 455 |

|  |     |
|--|-----|
| 16.96.1.1 category . . . . .   | 455 |
| 16.96.2 Field Documentation . . . . .  | 455 |
| 16.96.2.1 balanced . . . . .   | 455 |
| 16.97 FieldTraits< Givaro::ZRing< int32_t > > Struct Reference . . . . .                     | 455 |
| 16.97.1 Member Typedef Documentation . . . . .   | 455 |
| 16.97.1.1 category . . . . .   | 455 |
| 16.97.2 Field Documentation . . . . .  | 455 |
| 16.97.2.1 balanced . . . . .   | 456 |
| 16.98 FieldTraits< Givaro::ZRing< int64_t > > Struct Reference . . . . .                     | 456 |
| 16.98.1 Member Typedef Documentation . . . . .   | 456 |
| 16.98.1.1 category . . . . .   | 456 |
| 16.98.2 Field Documentation . . . . .  | 456 |
| 16.98.2.1 balanced . . . . .   | 456 |
| 16.99 FieldTraits< Givaro::ZRing< RecInt::ruint< K > > > Struct Template Reference . . . . . | 456 |
| 16.99.1 Member Typedef Documentation . . . . .   | 456 |
| 16.99.1.1 category . . . . .   | 456 |
| 16.99.2 Field Documentation . . . . .  | 457 |
| 16.99.2.1 balanced . . . . .   | 457 |
| 16.100 FieldTraits< Givaro::ZRing< uint16_t > > Struct Reference . . . . .                   | 457 |
| 16.100.1 Member Typedef Documentation . . . . .  | 457 |
| 16.100.1.1 category . . . . .  | 457 |
| 16.100.2 Field Documentation . . . . .   | 457 |
| 16.100.2.1 balanced . . . . .  | 457 |
| 16.101 FieldTraits< Givaro::ZRing< uint32_t > > Struct Reference . . . . .                   | 457 |
| 16.101.1 Member Typedef Documentation . . . . .  | 457 |
| 16.101.1.1 category . . . . .  | 458 |
| 16.101.2 Field Documentation . . . . .   | 458 |
| 16.101.2.1 balanced . . . . .  | 458 |
| 16.102 FieldTraits< Givaro::ZRing< uint64_t > > Struct Reference . . . . .                   | 458 |
| 16.102.1 Member Typedef Documentation . . . . .  | 458 |
| 16.102.1.1 category . . . . .  | 458 |
| 16.102.2 Field Documentation . . . . .   | 458 |
| 16.102.2.1 balanced . . . . .  | 458 |
| 16.103 Fixed Struct Reference . . . . .  | 458 |
| 16.104 FixedPreIntTag Struct Reference . . . . .   | 458 |
| 16.104.1 Detailed Description . . . . .  | 459 |
| 16.105 ForStrategy1D< blocksize_t, Cut, Param > Struct Template Reference . . . . .          | 459 |
| 16.105.1 Constructor & Destructor Documentation . . . . .                                    | 459 |
| 16.105.1.1 ForStrategy1D() [1/2] . . . . .   | 459 |
| 16.105.1.2 ForStrategy1D() [2/2] . . . . .   | 459 |
| 16.105.2 Member Function Documentation . . . . .   | 459 |
| 16.105.2.1 build() . . . . .   | 460 |

|   |     |
|---|-----|
| 16.105.2.2 initialize()   | 460 |
| 16.105.2.3 isTerminated()   | 460 |
| 16.105.2.4 begin()  | 460 |
| 16.105.2.5 end()  | 460 |
| 16.105.2.6 numblocks()  | 460 |
| 16.105.2.7 blockindex()   | 460 |
| 16.105.2.8 operator++()   | 460 |
| 16.105.3 Field Documentation  | 460 |
| 16.105.3.1 ibeg   | 460 |
| 16.105.3.2 iend   | 460 |
| 16.105.3.3 current  | 460 |
| 16.105.3.4 firstBlockSize   | 461 |
| 16.105.3.5 lastBlockSize  | 461 |
| 16.105.3.6 changeBS   | 461 |
| 16.105.3.7 numBlock   | 461 |
| 16.106 ForStrategy2D< blocksize_t, Cut, Param > Struct Template Reference | 461 |
| 16.106.1 Constructor & Destructor Documentation                           | 462 |
| 16.106.1.1 ForStrategy2D()  | 462 |
| 16.106.2 Member Function Documentation                                    | 462 |
| 16.106.2.1 initialize()   | 462 |
| 16.106.2.2 isTerminated()   | 462 |
| 16.106.2.3 ibegin()   | 462 |
| 16.106.2.4 jbegin()   | 462 |
| 16.106.2.5 iend()   | 462 |
| 16.106.2.6 jend()   | 462 |
| 16.106.2.7 operator++()   | 462 |
| 16.106.2.8 rownumblocks()   | 462 |
| 16.106.2.9 colnumblocks()   | 463 |
| 16.106.2.10 blockindex()  | 463 |
| 16.106.2.11 rowblockindex()   | 463 |
| 16.106.2.12 colblockindex()   | 463 |
| 16.106.3 Friends And Related Function Documentation                       | 463 |
| 16.106.3.1 operator<<   | 463 |
| 16.106.4 Field Documentation  | 463 |
| 16.106.4.1 _ibeg  | 463 |
| 16.106.4.2 _iend  | 463 |
| 16.106.4.3 _jbeg  | 463 |
| 16.106.4.4 _jend  | 463 |
| 16.106.4.5 rowBlockSize   | 463 |
| 16.106.4.6 colBlockSize   | 464 |
| 16.106.4.7 current  | 464 |
| 16.106.4.8 lastRBS  | 464 |

|  |     |
|--|-----|
| 16.106.4.9 lastCBS . . . . .   | 464 |
| 16.106.4.10 changeRBS . . . . .  | 464 |
| 16.106.4.11 changeCBS . . . . .  | 464 |
| 16.106.4.12 numRowsBlock . . . . .   | 464 |
| 16.106.4.13 numColBlock . . . . .  | 464 |
| 16.106.4.14 BLOCKS . . . . .   | 464 |
| 16.107 ftrmmLeftLowerNoTransNonUnit< Element > Class Template Reference . . . . .  | 464 |
| 16.108 ftrmmLeftLowerNoTransUnit< Element > Class Template Reference . . . . .     | 464 |
| 16.109 ftrmmLeftLowerTransNonUnit< Element > Class Template Reference . . . . .    | 465 |
| 16.110 ftrmmLeftLowerTransUnit< Element > Class Template Reference . . . . .       | 465 |
| 16.111 ftrmmLeftUpperNoTransNonUnit< Element > Class Template Reference . . . . .  | 465 |
| 16.112 ftrmmLeftUpperNoTransUnit< Element > Class Template Reference . . . . .     | 465 |
| 16.113 ftrmmLeftUpperTransNonUnit< Element > Class Template Reference . . . . .    | 465 |
| 16.114 ftrmmLeftUpperTransUnit< Element > Class Template Reference . . . . .       | 465 |
| 16.115 ftrmmRightLowerNoTransNonUnit< Element > Class Template Reference . . . . . | 465 |
| 16.116 ftrmmRightLowerNoTransUnit< Element > Class Template Reference . . . . .    | 465 |
| 16.117 ftrmmRightLowerTransNonUnit< Element > Class Template Reference . . . . .   | 466 |
| 16.118 ftrmmRightLowerTransUnit< Element > Class Template Reference . . . . .      | 466 |
| 16.119 ftrmmRightUpperNoTransNonUnit< Element > Class Template Reference . . . . . | 466 |
| 16.120 ftrmmRightUpperNoTransUnit< Element > Class Template Reference . . . . .    | 466 |
| 16.121 ftrmmRightUpperTransNonUnit< Element > Class Template Reference . . . . .   | 466 |
| 16.122 ftrmmRightUpperTransUnit< Element > Class Template Reference . . . . .      | 466 |
| 16.123 ftrsmLeftLowerNoTransNonUnit< Element > Class Template Reference . . . . .  | 466 |
| 16.124 ftrsmLeftLowerNoTransUnit< Element > Class Template Reference . . . . .     | 466 |
| 16.125 ftrsmLeftLowerTransNonUnit< Element > Class Template Reference . . . . .    | 467 |
| 16.126 ftrsmLeftLowerTransUnit< Element > Class Template Reference . . . . .       | 467 |
| 16.127 ftrsmLeftUpperNoTransNonUnit< Element > Class Template Reference . . . . .  | 467 |
| 16.127.1 Detailed Description . . . . .  | 467 |
| 16.128 ftrsmLeftUpperNoTransUnit< Element > Class Template Reference . . . . .     | 467 |
| 16.129 ftrsmLeftUpperTransNonUnit< Element > Class Template Reference . . . . .    | 467 |
| 16.130 ftrsmLeftUpperTransUnit< Element > Class Template Reference . . . . .       | 467 |
| 16.131 ftrsmRightLowerNoTransNonUnit< Element > Class Template Reference . . . . . | 468 |
| 16.132 ftrsmRightLowerNoTransUnit< Element > Class Template Reference . . . . .    | 468 |
| 16.133 ftrsmRightLowerTransNonUnit< Element > Class Template Reference . . . . .   | 468 |
| 16.134 ftrsmRightLowerTransUnit< Element > Class Template Reference . . . . .      | 468 |
| 16.135 ftrsmRightUpperNoTransNonUnit< Element > Class Template Reference . . . . . | 468 |
| 16.136 ftrsmRightUpperNoTransUnit< Element > Class Template Reference . . . . .    | 468 |
| 16.137 ftrsmRightUpperTransNonUnit< Element > Class Template Reference . . . . .   | 468 |
| 16.138 ftrsmRightUpperTransUnit< Element > Class Template Reference . . . . .      | 468 |
| 16.139 GenericTag Struct Reference . . . . .                                       | 469 |
| 16.139.1 Detailed Description . . . . .  | 469 |
| 16.140 GenericTag Struct Reference . . . . .                                       | 469 |

|   |     |
|---|-----|
| 16.140.1 Detailed Description . . . . .   | 469 |
| 16.141 Grain Struct Reference . . . . .   | 469 |
| 16.142 has_minus_eq_impl< C > Struct Template Reference . . . . .                               | 469 |
| 16.142.1 Field Documentation . . . . .  | 469 |
| 16.142.1.1 value . . . . .  | 469 |
| 16.143 has_minus_impl< C > Struct Template Reference . . . . .                                  | 469 |
| 16.143.1 Field Documentation . . . . .  | 470 |
| 16.143.1.1 value . . . . .  | 470 |
| 16.144 has_mul_eq_impl< C > Struct Template Reference . . . . .                                 | 470 |
| 16.144.1 Field Documentation . . . . .  | 470 |
| 16.144.1.1 value . . . . .  | 470 |
| 16.145 has_mul_impl< C > Struct Template Reference . . . . .                                    | 470 |
| 16.145.1 Field Documentation . . . . .  | 470 |
| 16.145.1.1 value . . . . .  | 470 |
| 16.146 has_operation< T > Struct Template Reference . . . . .                                   | 470 |
| 16.146.1 Field Documentation . . . . .  | 471 |
| 16.146.1.1 value . . . . .  | 471 |
| 16.147 has_plus_eq_impl< C > Struct Template Reference . . . . .                                | 471 |
| 16.147.1 Field Documentation . . . . .  | 471 |
| 16.147.1.1 value . . . . .  | 471 |
| 16.148 has_plus_impl< C > Struct Template Reference . . . . .                                   | 471 |
| 16.148.1 Field Documentation . . . . .  | 471 |
| 16.148.1.1 value . . . . .  | 471 |
| 16.149 HelperFlag Struct Reference . . . . .  | 471 |
| 16.149.1 Field Documentation . . . . .  | 472 |
| 16.149.1.1 none . . . . .   | 472 |
| 16.149.1.2 coo . . . . .  | 472 |
| 16.149.1.3 csr . . . . .  | 472 |
| 16.149.1.4 ell . . . . .  | 472 |
| 16.149.1.5 aut . . . . .  | 472 |
| 16.149.1.6 pm1 . . . . .  | 472 |
| 16.150 HelperMod< Field, ElementTraits > Struct Template Reference . . . . .                    | 472 |
| 16.151 HelperMod< Field, ElementCategories::MachineIntTag > Struct Template Reference . . . . . | 472 |
| 16.151.1 Constructor & Destructor Documentation . . . . .                                       | 473 |
| 16.151.1.1 HelperMod() [1/2] . . . . .  | 473 |
| 16.151.1.2 HelperMod() [2/2] . . . . .  | 473 |
| 16.151.2 Field Documentation . . . . .  | 473 |
| 16.151.2.1 p . . . . .  | 473 |
| 16.151.2.2 invp . . . . .   | 473 |
| 16.151.2.3 min . . . . .  | 473 |
| 16.151.2.4 max . . . . .  | 473 |
| 16.151.2.5 pow50rem . . . . .   | 473 |



|   |     |
|---|-----|
| 16.152 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPreIntTag > Struct Template Reference | 474 |
| 16.152.1 Constructor & Destructor Documentation   | 474 |
| 16.152.1.1 HelperMod() [1/2]  | 474 |
| 16.152.1.2 HelperMod() [2/2]  | 474 |
| 16.152.2 Field Documentation  | 474 |
| 16.152.2.1 p  | 474 |
| 16.153 HelperMod< Field, FFLAS::ElementCategories::FixedPreIntTag > Struct Template Reference     | 474 |
| 16.153.1 Constructor & Destructor Documentation   | 474 |
| 16.153.1.1 HelperMod() [1/2]  | 474 |
| 16.153.1.2 HelperMod() [2/2]  | 475 |
| 16.153.2 Field Documentation  | 475 |
| 16.153.2.1 p  | 475 |
| 16.154 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > Struct Template Reference    | 475 |
| 16.154.1 Constructor & Destructor Documentation   | 475 |
| 16.154.1.1 HelperMod() [1/2]  | 475 |
| 16.154.1.2 HelperMod() [2/2]  | 475 |
| 16.154.2 Field Documentation  | 475 |
| 16.154.2.1 p  | 475 |
| 16.154.2.2 invp   | 475 |
| 16.154.2.3 min  | 476 |
| 16.154.2.4 max  | 476 |
| 16.155 Hybrid Struct Reference  | 476 |
| 16.156 Info Struct Reference  | 476 |
| 16.156.1 Constructor & Destructor Documentation   | 476 |
| 16.156.1.1 Info() [1/4]   | 476 |
| 16.156.1.2 Info() [2/4]   | 476 |
| 16.156.1.3 Info() [3/4]   | 476 |
| 16.156.1.4 Info() [4/4]   | 477 |
| 16.156.2 Member Function Documentation  | 477 |
| 16.156.2.1 operator=() [1/2]  | 477 |
| 16.156.2.2 operator=() [2/2]  | 477 |
| 16.156.3 Field Documentation  | 477 |
| 16.156.3.1 size   | 477 |
| 16.156.3.2 perm   | 477 |
| 16.156.3.3 begin  | 477 |
| 16.157 Info Struct Reference  | 477 |
| 16.157.1 Constructor & Destructor Documentation   | 478 |
| 16.157.1.1 Info() [1/4]   | 478 |
| 16.157.1.2 Info() [2/4]   | 478 |
| 16.157.1.3 Info() [3/4]   | 478 |
| 16.157.1.4 Info() [4/4]   | 478 |
| 16.157.2 Member Function Documentation  | 478 |

|  |     |
|--|-----|
| 16.157.2.1 operator=() [1/2]   | 478 |
| 16.157.2.2 operator=() [2/2]   | 478 |
| 16.157.3 Field Documentation   | 478 |
| 16.157.3.1 size  | 478 |
| 16.157.3.2 perm  | 478 |
| 16.157.3.3 begin   | 479 |
| 16.158 is_simd< T > Struct Template Reference  | 479 |
| 16.158.1 Member Typedef Documentation  | 479 |
| 16.158.1.1 type  | 479 |
| 16.158.2 Field Documentation   | 479 |
| 16.158.2.1 value   | 479 |
| 16.159 isSparseMatrix< Field, M > Struct Template Reference  | 479 |
| 16.160 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > > Struct Template Reference           | 480 |
| 16.161 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference        | 480 |
| 16.162 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > > Struct Template Reference           | 480 |
| 16.163 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > > Struct Template Reference       | 480 |
| 16.164 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference        | 481 |
| 16.165 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > > Struct Template Reference           | 481 |
| 16.166 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > > Struct Template Reference      | 481 |
| 16.167 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference   | 482 |
| 16.168 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference        | 482 |
| 16.169 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > > Struct Template Reference        | 482 |
| 16.170 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > > Struct Template Reference          | 483 |
| 16.171 isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference       | 483 |
| 16.172 isSparseMatrixMKLFormat< F, M > Struct Template Reference   | 483 |
| 16.173 isSparseMatrixSimdFormat< F, M > Struct Template Reference  | 483 |
| 16.174 isZOSparseMatrix< F, M > Struct Template Reference  | 484 |
| 16.175 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > > Struct Template Reference      | 484 |
| 16.176 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > > Struct Template Reference      | 484 |
| 16.177 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > Struct Template Reference | 485 |
| 16.178 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > > Struct Template Reference      | 485 |
| 16.179 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > > Struct Template Reference     | 485 |
| 16.180 Iterative Struct Reference  | 485 |
| 16.181 LazyTag Struct Reference  | 486 |
| 16.181.1 Detailed Description  | 486 |
| 16.182 limits< T > Struct Template Reference   | 486 |
| 16.183 limits< char > Struct Reference   | 486 |
| 16.183.1 Member Typedef Documentation  | 486 |
| 16.183.1.1 T   | 486 |

|   |     |
|---|-----|
| 16.183.2 Member Function Documentation . . . . .            | 486 |
| 16.183.2.1 max() . . . . .                                  | 486 |
| 16.183.2.2 min() . . . . .                                  | 486 |
| 16.183.2.3 digits() . . . . .                               | 486 |
| 16.184 limits< double > Struct Reference . . . . .          | 487 |
| 16.184.1 Member Typedef Documentation . . . . .             | 487 |
| 16.184.1.1 T . . . . .                                      | 487 |
| 16.184.2 Member Function Documentation . . . . .            | 487 |
| 16.184.2.1 max() . . . . .                                  | 487 |
| 16.184.2.2 min() . . . . .                                  | 487 |
| 16.184.2.3 digits() . . . . .                               | 487 |
| 16.185 limits< float > Struct Reference . . . . .           | 487 |
| 16.185.1 Member Typedef Documentation . . . . .             | 488 |
| 16.185.1.1 T . . . . .                                      | 488 |
| 16.185.2 Member Function Documentation . . . . .            | 488 |
| 16.185.2.1 max() . . . . .                                  | 488 |
| 16.185.2.2 min() . . . . .                                  | 488 |
| 16.185.2.3 digits() . . . . .                               | 488 |
| 16.186 limits< Givaro::Integer > Struct Reference . . . . . | 488 |
| 16.186.1 Member Typedef Documentation . . . . .             | 488 |
| 16.186.1.1 T . . . . .                                      | 488 |
| 16.186.2 Member Function Documentation . . . . .            | 488 |
| 16.186.2.1 max() . . . . .                                  | 488 |
| 16.186.2.2 min() . . . . .                                  | 489 |
| 16.187 limits< int > Struct Reference . . . . .             | 489 |
| 16.187.1 Member Typedef Documentation . . . . .             | 489 |
| 16.187.1.1 T . . . . .                                      | 489 |
| 16.187.2 Member Function Documentation . . . . .            | 489 |
| 16.187.2.1 max() . . . . .                                  | 489 |
| 16.187.2.2 min() . . . . .                                  | 489 |
| 16.187.2.3 digits() . . . . .                               | 489 |
| 16.188 limits< long > Struct Reference . . . . .            | 489 |
| 16.188.1 Member Typedef Documentation . . . . .             | 490 |
| 16.188.1.1 T . . . . .                                      | 490 |
| 16.188.2 Member Function Documentation . . . . .            | 490 |
| 16.188.2.1 max() . . . . .                                  | 490 |
| 16.188.2.2 min() . . . . .                                  | 490 |
| 16.188.2.3 digits() . . . . .                               | 490 |
| 16.189 limits< long long > Struct Reference . . . . .       | 490 |
| 16.189.1 Member Typedef Documentation . . . . .             | 490 |
| 16.189.1.1 T . . . . .                                      | 490 |
| 16.189.2 Member Function Documentation . . . . .            | 491 |

|   |     |
|---|-----|
| 16.189.2.1 max()  | 491 |
| 16.189.2.2 min()  | 491 |
| 16.189.2.3 digits()   | 491 |
| 16.190 limits< Reclnt::rint< K > > Struct Template Reference  | 491 |
| 16.190.1 Member Typedef Documentation                         | 491 |
| 16.190.1.1 T  | 491 |
| 16.190.2 Member Function Documentation                        | 491 |
| 16.190.2.1 max()  | 491 |
| 16.190.2.2 min()  | 491 |
| 16.191 limits< Reclnt::ruint< K > > Struct Template Reference | 492 |
| 16.191.1 Member Typedef Documentation                         | 492 |
| 16.191.1.1 T  | 492 |
| 16.191.2 Member Function Documentation                        | 492 |
| 16.191.2.1 max()  | 492 |
| 16.191.2.2 min()  | 492 |
| 16.192 limits< short int > Struct Reference                   | 492 |
| 16.192.1 Member Typedef Documentation                         | 492 |
| 16.192.1.1 T  | 492 |
| 16.192.2 Member Function Documentation                        | 493 |
| 16.192.2.1 max()  | 493 |
| 16.192.2.2 min()  | 493 |
| 16.192.2.3 digits()   | 493 |
| 16.193 limits< signed char > Struct Reference                 | 493 |
| 16.193.1 Member Typedef Documentation                         | 493 |
| 16.193.1.1 T  | 493 |
| 16.193.2 Member Function Documentation                        | 493 |
| 16.193.2.1 max()  | 493 |
| 16.193.2.2 min()  | 493 |
| 16.193.2.3 digits()   | 494 |
| 16.194 limits< unsigned char > Struct Reference               | 494 |
| 16.194.1 Member Typedef Documentation                         | 494 |
| 16.194.1.1 T  | 494 |
| 16.194.2 Member Function Documentation                        | 494 |
| 16.194.2.1 max()  | 494 |
| 16.194.2.2 min()  | 494 |
| 16.194.2.3 digits()   | 494 |
| 16.195 limits< unsigned int > Struct Reference                | 494 |
| 16.195.1 Member Typedef Documentation                         | 495 |
| 16.195.1.1 T  | 495 |
| 16.195.2 Member Function Documentation                        | 495 |
| 16.195.2.1 max()  | 495 |
| 16.195.2.2 min()  | 495 |

|   |     |
|---|-----|
| 16.195.2.3 digits()   | 495 |
| 16.196 limits< unsigned long > Struct Reference                                       | 495 |
| 16.196.1 Member Typedef Documentation   | 495 |
| 16.196.1.1 T  | 495 |
| 16.196.2 Member Function Documentation  | 496 |
| 16.196.2.1 max()  | 496 |
| 16.196.2.2 min()  | 496 |
| 16.196.2.3 digits()   | 496 |
| 16.197 limits< unsigned long long > Struct Reference                                  | 496 |
| 16.197.1 Member Typedef Documentation   | 496 |
| 16.197.1.1 T  | 496 |
| 16.197.2 Member Function Documentation  | 496 |
| 16.197.2.1 max()  | 496 |
| 16.197.2.2 min()  | 496 |
| 16.197.2.3 digits()   | 497 |
| 16.198 limits< unsigned short int > Struct Reference                                  | 497 |
| 16.198.1 Member Typedef Documentation   | 497 |
| 16.198.1.1 T  | 497 |
| 16.198.2 Member Function Documentation  | 497 |
| 16.198.2.1 max()  | 497 |
| 16.198.2.2 min()  | 497 |
| 16.198.2.3 digits()   | 497 |
| 16.199 MachineFloatTag Struct Reference   | 497 |
| 16.199.1 Detailed Description   | 498 |
| 16.200 MachineIntTag Struct Reference   | 498 |
| 16.200.1 Detailed Description   | 498 |
| 16.201 MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > Struct Template Reference | 498 |
| 16.201.1 Member Typedef Documentation   | 499 |
| 16.201.1.1 Self_t   | 499 |
| 16.201.1.2 DelayedField_t   | 499 |
| 16.201.1.3 DelayedField   | 499 |
| 16.201.1.4 DFelt  | 499 |
| 16.201.2 Constructor & Destructor Documentation                                       | 499 |
| 16.201.2.1 MMHelper() [1/5]   | 499 |
| 16.201.2.2 MMHelper() [2/5]   | 500 |
| 16.201.2.3 MMHelper() [3/5]   | 500 |
| 16.201.2.4 MMHelper() [4/5]   | 500 |
| 16.201.2.5 MMHelper() [5/5]   | 500 |
| 16.201.3 Member Function Documentation  | 500 |
| 16.201.3.1 initC()  | 500 |
| 16.201.3.2 initA()  | 500 |
| 16.201.3.3 initB()  | 500 |

|             |  |     |
|-------------|--|-----|
| 16.201.3.4  | initOut()  | 500 |
| 16.201.3.5  | MaxDelayedDim()  | 501 |
| 16.201.3.6  | Aunfit()   | 501 |
| 16.201.3.7  | Bunfit()   | 501 |
| 16.201.3.8  | setOutBounds()   | 501 |
| 16.201.3.9  | checkA()   | 501 |
| 16.201.3.10 | checkB()   | 501 |
| 16.201.3.11 | checkOut()   | 501 |
| 16.201.4    | Friends And Related Function Documentation   | 501 |
| 16.201.4.1  | operator<<   | 502 |
| 16.201.5    | Field Documentation  | 502 |
| 16.201.5.1  | recLevel   | 502 |
| 16.201.5.2  | FieldMin   | 502 |
| 16.201.5.3  | FieldMax   | 502 |
| 16.201.5.4  | Amin   | 502 |
| 16.201.5.5  | Amax   | 502 |
| 16.201.5.6  | Bmin   | 502 |
| 16.201.5.7  | Bmax   | 502 |
| 16.201.5.8  | Cmin   | 502 |
| 16.201.5.9  | Cmax   | 502 |
| 16.201.5.10 | Outmin   | 502 |
| 16.201.5.11 | Outmax   | 503 |
| 16.201.5.12 | MaxStorableValue   | 503 |
| 16.201.5.13 | delayedField   | 503 |
| 16.201.5.14 | parseq   | 503 |
| 16.202      | MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait ><br>Struct Template Reference | 503 |
| 16.202.1    | Member Typedef Documentation   | 503 |
| 16.202.1.1  | Self_t   | 504 |
| 16.202.2    | Constructor & Destructor Documentation   | 504 |
| 16.202.2.1  | MMHelper() [1/5]   | 504 |
| 16.202.2.2  | MMHelper() [2/5]   | 504 |
| 16.202.2.3  | MMHelper() [3/5]   | 504 |
| 16.202.2.4  | MMHelper() [4/5]   | 504 |
| 16.202.2.5  | MMHelper() [5/5]   | 504 |
| 16.202.3    | Member Function Documentation  | 504 |
| 16.202.3.1  | setNorm()  | 504 |
| 16.202.4    | Friends And Related Function Documentation   | 504 |
| 16.202.4.1  | operator<<   | 505 |
| 16.202.5    | Field Documentation  | 505 |
| 16.202.5.1  | normA  | 505 |
| 16.202.5.2  | normB  | 505 |

|  |     |
|--|-----|
| 16.202.5.3 recLevel . . . . .  | 505 |
| 16.202.5.4 parseq . . . . .  | 505 |
| 16.203 MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeq←<br>Trait > Struct Template Reference . . . . .             | 505 |
| 16.203.1 Member Typedef Documentation . . . . .  | 506 |
| 16.203.1.1 Self_t . . . . .  | 506 |
| 16.203.2 Constructor & Destructor Documentation . . . . .  | 506 |
| 16.203.2.1 MMHelper() [1/5] . . . . .  | 506 |
| 16.203.2.2 MMHelper() [2/5] . . . . .  | 506 |
| 16.203.2.3 MMHelper() [3/5] . . . . .  | 506 |
| 16.203.2.4 MMHelper() [4/5] . . . . .  | 506 |
| 16.203.2.5 MMHelper() [5/5] . . . . .  | 506 |
| 16.203.3 Member Function Documentation . . . . .   | 506 |
| 16.203.3.1 setNorm() . . . . .   | 506 |
| 16.203.4 Friends And Related Function Documentation . . . . .  | 507 |
| 16.203.4.1 operator<< . . . . .  | 507 |
| 16.203.5 Field Documentation . . . . .   | 507 |
| 16.203.5.1 normA . . . . .   | 507 |
| 16.203.5.2 normB . . . . .   | 507 |
| 16.203.5.3 recLevel . . . . .  | 507 |
| 16.203.5.4 parseq . . . . .  | 507 |
| 16.204 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait > Struct Tem-<br>plate Reference . . . . .                           | 507 |
| 16.204.1 Member Typedef Documentation . . . . .  | 508 |
| 16.204.1.1 Self_t . . . . .  | 508 |
| 16.204.2 Constructor & Destructor Documentation . . . . .  | 508 |
| 16.204.2.1 MMHelper() [1/4] . . . . .  | 508 |
| 16.204.2.2 MMHelper() [2/4] . . . . .  | 508 |
| 16.204.2.3 MMHelper() [3/4] . . . . .  | 508 |
| 16.204.2.4 MMHelper() [4/4] . . . . .  | 508 |
| 16.204.3 Friends And Related Function Documentation . . . . .  | 508 |
| 16.204.3.1 operator<< . . . . .  | 508 |
| 16.204.4 Field Documentation . . . . .   | 508 |
| 16.204.4.1 recLevel . . . . .  | 508 |
| 16.204.4.2 parseq . . . . .  | 509 |
| 16.205 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag<br>>, ParSeqTrait > Struct Template Reference . . . . . | 509 |
| 16.205.1 Member Typedef Documentation . . . . .  | 509 |
| 16.205.1.1 Self_t . . . . .  | 509 |
| 16.205.2 Constructor & Destructor Documentation . . . . .  | 509 |
| 16.205.2.1 MMHelper() [1/5] . . . . .  | 509 |
| 16.205.2.2 MMHelper() [2/5] . . . . .  | 510 |
| 16.205.2.3 MMHelper() [3/5] . . . . .  | 510 |

|  |     |
|--|-----|
| 16.205.2.4 MMHelper() [ 4 / 5 ] . . . . .  | 510 |
| 16.205.2.5 MMHelper() [ 5 / 5 ] . . . . .  | 510 |
| 16.205.3 Member Function Documentation . . . . .   | 510 |
| 16.205.3.1 setNorm() . . . . .   | 510 |
| 16.205.4 Friends And Related Function Documentation . . . . .  | 510 |
| 16.205.4.1 operator<< . . . . .  | 510 |
| 16.205.5 Field Documentation . . . . .   | 510 |
| 16.205.5.1 normA . . . . .   | 510 |
| 16.205.5.2 normB . . . . .   | 511 |
| 16.205.5.3 recLevel . . . . .  | 511 |
| 16.205.5.4 parseq . . . . .  | 511 |
| 16.206 MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference | 511 |
| 16.206.1 Detailed Description . . . . .  | 511 |
| 16.206.2 Member Typedef Documentation . . . . .  | 511 |
| 16.206.2.1 Self_t . . . . .  | 511 |
| 16.206.3 Constructor & Destructor Documentation . . . . .  | 512 |
| 16.206.3.1 MMHelper() [ 1 / 4 ] . . . . .  | 512 |
| 16.206.3.2 MMHelper() [ 2 / 4 ] . . . . .  | 512 |
| 16.206.3.3 MMHelper() [ 3 / 4 ] . . . . .  | 512 |
| 16.206.3.4 MMHelper() [ 4 / 4 ] . . . . .  | 512 |
| 16.206.4 Friends And Related Function Documentation . . . . .  | 512 |
| 16.206.4.1 operator<< . . . . .  | 512 |
| 16.206.5 Field Documentation . . . . .   | 512 |
| 16.206.5.1 recLevel . . . . .  | 512 |
| 16.206.5.2 parseq . . . . .  | 512 |
| 16.207 ModeTraits< Field > Struct Template Reference . . . . .   | 513 |
| 16.207.1 Detailed Description . . . . .  | 513 |
| 16.207.2 Member Typedef Documentation . . . . .  | 513 |
| 16.207.2.1 value . . . . .   | 513 |
| 16.208 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference . . . . .           | 513 |
| 16.208.1 Member Typedef Documentation . . . . .  | 513 |
| 16.208.1.1 value . . . . .   | 513 |
| 16.209 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference . . . . .   | 513 |
| 16.209.1 Member Typedef Documentation . . . . .  | 514 |
| 16.209.1.1 value . . . . .   | 514 |
| 16.210 ModeTraits< Givaro::Modular< int16_t, Compute > > Struct Template Reference . . . . .           | 514 |
| 16.210.1 Member Typedef Documentation . . . . .  | 514 |
| 16.210.1.1 value . . . . .   | 514 |
| 16.211 ModeTraits< Givaro::Modular< int32_t, Compute > > Struct Template Reference . . . . .           | 514 |
| 16.211.1 Member Typedef Documentation . . . . .  | 514 |
| 16.211.1.1 value . . . . .   | 514 |
| 16.212 ModeTraits< Givaro::Modular< int8_t, Compute > > Struct Template Reference . . . . .            | 514 |



|  |     |
|--|-----|
| 16.212.1 Member Typedef Documentation  | 515 |
| 16.212.1.1 value   | 515 |
| 16.213 ModeTraits< Givaro::Modular< ReclInt::ruint< K >, Compute > > Struct Template Reference | 515 |
| 16.213.1 Member Typedef Documentation  | 515 |
| 16.213.1.1 value   | 515 |
| 16.214 ModeTraits< Givaro::Modular< uint16_t, Compute > > Struct Template Reference            | 515 |
| 16.214.1 Member Typedef Documentation  | 515 |
| 16.214.1.1 value   | 515 |
| 16.215 ModeTraits< Givaro::Modular< uint32_t, Compute > > Struct Template Reference            | 516 |
| 16.215.1 Member Typedef Documentation  | 516 |
| 16.215.1.1 value   | 516 |
| 16.216 ModeTraits< Givaro::Modular< uint8_t, Compute > > Struct Template Reference             | 516 |
| 16.216.1 Member Typedef Documentation  | 516 |
| 16.216.1.1 value   | 516 |
| 16.217 ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference              | 516 |
| 16.217.1 Member Typedef Documentation  | 516 |
| 16.217.1.1 value   | 517 |
| 16.218 ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference               | 517 |
| 16.218.1 Member Typedef Documentation  | 517 |
| 16.218.1.1 value   | 517 |
| 16.219 ModeTraits< Givaro::ModularBalanced< int16_t > > Struct Reference                       | 517 |
| 16.219.1 Member Typedef Documentation  | 517 |
| 16.219.1.1 value   | 517 |
| 16.220 ModeTraits< Givaro::ModularBalanced< int32_t > > Struct Reference                       | 517 |
| 16.220.1 Member Typedef Documentation  | 518 |
| 16.220.1.1 value   | 518 |
| 16.221 ModeTraits< Givaro::ModularBalanced< int8_t > > Struct Reference                        | 518 |
| 16.221.1 Member Typedef Documentation  | 518 |
| 16.221.1.1 value   | 518 |
| 16.222 ModeTraits< Givaro::Montgomery< T > > Struct Template Reference                         | 518 |
| 16.222.1 Member Typedef Documentation  | 518 |
| 16.222.1.1 value   | 518 |
| 16.223 ModeTraits< Givaro::ZRing< double > > Struct Reference                                  | 518 |
| 16.223.1 Member Typedef Documentation  | 519 |
| 16.223.1.1 value   | 519 |
| 16.224 ModeTraits< Givaro::ZRing< float > > Struct Reference                                   | 519 |
| 16.224.1 Member Typedef Documentation  | 519 |
| 16.224.1.1 value   | 519 |
| 16.225 ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference                         | 519 |
| 16.225.1 Member Typedef Documentation  | 519 |
| 16.225.1.1 value   | 519 |
| 16.226 ModularBalanced< T > Class Template Reference   | 519 |

|  |     |
|--|-----|
| 16.227 ModularTag Struct Reference . . . . .   | 520 |
| 16.227.1 Detailed Description . . . . .  | 520 |
| 16.228 Montgomery< T > Class Template Reference . . . . .  | 520 |
| 16.229 need_field_characteristic< Field > Struct Template Reference . . . . .                            | 520 |
| 16.229.1 Field Documentation . . . . .   | 520 |
| 16.229.1.1 value . . . . .   | 520 |
| 16.230 need_field_characteristic< Givaro::Modular< Field > > Struct Template Reference . . . . .         | 520 |
| 16.230.1 Field Documentation . . . . .   | 520 |
| 16.230.1.1 value . . . . .   | 520 |
| 16.231 need_field_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference . . . . . | 520 |
| 16.231.1 Field Documentation . . . . .   | 521 |
| 16.231.1.1 value . . . . .   | 521 |
| 16.232 NoSimd< T > Struct Template Reference . . . . .   | 521 |
| 16.232.1 Member Typedef Documentation . . . . .  | 521 |
| 16.232.1.1 vect_t . . . . .  | 521 |
| 16.232.1.2 scalar_t . . . . .  | 521 |
| 16.232.2 Member Function Documentation . . . . .   | 521 |
| 16.232.2.1 type_string() . . . . .   | 521 |
| 16.232.2.2 valid() . . . . .   | 521 |
| 16.232.2.3 compliant() . . . . .   | 522 |
| 16.232.3 Field Documentation . . . . .   | 522 |
| 16.232.3.1 vect_size . . . . .   | 522 |
| 16.233 Parallel< C, P > Struct Template Reference . . . . .  | 522 |
| 16.233.1 Member Typedef Documentation . . . . .  | 522 |
| 16.233.1.1 Cut . . . . .   | 522 |
| 16.233.1.2 Param . . . . .   | 522 |
| 16.233.2 Constructor & Destructor Documentation . . . . .  | 522 |
| 16.233.2.1 Parallel() . . . . .  | 522 |
| 16.233.3 Member Function Documentation . . . . .   | 522 |
| 16.233.3.1 numthreads() . . . . .  | 523 |
| 16.233.3.2 set_numthreads() . . . . .  | 523 |
| 16.233.4 Friends And Related Function Documentation . . . . .  | 523 |
| 16.233.4.1 operator<< . . . . .  | 523 |
| 16.234 RNSInteger< RNS >::RandIter Class Reference . . . . .   | 523 |
| 16.234.1 Constructor & Destructor Documentation . . . . .  | 523 |
| 16.234.1.1 RandIter() . . . . .  | 523 |
| 16.234.2 Member Function Documentation . . . . .   | 523 |
| 16.234.2.1 random() [1/2] . . . . .  | 524 |
| 16.234.2.2 random() [2/2] . . . . .  | 524 |
| 16.234.2.3 operator>() [1/2] . . . . .   | 524 |
| 16.234.2.4 operator>() [2/2] . . . . .   | 524 |
| 16.234.2.5 ring() . . . . .  | 524 |

|  |     |
|--|-----|
| 16.235 RNSIntegerMod< RNS >::Randlter Class Reference              | 524 |
| 16.235.1 Constructor & Destructor Documentation                    | 524 |
| 16.235.1.1 Randlter()  | 525 |
| 16.235.2 Member Function Documentation                             | 525 |
| 16.235.2.1 random() [1/2]  | 525 |
| 16.235.2.2 random() [2/2]  | 525 |
| 16.235.2.3 operator>() [1/2]                                       | 525 |
| 16.235.2.4 operator>() [2/2]                                       | 525 |
| 16.235.2.5 ring()  | 525 |
| 16.236 readMyMachineType< Field, T > Struct Template Reference     | 525 |
| 16.236.1 Member Typedef Documentation                              | 525 |
| 16.236.1.1 Element   | 526 |
| 16.236.1.2 Element_ptr   | 526 |
| 16.236.2 Member Function Documentation                             | 526 |
| 16.236.2.1 operator>()   | 526 |
| 16.237 readMyMachineType< Field, mpz_t > Struct Template Reference | 526 |
| 16.237.1 Member Typedef Documentation                              | 526 |
| 16.237.1.1 Element   | 526 |
| 16.237.1.2 Element_ptr   | 526 |
| 16.237.2 Member Function Documentation                             | 526 |
| 16.237.2.1 operator>()   | 527 |
| 16.238 Recursive Struct Reference                                  | 527 |
| 16.239 Recursive Struct Reference                                  | 527 |
| 16.240 rint< K > Class Template Reference                          | 527 |
| 16.241 rns_double Struct Reference                                 | 527 |
| 16.241.1 Member Typedef Documentation                              | 528 |
| 16.241.1.1 integer   | 528 |
| 16.241.1.2 ModField  | 528 |
| 16.241.1.3 BasisElement  | 528 |
| 16.241.1.4 Element   | 528 |
| 16.241.1.5 Element_ptr   | 529 |
| 16.241.1.6 ConstElement_ptr  | 529 |
| 16.241.2 Constructor & Destructor Documentation                    | 529 |
| 16.241.2.1 rns_double() [1/4]                                      | 529 |
| 16.241.2.2 rns_double() [2/4]                                      | 529 |
| 16.241.2.3 rns_double() [3/4]                                      | 529 |
| 16.241.2.4 rns_double() [4/4]                                      | 529 |
| 16.241.3 Member Function Documentation                             | 529 |
| 16.241.3.1 precompute_cst()  | 529 |
| 16.241.3.2 init() [1/3]  | 529 |
| 16.241.3.3 init() [2/3]  | 530 |
| 16.241.3.4 init_transpose()  | 530 |

|             |  |     |
|-------------|--|-----|
| 16.241.3.5  | <a href="#">convert()</a> [1/2]                        | 530 |
| 16.241.3.6  | <a href="#">convert_transpose()</a>                    | 530 |
| 16.241.3.7  | <a href="#">reduce()</a>                               | 531 |
| 16.241.3.8  | <a href="#">init()</a> [3/3]                           | 531 |
| 16.241.3.9  | <a href="#">convert()</a> [2/2]                        | 531 |
| 16.241.4    | Field Documentation                                    | 531 |
| 16.241.4.1  | <a href="#">_basis</a>                                 | 531 |
| 16.241.4.2  | <a href="#">_basisMax</a>                              | 531 |
| 16.241.4.3  | <a href="#">_negbasis</a>                              | 531 |
| 16.241.4.4  | <a href="#">_invbasis</a>                              | 531 |
| 16.241.4.5  | <a href="#">_field_rns</a>                             | 532 |
| 16.241.4.6  | <a href="#">_M</a>                                     | 532 |
| 16.241.4.7  | <a href="#">_Mi</a>                                    | 532 |
| 16.241.4.8  | <a href="#">_MMi</a>                                   | 532 |
| 16.241.4.9  | <a href="#">_crt_in</a>                                | 532 |
| 16.241.4.10 | <a href="#">_crt_out</a>                               | 532 |
| 16.241.4.11 | <a href="#">_size</a>                                  | 532 |
| 16.241.4.12 | <a href="#">_pbits</a>                                 | 532 |
| 16.241.4.13 | <a href="#">_ldm</a>                                   | 532 |
| 16.241.4.14 | <a href="#">_mi_sum</a>                                | 532 |
| 16.242      | <a href="#">rns_double_elt</a> Struct Reference        | 532 |
| 16.242.1    | Constructor & Destructor Documentation                 | 533 |
| 16.242.1.1  | <a href="#">rns_double_elt()</a> [1/3]                 | 533 |
| 16.242.1.2  | <a href="#">~rns_double_elt()</a>                      | 533 |
| 16.242.1.3  | <a href="#">rns_double_elt()</a> [2/3]                 | 533 |
| 16.242.1.4  | <a href="#">rns_double_elt()</a> [3/3]                 | 533 |
| 16.242.2    | Member Function Documentation                          | 533 |
| 16.242.2.1  | <a href="#">operator&amp;()</a> [1/2]                  | 533 |
| 16.242.2.2  | <a href="#">operator&amp;()</a> [2/2]                  | 534 |
| 16.242.3    | Field Documentation                                    | 534 |
| 16.242.3.1  | <a href="#">_ptr</a>                                   | 534 |
| 16.242.3.2  | <a href="#">_stride</a>                                | 534 |
| 16.242.3.3  | <a href="#">_alloc</a>                                 | 534 |
| 16.243      | <a href="#">rns_double_elt_cstptr</a> Struct Reference | 534 |
| 16.243.1    | Constructor & Destructor Documentation                 | 535 |
| 16.243.1.1  | <a href="#">rns_double_elt_cstptr()</a> [1/5]          | 535 |
| 16.243.1.2  | <a href="#">rns_double_elt_cstptr()</a> [2/5]          | 535 |
| 16.243.1.3  | <a href="#">rns_double_elt_cstptr()</a> [3/5]          | 535 |
| 16.243.1.4  | <a href="#">rns_double_elt_cstptr()</a> [4/5]          | 535 |
| 16.243.1.5  | <a href="#">rns_double_elt_cstptr()</a> [5/5]          | 535 |
| 16.243.2    | Member Function Documentation                          | 535 |
| 16.243.2.1  | <a href="#">operator&amp;()</a> [1/2]                  | 535 |

|   |     |
|---|-----|
| 16.243.2.2 operator*()                          | 535 |
| 16.243.2.3 operator[]() [1/2]                   | 535 |
| 16.243.2.4 operator[]() [2/2]                   | 536 |
| 16.243.2.5 operator++()                         | 536 |
| 16.243.2.6 operator--()                         | 536 |
| 16.243.2.7 operator+()                          | 536 |
| 16.243.2.8 operator-()                          | 536 |
| 16.243.2.9 operator+=()                         | 536 |
| 16.243.2.10 operator-=()                        | 536 |
| 16.243.2.11 operator=()                         | 536 |
| 16.243.2.12 operator<()                         | 536 |
| 16.243.2.13 operator"!=()                       | 536 |
| 16.243.2.14 operator&() [2/2]                   | 536 |
| 16.243.3 Field Documentation                    | 537 |
| 16.243.3.1 other                                | 537 |
| 16.243.3.2 _ptr                                 | 537 |
| 16.243.3.3 _stride                              | 537 |
| 16.243.3.4 _alloc                               | 537 |
| 16.244 rns_double_elt_ptr Struct Reference      | 537 |
| 16.244.1 Constructor & Destructor Documentation | 538 |
| 16.244.1.1 rns_double_elt_ptr() [1/5]           | 538 |
| 16.244.1.2 rns_double_elt_ptr() [2/5]           | 538 |
| 16.244.1.3 rns_double_elt_ptr() [3/5]           | 538 |
| 16.244.1.4 rns_double_elt_ptr() [4/5]           | 538 |
| 16.244.1.5 rns_double_elt_ptr() [5/5]           | 538 |
| 16.244.2 Member Function Documentation          | 538 |
| 16.244.2.1 operator&() [1/2]                    | 538 |
| 16.244.2.2 operator*()                          | 538 |
| 16.244.2.3 operator[]() [1/2]                   | 538 |
| 16.244.2.4 operator[]() [2/2]                   | 539 |
| 16.244.2.5 operator++()                         | 539 |
| 16.244.2.6 operator--()                         | 539 |
| 16.244.2.7 operator+()                          | 539 |
| 16.244.2.8 operator-()                          | 539 |
| 16.244.2.9 operator+=()                         | 539 |
| 16.244.2.10 operator-=()                        | 539 |
| 16.244.2.11 operator=()                         | 539 |
| 16.244.2.12 operator<()                         | 539 |
| 16.244.2.13 operator"!=()                       | 539 |
| 16.244.2.14 operator&() [2/2]                   | 539 |
| 16.244.3 Field Documentation                    | 540 |
| 16.244.3.1 other                                | 540 |

|             |  |     |
|-------------|--|-----|
| 16.244.3.2  | <a href="#">_ptr</a>   | 540 |
| 16.244.3.3  | <a href="#">_stride</a>  | 540 |
| 16.244.3.4  | <a href="#">_alloc</a>   | 540 |
| 16.245      | <a href="#">rns_double_extended Struct Reference</a>             | 540 |
| 16.245.1    | <a href="#">Member Typedef Documentation</a>                     | 541 |
| 16.245.1.1  | <a href="#">integer</a>  | 541 |
| 16.245.1.2  | <a href="#">ModField</a>   | 541 |
| 16.245.1.3  | <a href="#">BasisElement</a>                                     | 541 |
| 16.245.1.4  | <a href="#">Element</a>  | 541 |
| 16.245.1.5  | <a href="#">Element_ptr</a>                                      | 541 |
| 16.245.1.6  | <a href="#">ConstElement_ptr</a>                                 | 541 |
| 16.245.2    | <a href="#">Constructor &amp; Destructor Documentation</a>       | 541 |
| 16.245.2.1  | <a href="#">rns_double_extended()</a> [1/3]                      | 542 |
| 16.245.2.2  | <a href="#">rns_double_extended()</a> [2/3]                      | 542 |
| 16.245.2.3  | <a href="#">rns_double_extended()</a> [3/3]                      | 542 |
| 16.245.3    | <a href="#">Member Function Documentation</a>                    | 542 |
| 16.245.3.1  | <a href="#">precompute_cst()</a>                                 | 542 |
| 16.245.3.2  | <a href="#">init()</a> [1/3]                                     | 542 |
| 16.245.3.3  | <a href="#">init()</a> [2/3]                                     | 542 |
| 16.245.3.4  | <a href="#">convert()</a> [1/2]                                  | 543 |
| 16.245.3.5  | <a href="#">init()</a> [3/3]                                     | 543 |
| 16.245.3.6  | <a href="#">convert()</a> [2/2]                                  | 543 |
| 16.245.3.7  | <a href="#">reduce()</a>   | 543 |
| 16.245.4    | <a href="#">Field Documentation</a>                              | 543 |
| 16.245.4.1  | <a href="#">_basis</a>   | 543 |
| 16.245.4.2  | <a href="#">_basisMax</a>  | 543 |
| 16.245.4.3  | <a href="#">_negbasis</a>  | 543 |
| 16.245.4.4  | <a href="#">_invbasis</a>  | 544 |
| 16.245.4.5  | <a href="#">_field_rns</a>                                       | 544 |
| 16.245.4.6  | <a href="#">_M</a>   | 544 |
| 16.245.4.7  | <a href="#">_Mi</a>  | 544 |
| 16.245.4.8  | <a href="#">_MMi</a>   | 544 |
| 16.245.4.9  | <a href="#">_crt_in</a>  | 544 |
| 16.245.4.10 | <a href="#">_crt_out</a>   | 544 |
| 16.245.4.11 | <a href="#">_size</a>  | 544 |
| 16.245.4.12 | <a href="#">_pbits</a>   | 544 |
| 16.245.4.13 | <a href="#">_ldm</a>   | 544 |
| 16.246      | <a href="#">RNSElementTag Struct Reference</a>                   | 544 |
| 16.246.1    | <a href="#">Detailed Description</a>                             | 545 |
| 16.247      | <a href="#">RNSInteger&lt; RNS &gt; Class Template Reference</a> | 545 |
| 16.247.1    | <a href="#">Member Typedef Documentation</a>                     | 546 |
| 16.247.1.1  | <a href="#">BasisElement</a>                                     | 546 |

|  |     |
|--|-----|
| 16.247.1.2 integer                                   | 546 |
| 16.247.1.3 Element                                   | 546 |
| 16.247.1.4 Element_ptr                               | 546 |
| 16.247.1.5 ConstElement_ptr                          | 546 |
| 16.247.2 Constructor & Destructor Documentation      | 546 |
| 16.247.2.1 RNSInteger() [1/2]                        | 546 |
| 16.247.2.2 RNSInteger() [2/2]                        | 546 |
| 16.247.3 Member Function Documentation               | 546 |
| 16.247.3.1 rns()                                     | 546 |
| 16.247.3.2 size()                                    | 546 |
| 16.247.3.3 isOne()                                   | 547 |
| 16.247.3.4 isMOne()                                  | 547 |
| 16.247.3.5 isZero()                                  | 547 |
| 16.247.3.6 characteristic()                          | 547 |
| 16.247.3.7 cardinality()                             | 547 |
| 16.247.3.8 init() [1/2]                              | 547 |
| 16.247.3.9 init() [2/2]                              | 547 |
| 16.247.3.10 reduce() [1/2]                           | 547 |
| 16.247.3.11 reduce() [2/2]                           | 547 |
| 16.247.3.12 convert()                                | 548 |
| 16.247.3.13 assign()                                 | 548 |
| 16.247.3.14 write() [1/2]                            | 548 |
| 16.247.3.15 write() [2/2]                            | 548 |
| 16.247.4 Field Documentation                         | 548 |
| 16.247.4.1 _rns                                      | 548 |
| 16.247.4.2 one                                       | 548 |
| 16.247.4.3 mOne                                      | 548 |
| 16.247.4.4 zero                                      | 548 |
| 16.248 RNSIntegerMod< RNS > Class Template Reference | 548 |
| 16.248.1 Member Typedef Documentation                | 550 |
| 16.248.1.1 Element                                   | 550 |
| 16.248.1.2 Element_ptr                               | 550 |
| 16.248.1.3 ConstElement_ptr                          | 550 |
| 16.248.1.4 BasisElement                              | 550 |
| 16.248.1.5 ModField                                  | 550 |
| 16.248.1.6 integer                                   | 550 |
| 16.248.2 Constructor & Destructor Documentation      | 550 |
| 16.248.2.1 RNSIntegerMod()                           | 550 |
| 16.248.3 Member Function Documentation               | 550 |
| 16.248.3.1 rns()                                     | 550 |
| 16.248.3.2 delayed()                                 | 551 |
| 16.248.3.3 size()                                    | 551 |

|  |     |
|--|-----|
| 16.248.3.4 isOne()                                 | 551 |
| 16.248.3.5 isMOne()                                | 551 |
| 16.248.3.6 isZero()                                | 551 |
| 16.248.3.7 characteristic() [1/2]                  | 551 |
| 16.248.3.8 characteristic() [2/2]                  | 551 |
| 16.248.3.9 cardinality() [1/2]                     | 551 |
| 16.248.3.10 cardinality() [2/2]                    | 551 |
| 16.248.3.11 minElement()                           | 551 |
| 16.248.3.12 maxElement()                           | 551 |
| 16.248.3.13 init() [1/3]                           | 552 |
| 16.248.3.14 init() [2/3]                           | 552 |
| 16.248.3.15 reduce() [1/2]                         | 552 |
| 16.248.3.16 reduce() [2/2]                         | 552 |
| 16.248.3.17 init() [3/3]                           | 552 |
| 16.248.3.18 convert()                              | 552 |
| 16.248.3.19 assign()                               | 552 |
| 16.248.3.20 add()                                  | 552 |
| 16.248.3.21 sub()                                  | 552 |
| 16.248.3.22 neg()                                  | 553 |
| 16.248.3.23 mul()                                  | 553 |
| 16.248.3.24 axpyin()                               | 553 |
| 16.248.3.25 inv()                                  | 553 |
| 16.248.3.26 areEqual()                             | 553 |
| 16.248.3.27 write() [1/2]                          | 553 |
| 16.248.3.28 write() [2/2]                          | 553 |
| 16.248.3.29 reduce_modp() [1/2]                    | 553 |
| 16.248.3.30 write_matrix()                         | 554 |
| 16.248.3.31 write_matrix_long()                    | 554 |
| 16.248.3.32 reduce_modp() [2/2]                    | 554 |
| 16.248.3.33 reduce_modp_rnsmajor()                 | 554 |
| 16.248.4 Field Documentation                       | 554 |
| 16.248.4.1 _p                                      | 554 |
| 16.248.4.2 _Mi_modp_rns                            | 554 |
| 16.248.4.3 _iM_modp_rns                            | 554 |
| 16.248.4.4 _rns                                    | 554 |
| 16.248.4.5 _F                                      | 555 |
| 16.248.4.6 _RNSdelayed                             | 555 |
| 16.248.4.7 one                                     | 555 |
| 16.248.4.8 mOne                                    | 555 |
| 16.248.4.9 zero                                    | 555 |
| 16.249 rnsRandIter< RNS > Class Template Reference | 555 |
| 16.249.1 Constructor & Destructor Documentation    | 555 |



|   |     |
|---|-----|
| 16.249.1.1 rnsRandIter()  | 555 |
| 16.249.2 Member Function Documentation  | 556 |
| 16.249.2.1 random() [1/2]   | 556 |
| 16.249.2.2 operator>() [1/2]  | 556 |
| 16.249.2.3 operator>() [2/2]  | 556 |
| 16.249.2.4 random() [2/2]   | 556 |
| 16.249.2.5 ring()   | 556 |
| 16.250 Row Struct Reference   | 556 |
| 16.251 ruint< K > Class Template Reference  | 556 |
| 16.252 ScalFunctions< Element, Enable > Struct Template Reference   | 556 |
| 16.253 ScalFunctions< Element, typename enable_if< is_floating_point< Element >::value >::type ><br>Struct Template Reference | 557 |
| 16.253.1 Member Function Documentation  | 557 |
| 16.253.1.1 zero()   | 557 |
| 16.253.1.2 vand()   | 557 |
| 16.253.1.3 vor()  | 557 |
| 16.253.1.4 vxor()   | 558 |
| 16.253.1.5 vandnot()  | 558 |
| 16.253.1.6 ceil()   | 558 |
| 16.253.1.7 floor()  | 558 |
| 16.253.1.8 round()  | 558 |
| 16.253.1.9 add()  | 558 |
| 16.253.1.10 addin()   | 558 |
| 16.253.1.11 sub()   | 558 |
| 16.253.1.12 subin()   | 558 |
| 16.253.1.13 mul()   | 559 |
| 16.253.1.14 mulin()   | 559 |
| 16.253.1.15 div()   | 559 |
| 16.253.1.16 fmadd()   | 559 |
| 16.253.1.17 fmaddin()   | 559 |
| 16.253.1.18 fmsub()   | 559 |
| 16.253.1.19 fmsubin()   | 559 |
| 16.253.1.20 fnmadd()  | 559 |
| 16.253.1.21 fnmaddin()  | 560 |
| 16.253.1.22 lesser()  | 560 |
| 16.253.1.23 lesser_eq()   | 560 |
| 16.253.1.24 greater()   | 560 |
| 16.253.1.25 greater_eq()  | 560 |
| 16.253.1.26 eq()  | 560 |
| 16.254 ScalFunctions< Element, typename enable_if< is_integral< Element >::value >::type > Struct<br>Template Reference       | 560 |
| 16.254.1 Member Function Documentation  | 561 |
| 16.254.1.1 zero()   | 561 |

|   |     |
|---|-----|
| 16.254.1.2 round()                                  | 561 |
| 16.254.1.3 vand()                                   | 561 |
| 16.254.1.4 vor()                                    | 562 |
| 16.254.1.5 vxor()                                   | 562 |
| 16.254.1.6 vandnot()                                | 562 |
| 16.254.1.7 add()                                    | 562 |
| 16.254.1.8 addin()                                  | 562 |
| 16.254.1.9 sub()                                    | 562 |
| 16.254.1.10 subin()                                 | 562 |
| 16.254.1.11 mul()                                   | 562 |
| 16.254.1.12 mullo()                                 | 563 |
| 16.254.1.13 mulhi()                                 | 563 |
| 16.254.1.14 mulx()                                  | 563 |
| 16.254.1.15 fmadd()                                 | 563 |
| 16.254.1.16 fmaddin()                               | 563 |
| 16.254.1.17 fmaddx()                                | 563 |
| 16.254.1.18 fmaddxin()                              | 563 |
| 16.254.1.19 fmsub()                                 | 563 |
| 16.254.1.20 fmsubin()                               | 564 |
| 16.254.1.21 fmsubx()                                | 564 |
| 16.254.1.22 fmsubxin()                              | 564 |
| 16.254.1.23 fnmadd()                                | 564 |
| 16.254.1.24 fnmaddin()                              | 564 |
| 16.254.1.25 fnmaddx()                               | 564 |
| 16.254.1.26 fnmaddxin()                             | 564 |
| 16.254.1.27 sra() [1/2]                             | 564 |
| 16.254.1.28 sra() [2/2]                             | 565 |
| 16.254.1.29 srl()                                   | 565 |
| 16.254.1.30 sll()                                   | 565 |
| 16.254.1.31 lesser()                                | 565 |
| 16.254.1.32 lesser_eq()                             | 565 |
| 16.254.1.33 greater()                               | 565 |
| 16.254.1.34 greater_eq()                            | 565 |
| 16.254.1.35 eq()                                    | 565 |
| 16.255 Sequential Struct Reference                  | 566 |
| 16.255.1 Constructor & Destructor Documentation     | 566 |
| 16.255.1.1 Sequential() [1/3]                       | 566 |
| 16.255.1.2 Sequential() [2/3]                       | 566 |
| 16.255.1.3 Sequential() [3/3]                       | 566 |
| 16.255.2 Member Function Documentation              | 566 |
| 16.255.2.1 numthreads()                             | 566 |
| 16.255.3 Friends And Related Function Documentation | 566 |

|   |     |
|---|-----|
| 16.255.3.1 operator<< . . . . .   | 566 |
| 16.256 Simd128_impl< ArithType, Int, Signed, Size > Struct Template Reference . . . . . | 567 |
| 16.257 Simd128_impl< true, false, true, 4 > Struct Reference . . . . .                  | 567 |
| 16.257.1 Member Function Documentation . . . . .  | 567 |
| 16.257.1.1 type_string() . . . . .  | 567 |
| 16.258 Simd128_impl< true, false, true, 8 > Struct Reference . . . . .                  | 567 |
| 16.258.1 Member Function Documentation . . . . .  | 567 |
| 16.258.1.1 type_string() . . . . .  | 567 |
| 16.259 Simd128_impl< true, true, false, 2 > Struct Reference . . . . .                  | 568 |
| 16.259.1 Member Typedef Documentation . . . . .   | 569 |
| 16.259.1.1 scalar_t . . . . .   | 569 |
| 16.259.1.2 vect_t . . . . .   | 570 |
| 16.259.2 Member Function Documentation . . . . .  | 570 |
| 16.259.2.1 set1() [1/2] . . . . .   | 570 |
| 16.259.2.2 set() [1/2] . . . . .  | 570 |
| 16.259.2.3 gather() [1/2] . . . . .   | 570 |
| 16.259.2.4 load() [1/2] . . . . .   | 570 |
| 16.259.2.5 loadu() [1/2] . . . . .  | 570 |
| 16.259.2.6 store() [1/2] . . . . .  | 570 |
| 16.259.2.7 storeu() [1/2] . . . . .   | 570 |
| 16.259.2.8 stream() [1/2] . . . . .   | 571 |
| 16.259.2.9 sra() . . . . .  | 571 |
| 16.259.2.10 greater() . . . . .   | 571 |
| 16.259.2.11 lesser() . . . . .  | 571 |
| 16.259.2.12 greater_eq() . . . . .  | 571 |
| 16.259.2.13 lesser_eq() . . . . .   | 571 |
| 16.259.2.14 mulhi() . . . . .   | 571 |
| 16.259.2.15 mulx() . . . . .  | 571 |
| 16.259.2.16 fmaddx() . . . . .  | 571 |
| 16.259.2.17 fmaddxin() . . . . .  | 572 |
| 16.259.2.18 fnmaddx() . . . . .   | 572 |
| 16.259.2.19 fnmaddxin() . . . . .   | 572 |
| 16.259.2.20 fmsubx() . . . . .  | 572 |
| 16.259.2.21 fmsubxin() . . . . .  | 572 |
| 16.259.2.22 hadd_to_scal() . . . . .  | 572 |
| 16.259.2.23 valid() . . . . .   | 572 |
| 16.259.2.24 compliant() . . . . .   | 573 |
| 16.259.2.25 set1() [2/2] . . . . .  | 573 |
| 16.259.2.26 set() [2/2] . . . . .   | 573 |
| 16.259.2.27 gather() [2/2] . . . . .  | 573 |
| 16.259.2.28 load() [2/2] . . . . .  | 573 |
| 16.259.2.29 loadu() [2/2] . . . . .   | 573 |

|  |     |
|--|-----|
| 16.259.2.30 store() [2/2]                                    | 573 |
| 16.259.2.31 storeu() [2/2]                                   | 573 |
| 16.259.2.32 stream() [2/2]                                   | 574 |
| 16.259.2.33 sll()  | 574 |
| 16.259.2.34 srl()  | 574 |
| 16.259.2.35 shuffle()  | 574 |
| 16.259.2.36 unpacklo()                                       | 574 |
| 16.259.2.37 unpackhi()                                       | 574 |
| 16.259.2.38 blend()  | 574 |
| 16.259.2.39 add()  | 574 |
| 16.259.2.40 addin()  | 574 |
| 16.259.2.41 sub()  | 575 |
| 16.259.2.42 subin()  | 575 |
| 16.259.2.43 mullo()  | 575 |
| 16.259.2.44 mul()  | 575 |
| 16.259.2.45 fmadd()  | 575 |
| 16.259.2.46 fmaddin()  | 575 |
| 16.259.2.47 fnmadd()   | 575 |
| 16.259.2.48 fnmaddin()                                       | 575 |
| 16.259.2.49 fmsub()  | 576 |
| 16.259.2.50 fmsubin()  | 576 |
| 16.259.2.51 eq()   | 576 |
| 16.259.2.52 round()  | 576 |
| 16.259.2.53 mod()  | 576 |
| 16.259.2.54 type_string()                                    | 576 |
| 16.259.2.55 zero()   | 576 |
| 16.259.2.56 sll128()   | 576 |
| 16.259.2.57 srl128()   | 577 |
| 16.259.2.58 vand()   | 577 |
| 16.259.2.59 vor()  | 577 |
| 16.259.2.60 vxor()   | 577 |
| 16.259.2.61 vandnot()  | 577 |
| 16.259.3 Field Documentation                                 | 577 |
| 16.259.3.1 vect_size   | 577 |
| 16.259.3.2 alignment   | 577 |
| 16.260 Simd128_impl< true, true, false, 4 > Struct Reference | 577 |
| 16.260.1 Member Typedef Documentation                        | 579 |
| 16.260.1.1 scalar_t  | 579 |
| 16.260.1.2 vect_t  | 579 |
| 16.260.2 Member Function Documentation                       | 580 |
| 16.260.2.1 set1() [1/2]                                      | 580 |
| 16.260.2.2 set() [1/2]                                       | 580 |

|                            |     |
|----------------------------|-----|
| 16.260.2.3 gather() [1/2]  | 580 |
| 16.260.2.4 load() [1/2]    | 580 |
| 16.260.2.5 loadu() [1/2]   | 580 |
| 16.260.2.6 store() [1/2]   | 580 |
| 16.260.2.7 storeu() [1/2]  | 580 |
| 16.260.2.8 stream() [1/2]  | 580 |
| 16.260.2.9 sra()           | 581 |
| 16.260.2.10 greater()      | 581 |
| 16.260.2.11 lesser()       | 581 |
| 16.260.2.12 greater_eq()   | 581 |
| 16.260.2.13 lesser_eq()    | 581 |
| 16.260.2.14 mulhi()        | 581 |
| 16.260.2.15 mulx()         | 581 |
| 16.260.2.16 fmaddx()       | 581 |
| 16.260.2.17 fmaddxin()     | 581 |
| 16.260.2.18 fnmaddx()      | 582 |
| 16.260.2.19 fnmaddxin()    | 582 |
| 16.260.2.20 fmsubx()       | 582 |
| 16.260.2.21 fmsubxin()     | 582 |
| 16.260.2.22 hadd_to_scal() | 582 |
| 16.260.2.23 valid()        | 582 |
| 16.260.2.24 compliant()    | 582 |
| 16.260.2.25 set1() [2/2]   | 582 |
| 16.260.2.26 set() [2/2]    | 583 |
| 16.260.2.27 gather() [2/2] | 583 |
| 16.260.2.28 load() [2/2]   | 583 |
| 16.260.2.29 loadu() [2/2]  | 583 |
| 16.260.2.30 store() [2/2]  | 583 |
| 16.260.2.31 storeu() [2/2] | 583 |
| 16.260.2.32 stream() [2/2] | 583 |
| 16.260.2.33 sll()          | 583 |
| 16.260.2.34 srl()          | 583 |
| 16.260.2.35 shuffle()      | 584 |
| 16.260.2.36 unpacklo()     | 584 |
| 16.260.2.37 unpackhi()     | 584 |
| 16.260.2.38 blend()        | 584 |
| 16.260.2.39 add()          | 584 |
| 16.260.2.40 addin()        | 584 |
| 16.260.2.41 sub()          | 584 |
| 16.260.2.42 subin()        | 584 |
| 16.260.2.43 mullo()        | 584 |
| 16.260.2.44 mul()          | 585 |

|  |     |
|--|-----|
| 16.260.2.45 fmadd()  | 585 |
| 16.260.2.46 fmaddin()  | 585 |
| 16.260.2.47 fnmadd()   | 585 |
| 16.260.2.48 fnmaddin()                                       | 585 |
| 16.260.2.49 fmsub()  | 585 |
| 16.260.2.50 fmsubin()  | 585 |
| 16.260.2.51 eq()   | 586 |
| 16.260.2.52 round()  | 586 |
| 16.260.2.53 mod()  | 586 |
| 16.260.2.54 type_string()                                    | 586 |
| 16.260.2.55 zero()   | 586 |
| 16.260.2.56 sll128()   | 586 |
| 16.260.2.57 srl128()   | 586 |
| 16.260.2.58 vand()   | 586 |
| 16.260.2.59 vor()  | 586 |
| 16.260.2.60 vxor()   | 587 |
| 16.260.2.61 vandnot()  | 587 |
| 16.260.3 Field Documentation                                 | 587 |
| 16.260.3.1 vect_size   | 587 |
| 16.260.3.2 alignment   | 587 |
| 16.261 Simd128_impl< true, true, false, 8 > Struct Reference | 587 |
| 16.261.1 Member Typedef Documentation                        | 589 |
| 16.261.1.1 scalar_t  | 589 |
| 16.261.1.2 vect_t  | 589 |
| 16.261.2 Member Function Documentation                       | 589 |
| 16.261.2.1 set1() [1/2]                                      | 589 |
| 16.261.2.2 set() [1/2]                                       | 589 |
| 16.261.2.3 gather() [1/2]                                    | 590 |
| 16.261.2.4 load() [1/2]                                      | 590 |
| 16.261.2.5 loadu() [1/2]                                     | 590 |
| 16.261.2.6 store() [1/2]                                     | 590 |
| 16.261.2.7 storeu() [1/2]                                    | 590 |
| 16.261.2.8 stream() [1/2]                                    | 590 |
| 16.261.2.9 sra()   | 590 |
| 16.261.2.10 greater()  | 590 |
| 16.261.2.11 lesser()   | 590 |
| 16.261.2.12 greater_eq()                                     | 591 |
| 16.261.2.13 lesser_eq()                                      | 591 |
| 16.261.2.14 mullo()  | 591 |
| 16.261.2.15 mulx()   | 591 |
| 16.261.2.16 fmaddx()   | 591 |
| 16.261.2.17 fmaddxin()                                       | 591 |

|                              |     |
|------------------------------|-----|
| 16.261.2.18 fmaddx()         | 591 |
| 16.261.2.19 fmaddxin()       | 591 |
| 16.261.2.20 fmsubx()         | 592 |
| 16.261.2.21 fmsubxin() [1/2] | 592 |
| 16.261.2.22 hadd_to_scal()   | 592 |
| 16.261.2.23 valid()          | 592 |
| 16.261.2.24 compliant()      | 592 |
| 16.261.2.25 set1() [2/2]     | 592 |
| 16.261.2.26 set() [2/2]      | 592 |
| 16.261.2.27 gather() [2/2]   | 592 |
| 16.261.2.28 get()            | 592 |
| 16.261.2.29 load() [2/2]     | 593 |
| 16.261.2.30 loadu() [2/2]    | 593 |
| 16.261.2.31 store() [2/2]    | 593 |
| 16.261.2.32 storeu() [2/2]   | 593 |
| 16.261.2.33 stream() [2/2]   | 593 |
| 16.261.2.34 sll()            | 593 |
| 16.261.2.35 srl()            | 593 |
| 16.261.2.36 shuffle()        | 593 |
| 16.261.2.37 unpacklo()       | 593 |
| 16.261.2.38 unpackhi()       | 594 |
| 16.261.2.39 blend()          | 594 |
| 16.261.2.40 add()            | 594 |
| 16.261.2.41 addin()          | 594 |
| 16.261.2.42 sub()            | 594 |
| 16.261.2.43 subin()          | 594 |
| 16.261.2.44 mul()            | 594 |
| 16.261.2.45 fmadd()          | 594 |
| 16.261.2.46 fmaddin()        | 595 |
| 16.261.2.47 fnmadd()         | 595 |
| 16.261.2.48 fnmaddin()       | 595 |
| 16.261.2.49 fmsub()          | 595 |
| 16.261.2.50 fmsubin()        | 595 |
| 16.261.2.51 fmsubxin() [2/2] | 595 |
| 16.261.2.52 eq()             | 595 |
| 16.261.2.53 round()          | 595 |
| 16.261.2.54 mask_high()      | 596 |
| 16.261.2.55 mulhi_fast()     | 596 |
| 16.261.2.56 mod()            | 596 |
| 16.261.2.57 signbits()       | 596 |
| 16.261.2.58 type_string()    | 596 |
| 16.261.2.59 zero()           | 596 |

|   |     |
|---|-----|
| 16.261.2.60 sll128()  | 596 |
| 16.261.2.61 srl128()  | 596 |
| 16.261.2.62 vand()  | 596 |
| 16.261.2.63 vor()   | 597 |
| 16.261.2.64 vxor()  | 597 |
| 16.261.2.65 vandnot()                                       | 597 |
| 16.261.3 Field Documentation                                | 597 |
| 16.261.3.1 vect_size  | 597 |
| 16.261.3.2 alignment  | 597 |
| 16.262 Simd128_impl< true, true, true, 2 > Struct Reference | 597 |
| 16.262.1 Member Typedef Documentation                       | 599 |
| 16.262.1.1 vect_t   | 599 |
| 16.262.1.2 scalar_t   | 599 |
| 16.262.2 Member Function Documentation                      | 599 |
| 16.262.2.1 valid()  | 599 |
| 16.262.2.2 compliant()                                      | 599 |
| 16.262.2.3 set1()   | 599 |
| 16.262.2.4 set()  | 600 |
| 16.262.2.5 gather()   | 600 |
| 16.262.2.6 load()   | 600 |
| 16.262.2.7 loadu()  | 600 |
| 16.262.2.8 store()  | 600 |
| 16.262.2.9 storeu()   | 600 |
| 16.262.2.10 stream()  | 600 |
| 16.262.2.11 sll()   | 600 |
| 16.262.2.12 srl()   | 601 |
| 16.262.2.13 sra()   | 601 |
| 16.262.2.14 shuffle()                                       | 601 |
| 16.262.2.15 unpacklo()                                      | 601 |
| 16.262.2.16 unpackhi()                                      | 601 |
| 16.262.2.17 blend()   | 601 |
| 16.262.2.18 add()   | 601 |
| 16.262.2.19 addin()   | 601 |
| 16.262.2.20 sub()   | 601 |
| 16.262.2.21 subin()   | 602 |
| 16.262.2.22 mullo()   | 602 |
| 16.262.2.23 mul()   | 602 |
| 16.262.2.24 mulhi()   | 602 |
| 16.262.2.25 mulx()  | 602 |
| 16.262.2.26 fmadd()   | 602 |
| 16.262.2.27 fmaddin()                                       | 602 |
| 16.262.2.28 fmaddx()  | 602 |



|   |     |
|---|-----|
| 16.262.2.29 fmaddxin()                                      | 603 |
| 16.262.2.30 fnmadd()  | 603 |
| 16.262.2.31 fnmaddin()                                      | 603 |
| 16.262.2.32 fnmaddx()                                       | 603 |
| 16.262.2.33 fnmaddxin()                                     | 603 |
| 16.262.2.34 fmsub()   | 603 |
| 16.262.2.35 fmsubin()                                       | 603 |
| 16.262.2.36 fmsubx()  | 603 |
| 16.262.2.37 fmsubxin()                                      | 604 |
| 16.262.2.38 eq()  | 604 |
| 16.262.2.39 greater()                                       | 604 |
| 16.262.2.40 lesser()  | 604 |
| 16.262.2.41 greater_eq()                                    | 604 |
| 16.262.2.42 lesser_eq()                                     | 604 |
| 16.262.2.43 hadd_to_scal()                                  | 604 |
| 16.262.2.44 round()   | 604 |
| 16.262.2.45 mod()   | 605 |
| 16.262.2.46 type_string()                                   | 605 |
| 16.262.2.47 zero()  | 605 |
| 16.262.2.48 sll128()  | 605 |
| 16.262.2.49 srl128()  | 605 |
| 16.262.2.50 vand()  | 605 |
| 16.262.2.51 vor()   | 605 |
| 16.262.2.52 vxor()  | 605 |
| 16.262.2.53 vandnot()                                       | 606 |
| 16.262.3 Field Documentation                                | 606 |
| 16.262.3.1 vect_size  | 606 |
| 16.262.3.2 alignment  | 606 |
| 16.263 Simd128_impl< true, true, true, 4 > Struct Reference | 606 |
| 16.263.1 Member Typedef Documentation                       | 608 |
| 16.263.1.1 vect_t   | 608 |
| 16.263.1.2 scalar_t   | 608 |
| 16.263.2 Member Function Documentation                      | 608 |
| 16.263.2.1 valid()  | 608 |
| 16.263.2.2 compliant()                                      | 608 |
| 16.263.2.3 set1()   | 608 |
| 16.263.2.4 set()  | 608 |
| 16.263.2.5 gather()   | 608 |
| 16.263.2.6 load()   | 609 |
| 16.263.2.7 loadu()  | 609 |
| 16.263.2.8 store()  | 609 |
| 16.263.2.9 storeu()   | 609 |

|                            |     |
|----------------------------|-----|
| 16.263.2.10 stream()       | 609 |
| 16.263.2.11 sll()          | 609 |
| 16.263.2.12 srl()          | 609 |
| 16.263.2.13 sra()          | 609 |
| 16.263.2.14 shuffle()      | 609 |
| 16.263.2.15 unpacklo()     | 610 |
| 16.263.2.16 unpackhi()     | 610 |
| 16.263.2.17 blend()        | 610 |
| 16.263.2.18 add()          | 610 |
| 16.263.2.19 addin()        | 610 |
| 16.263.2.20 sub()          | 610 |
| 16.263.2.21 subin()        | 610 |
| 16.263.2.22 mullo()        | 610 |
| 16.263.2.23 mul()          | 611 |
| 16.263.2.24 mulhi()        | 611 |
| 16.263.2.25 mulx()         | 611 |
| 16.263.2.26 fmadd()        | 611 |
| 16.263.2.27 fmaddin()      | 611 |
| 16.263.2.28 fmaddx()       | 611 |
| 16.263.2.29 fmaddxin()     | 611 |
| 16.263.2.30 fnmadd()       | 611 |
| 16.263.2.31 fnmaddin()     | 612 |
| 16.263.2.32 fnmaddx()      | 612 |
| 16.263.2.33 fnmaddxin()    | 612 |
| 16.263.2.34 fmsub()        | 612 |
| 16.263.2.35 fmsubin()      | 612 |
| 16.263.2.36 fmsubx()       | 612 |
| 16.263.2.37 fmsubxin()     | 612 |
| 16.263.2.38 eq()           | 612 |
| 16.263.2.39 greater()      | 613 |
| 16.263.2.40 lesser()       | 613 |
| 16.263.2.41 greater_eq()   | 613 |
| 16.263.2.42 lesser_eq()    | 613 |
| 16.263.2.43 hadd_to_scal() | 613 |
| 16.263.2.44 round()        | 613 |
| 16.263.2.45 mod()          | 613 |
| 16.263.2.46 type_string()  | 613 |
| 16.263.2.47 zero()         | 614 |
| 16.263.2.48 sll128()       | 614 |
| 16.263.2.49 srl128()       | 614 |
| 16.263.2.50 vand()         | 614 |
| 16.263.2.51 vor()          | 614 |

|   |     |
|---|-----|
| 16.263.2.52 vxor()  | 614 |
| 16.263.2.53 vandnot()                                       | 614 |
| 16.263.3 Field Documentation                                | 614 |
| 16.263.3.1 vect_size  | 614 |
| 16.263.3.2 alignment  | 614 |
| 16.264 Simd128_impl< true, true, true, 8 > Struct Reference | 615 |
| 16.264.1 Member Typedef Documentation                       | 616 |
| 16.264.1.1 vect_t   | 616 |
| 16.264.1.2 scalar_t   | 616 |
| 16.264.2 Member Function Documentation                      | 617 |
| 16.264.2.1 valid()  | 617 |
| 16.264.2.2 compliant()                                      | 617 |
| 16.264.2.3 set1()   | 617 |
| 16.264.2.4 set()  | 617 |
| 16.264.2.5 gather()   | 617 |
| 16.264.2.6 get()  | 617 |
| 16.264.2.7 load()   | 617 |
| 16.264.2.8 loadu()  | 617 |
| 16.264.2.9 store()  | 617 |
| 16.264.2.10 storeu()  | 618 |
| 16.264.2.11 stream()  | 618 |
| 16.264.2.12 sll()   | 618 |
| 16.264.2.13 srl()   | 618 |
| 16.264.2.14 sra()   | 618 |
| 16.264.2.15 shuffle()                                       | 618 |
| 16.264.2.16 unpacklo()                                      | 618 |
| 16.264.2.17 unpackhi()                                      | 618 |
| 16.264.2.18 blend()   | 618 |
| 16.264.2.19 add()   | 619 |
| 16.264.2.20 addin()   | 619 |
| 16.264.2.21 sub()   | 619 |
| 16.264.2.22 subin()   | 619 |
| 16.264.2.23 mullo()   | 619 |
| 16.264.2.24 mul()   | 619 |
| 16.264.2.25 mulx()  | 619 |
| 16.264.2.26 fmadd()   | 619 |
| 16.264.2.27 fmaddin()                                       | 620 |
| 16.264.2.28 fmaddx()  | 620 |
| 16.264.2.29 fmaddxin()                                      | 620 |
| 16.264.2.30 fnmadd()  | 620 |
| 16.264.2.31 fnmaddin()                                      | 620 |
| 16.264.2.32 fnmaddx()                                       | 620 |

|  |     |
|--|-----|
| 16.264.2.33 fmaddxin()                 | 620 |
| 16.264.2.34 fmsub()                    | 620 |
| 16.264.2.35 fmsubin()                  | 621 |
| 16.264.2.36 fmsubx()                   | 621 |
| 16.264.2.37 fmsubxin()                 | 621 |
| 16.264.2.38 eq()                       | 621 |
| 16.264.2.39 greater()                  | 621 |
| 16.264.2.40 lesser()                   | 621 |
| 16.264.2.41 greater_eq()               | 621 |
| 16.264.2.42 lesser_eq()                | 622 |
| 16.264.2.43 hadd_to_scal()             | 622 |
| 16.264.2.44 round()                    | 622 |
| 16.264.2.45 mask_high()                | 622 |
| 16.264.2.46 mulhi_fast()               | 622 |
| 16.264.2.47 mod()                      | 622 |
| 16.264.2.48 signbits()                 | 622 |
| 16.264.2.49 type_string()              | 622 |
| 16.264.2.50 zero()                     | 622 |
| 16.264.2.51 sll128()                   | 623 |
| 16.264.2.52 srl128()                   | 623 |
| 16.264.2.53 vand()                     | 623 |
| 16.264.2.54 vor()                      | 623 |
| 16.264.2.55 vxor()                     | 623 |
| 16.264.2.56 vandnot()                  | 623 |
| 16.264.3 Field Documentation           | 623 |
| 16.264.3.1 vect_size                   | 623 |
| 16.264.3.2 alignment                   | 623 |
| 16.265 Simd128fp_base Struct Reference | 623 |
| 16.265.1 Member Function Documentation | 624 |
| 16.265.1.1 type_string()               | 624 |
| 16.266 Simd128i_base Struct Reference  | 624 |
| 16.266.1 Member Typedef Documentation  | 624 |
| 16.266.1.1 vect_t                      | 624 |
| 16.266.2 Member Function Documentation | 625 |
| 16.266.2.1 type_string()               | 625 |
| 16.266.2.2 zero()                      | 625 |
| 16.266.2.3 sll128()                    | 625 |
| 16.266.2.4 srl128()                    | 625 |
| 16.266.2.5 vand()                      | 625 |
| 16.266.2.6 vor()                       | 625 |
| 16.266.2.7 vxor()                      | 625 |
| 16.266.2.8 vandnot()                   | 625 |

|   |     |
|---|-----|
| 16.267 Simd256_impl< ArithType, Int, Signed, Size > Struct Template Reference . . . . . | 626 |
| 16.268 Simd256_impl< true, false, true, 4 > Struct Reference . . . . .                  | 626 |
| 16.269 Simd256_impl< true, false, true, 8 > Struct Reference . . . . .                  | 626 |
| 16.269.1 Member Typedef Documentation . . . . .   | 627 |
| 16.269.1.1 vect_t . . . . .   | 627 |
| 16.269.1.2 scalar_t . . . . .   | 627 |
| 16.269.2 Member Function Documentation . . . . .  | 627 |
| 16.269.2.1 valid() . . . . .  | 628 |
| 16.269.2.2 compliant() . . . . .  | 628 |
| 16.269.2.3 zero() . . . . .   | 628 |
| 16.269.2.4 set1() . . . . .   | 628 |
| 16.269.2.5 set() . . . . .  | 628 |
| 16.269.2.6 gather() . . . . .   | 628 |
| 16.269.2.7 load() . . . . .   | 628 |
| 16.269.2.8 loadu() . . . . .  | 628 |
| 16.269.2.9 store() . . . . .  | 628 |
| 16.269.2.10 storeu() . . . . .  | 629 |
| 16.269.2.11 stream() . . . . .  | 629 |
| 16.269.2.12 unpacklo_twice() . . . . .  | 629 |
| 16.269.2.13 unpackhi_twice() . . . . .  | 629 |
| 16.269.2.14 blend() . . . . .   | 629 |
| 16.269.2.15 blendv() . . . . .  | 629 |
| 16.269.2.16 add() . . . . .   | 629 |
| 16.269.2.17 addin() . . . . .   | 629 |
| 16.269.2.18 sub() . . . . .   | 630 |
| 16.269.2.19 subin() . . . . .   | 630 |
| 16.269.2.20 mul() . . . . .   | 630 |
| 16.269.2.21 mulin() . . . . .   | 630 |
| 16.269.2.22 div() . . . . .   | 630 |
| 16.269.2.23 fmadd() . . . . .   | 630 |
| 16.269.2.24 fmaddin() . . . . .   | 630 |
| 16.269.2.25 fnmadd() . . . . .  | 630 |
| 16.269.2.26 fnmaddin() . . . . .  | 631 |
| 16.269.2.27 fmsub() . . . . .   | 631 |
| 16.269.2.28 fmsubin() . . . . .   | 631 |
| 16.269.2.29 eq() . . . . .  | 631 |
| 16.269.2.30 lesser() . . . . .  | 631 |
| 16.269.2.31 lesser_eq() . . . . .   | 631 |
| 16.269.2.32 greater() . . . . .   | 631 |
| 16.269.2.33 greater_eq() . . . . .  | 631 |
| 16.269.2.34 vand() . . . . .  | 632 |
| 16.269.2.35 vor() . . . . .   | 632 |

|  |     |
|--|-----|
| 16.269.2.36 vxor()   | 632 |
| 16.269.2.37 vandnot()  | 632 |
| 16.269.2.38 floor()  | 632 |
| 16.269.2.39 ceil()   | 632 |
| 16.269.2.40 round()  | 632 |
| 16.269.2.41 hadd()   | 632 |
| 16.269.2.42 hadd_to_scal()                                   | 632 |
| 16.269.2.43 mod()  | 633 |
| 16.269.3 Field Documentation                                 | 633 |
| 16.269.3.1 vect_size   | 633 |
| 16.269.3.2 alignment   | 633 |
| 16.270 Simd256_impl< true, true, false, 2 > Struct Reference | 633 |
| 16.270.1 Member Typedef Documentation                        | 635 |
| 16.270.1.1 scalar_t  | 635 |
| 16.270.1.2 simdHalf  | 635 |
| 16.270.1.3 vect_t  | 635 |
| 16.270.1.4 half_t  | 635 |
| 16.270.2 Member Function Documentation                       | 635 |
| 16.270.2.1 set1() [1/2]                                      | 635 |
| 16.270.2.2 set() [1/2]                                       | 636 |
| 16.270.2.3 gather() [1/2]                                    | 636 |
| 16.270.2.4 load() [1/2]                                      | 636 |
| 16.270.2.5 loadu() [1/2]                                     | 636 |
| 16.270.2.6 store() [1/2]                                     | 636 |
| 16.270.2.7 storeu() [1/2]                                    | 636 |
| 16.270.2.8 stream() [1/2]                                    | 636 |
| 16.270.2.9 sra()   | 637 |
| 16.270.2.10 greater()  | 637 |
| 16.270.2.11 lesser()   | 637 |
| 16.270.2.12 greater_eq()                                     | 637 |
| 16.270.2.13 lesser_eq()                                      | 637 |
| 16.270.2.14 mulhi()  | 637 |
| 16.270.2.15 mulx()   | 637 |
| 16.270.2.16 fmaddx()   | 637 |
| 16.270.2.17 fmaddxin()                                       | 637 |
| 16.270.2.18 fnmaddx()  | 638 |
| 16.270.2.19 fnmaddxin()                                      | 638 |
| 16.270.2.20 fmsubx()   | 638 |
| 16.270.2.21 fmsubxin()                                       | 638 |
| 16.270.2.22 hadd_to_scal()                                   | 638 |
| 16.270.2.23 valid()  | 638 |
| 16.270.2.24 compliant()                                      | 638 |

|  |     |
|--|-----|
| 16.270.2.25 set1() [2/2]                                     | 638 |
| 16.270.2.26 set() [2/2]                                      | 639 |
| 16.270.2.27 gather() [2/2]                                   | 639 |
| 16.270.2.28 load() [2/2]                                     | 639 |
| 16.270.2.29 loadu() [2/2]                                    | 639 |
| 16.270.2.30 store() [2/2]                                    | 639 |
| 16.270.2.31 storeu() [2/2]                                   | 639 |
| 16.270.2.32 stream() [2/2]                                   | 639 |
| 16.270.2.33 sll()  | 640 |
| 16.270.2.34 srl()  | 640 |
| 16.270.2.35 shuffle()  | 640 |
| 16.270.2.36 unpacklo_twice()                                 | 640 |
| 16.270.2.37 unpackhi_twice()                                 | 640 |
| 16.270.2.38 unpacklo()                                       | 640 |
| 16.270.2.39 unpackhi()                                       | 640 |
| 16.270.2.40 unpacklohi()                                     | 640 |
| 16.270.2.41 blend_twice()                                    | 640 |
| 16.270.2.42 add()  | 641 |
| 16.270.2.43 addin()  | 641 |
| 16.270.2.44 sub()  | 641 |
| 16.270.2.45 subin()  | 641 |
| 16.270.2.46 mullo()  | 641 |
| 16.270.2.47 mul()  | 641 |
| 16.270.2.48 fmadd()  | 641 |
| 16.270.2.49 fmaddin()  | 641 |
| 16.270.2.50 fnmadd()   | 642 |
| 16.270.2.51 fnmaddin()                                       | 642 |
| 16.270.2.52 fmsub()  | 642 |
| 16.270.2.53 fmsubin()  | 642 |
| 16.270.2.54 eq()   | 642 |
| 16.270.2.55 round()  | 642 |
| 16.270.2.56 mod()  | 642 |
| 16.270.2.57 type_string()                                    | 643 |
| 16.270.2.58 zero()   | 643 |
| 16.270.3 Field Documentation                                 | 643 |
| 16.270.3.1 vect_size   | 643 |
| 16.270.3.2 alignment   | 643 |
| 16.271 Simd256_impl< true, true, false, 4 > Struct Reference | 643 |
| 16.271.1 Member Typedef Documentation                        | 646 |
| 16.271.1.1 scalar_t [1/2]                                    | 646 |
| 16.271.1.2 simdHalf [1/2]                                    | 646 |
| 16.271.1.3 scalar_t [2/2]                                    | 646 |

|  |     |
|--|-----|
| 16.271.1.4 simdHalf [2/2]              | 646 |
| 16.271.1.5 vect_t [1/2]                | 646 |
| 16.271.1.6 vect_t [2/2]                | 646 |
| 16.271.1.7 half_t [1/2]                | 647 |
| 16.271.1.8 half_t [2/2]                | 647 |
| 16.271.2 Member Function Documentation | 647 |
| 16.271.2.1 set1() [1/3]                | 647 |
| 16.271.2.2 set() [1/4]                 | 647 |
| 16.271.2.3 gather() [1/3]              | 647 |
| 16.271.2.4 load() [1/3]                | 647 |
| 16.271.2.5 loadu() [1/3]               | 647 |
| 16.271.2.6 store() [1/3]               | 647 |
| 16.271.2.7 storeu() [1/3]              | 648 |
| 16.271.2.8 stream() [1/3]              | 648 |
| 16.271.2.9 sra() [1/2]                 | 648 |
| 16.271.2.10 greater() [1/2]            | 648 |
| 16.271.2.11 lesser() [1/2]             | 648 |
| 16.271.2.12 greater_eq() [1/2]         | 648 |
| 16.271.2.13 lesser_eq() [1/2]          | 648 |
| 16.271.2.14 mulhi() [1/2]              | 648 |
| 16.271.2.15 mulx() [1/2]               | 648 |
| 16.271.2.16 fmaddx() [1/2]             | 649 |
| 16.271.2.17 fmaddxin() [1/2]           | 649 |
| 16.271.2.18 fnmaddx() [1/2]            | 649 |
| 16.271.2.19 fnmaddxin() [1/2]          | 649 |
| 16.271.2.20 fmsubx() [1/2]             | 649 |
| 16.271.2.21 fmsubxin() [1/2]           | 649 |
| 16.271.2.22 hadd_to_scal() [1/2]       | 649 |
| 16.271.2.23 set1() [2/3]               | 650 |
| 16.271.2.24 set() [2/4]                | 650 |
| 16.271.2.25 gather() [2/3]             | 650 |
| 16.271.2.26 load() [2/3]               | 650 |
| 16.271.2.27 loadu() [2/3]              | 650 |
| 16.271.2.28 store() [2/3]              | 650 |
| 16.271.2.29 storeu() [2/3]             | 650 |
| 16.271.2.30 stream() [2/3]             | 650 |
| 16.271.2.31 sra() [2/2]                | 651 |
| 16.271.2.32 greater() [2/2]            | 651 |
| 16.271.2.33 lesser() [2/2]             | 651 |
| 16.271.2.34 greater_eq() [2/2]         | 651 |
| 16.271.2.35 lesser_eq() [2/2]          | 651 |
| 16.271.2.36 mulhi() [2/2]              | 651 |



|                                   |     |
|-----------------------------------|-----|
| 16.271.2.37 mulx() [2/2]          | 651 |
| 16.271.2.38 fmaddx() [2/2]        | 651 |
| 16.271.2.39 fmaddxin() [2/2]      | 651 |
| 16.271.2.40 fnmaddx() [2/2]       | 652 |
| 16.271.2.41 fnmaddxin() [2/2]     | 652 |
| 16.271.2.42 fmsubx() [2/2]        | 652 |
| 16.271.2.43 fmsubxin() [2/2]      | 652 |
| 16.271.2.44 hadd_to_scal() [2/2]  | 652 |
| 16.271.2.45 valid() [1/2]         | 652 |
| 16.271.2.46 valid() [2/2]         | 652 |
| 16.271.2.47 compliant() [1/2]     | 652 |
| 16.271.2.48 compliant() [2/2]     | 653 |
| 16.271.2.49 set1() [3/3]          | 653 |
| 16.271.2.50 set() [3/4]           | 653 |
| 16.271.2.51 set() [4/4]           | 653 |
| 16.271.2.52 gather() [3/3]        | 653 |
| 16.271.2.53 load() [3/3]          | 653 |
| 16.271.2.54 loadu() [3/3]         | 654 |
| 16.271.2.55 store() [3/3]         | 654 |
| 16.271.2.56 storeu() [3/3]        | 654 |
| 16.271.2.57 stream() [3/3]        | 654 |
| 16.271.2.58 sll() [1/2]           | 654 |
| 16.271.2.59 sll() [2/2]           | 654 |
| 16.271.2.60 srl() [1/2]           | 654 |
| 16.271.2.61 srl() [2/2]           | 654 |
| 16.271.2.62 shuffle_twice() [1/2] | 654 |
| 16.271.2.63 shuffle_twice() [2/2] | 655 |
| 16.271.2.64 shuffle() [1/2]       | 655 |
| 16.271.2.65 shuffle() [2/2]       | 655 |
| 16.271.2.66 unpacklo_twice()      | 655 |
| 16.271.2.67 unpackhi_twice()      | 655 |
| 16.271.2.68 unpacklo()            | 655 |
| 16.271.2.69 unpackhi()            | 655 |
| 16.271.2.70 unpacklohi()          | 655 |
| 16.271.2.71 blend()               | 655 |
| 16.271.2.72 add() [1/2]           | 656 |
| 16.271.2.73 add() [2/2]           | 656 |
| 16.271.2.74 addin() [1/2]         | 656 |
| 16.271.2.75 addin() [2/2]         | 656 |
| 16.271.2.76 sub() [1/2]           | 656 |
| 16.271.2.77 sub() [2/2]           | 656 |
| 16.271.2.78 subin() [1/2]         | 656 |

|  |     |
|--|-----|
| 16.271.2.79 subin() [2/2]                                    | 656 |
| 16.271.2.80 mullo() [1/2]                                    | 657 |
| 16.271.2.81 mullo() [2/2]                                    | 657 |
| 16.271.2.82 mul() [1/2]                                      | 657 |
| 16.271.2.83 mul() [2/2]                                      | 657 |
| 16.271.2.84 fmadd() [1/2]                                    | 657 |
| 16.271.2.85 fmadd() [2/2]                                    | 657 |
| 16.271.2.86 fmaddin() [1/2]                                  | 657 |
| 16.271.2.87 fmaddin() [2/2]                                  | 657 |
| 16.271.2.88 fnmadd() [1/2]                                   | 658 |
| 16.271.2.89 fnmadd() [2/2]                                   | 658 |
| 16.271.2.90 fnmaddin() [1/2]                                 | 658 |
| 16.271.2.91 fnmaddin() [2/2]                                 | 658 |
| 16.271.2.92 fmsub() [1/2]                                    | 658 |
| 16.271.2.93 fmsub() [2/2]                                    | 658 |
| 16.271.2.94 fmsubin() [1/2]                                  | 658 |
| 16.271.2.95 fmsubin() [2/2]                                  | 658 |
| 16.271.2.96 eq() [1/2]                                       | 659 |
| 16.271.2.97 eq() [2/2]                                       | 659 |
| 16.271.2.98 round() [1/2]                                    | 659 |
| 16.271.2.99 round() [2/2]                                    | 659 |
| 16.271.2.100 mod() [1/2]                                     | 659 |
| 16.271.2.101 mod() [2/2]                                     | 659 |
| 16.271.2.102 type_string() [1/2]                             | 659 |
| 16.271.2.103 type_string() [2/2]                             | 660 |
| 16.271.2.104 zero() [1/2]                                    | 660 |
| 16.271.2.105 zero() [2/2]                                    | 660 |
| 16.271.2.106 vor()   | 660 |
| 16.271.2.107 vxor()  | 660 |
| 16.271.2.108 vand()  | 660 |
| 16.271.2.109 vandnot()                                       | 660 |
| 16.271.3 Field Documentation                                 | 660 |
| 16.271.3.1 vect_size   | 660 |
| 16.271.3.2 alignment   | 660 |
| 16.272 Simd256_impl< true, true, false, 8 > Struct Reference | 661 |
| 16.272.1 Member Typedef Documentation                        | 662 |
| 16.272.1.1 scalar_t  | 663 |
| 16.272.1.2 simdHalf  | 663 |
| 16.272.1.3 vect_t  | 663 |
| 16.272.1.4 half_t  | 663 |
| 16.272.2 Member Function Documentation                       | 663 |
| 16.272.2.1 set1() [1/2]                                      | 663 |

|                              |     |
|------------------------------|-----|
| 16.272.2.2 set() [1/2]       | 663 |
| 16.272.2.3 gather() [1/2]    | 663 |
| 16.272.2.4 load() [1/2]      | 663 |
| 16.272.2.5 loadu() [1/2]     | 663 |
| 16.272.2.6 store() [1/2]     | 664 |
| 16.272.2.7 storeu() [1/2]    | 664 |
| 16.272.2.8 stream() [1/2]    | 664 |
| 16.272.2.9 sra()             | 664 |
| 16.272.2.10 greater()        | 664 |
| 16.272.2.11 lesser()         | 664 |
| 16.272.2.12 greater_eq()     | 664 |
| 16.272.2.13 lesser_eq()      | 664 |
| 16.272.2.14 mullo()          | 664 |
| 16.272.2.15 mulx()           | 665 |
| 16.272.2.16 fmaddx()         | 665 |
| 16.272.2.17 fmaddxin()       | 665 |
| 16.272.2.18 fnmaddx()        | 665 |
| 16.272.2.19 fnmaddxin()      | 665 |
| 16.272.2.20 fmsubx()         | 665 |
| 16.272.2.21 fmsubxin()       | 665 |
| 16.272.2.22 hadd_to_scal()   | 666 |
| 16.272.2.23 valid()          | 666 |
| 16.272.2.24 compliant()      | 666 |
| 16.272.2.25 set1() [2/2]     | 666 |
| 16.272.2.26 set() [2/2]      | 666 |
| 16.272.2.27 gather() [2/2]   | 666 |
| 16.272.2.28 get()            | 666 |
| 16.272.2.29 load() [2/2]     | 666 |
| 16.272.2.30 loadu() [2/2]    | 666 |
| 16.272.2.31 store() [2/2]    | 667 |
| 16.272.2.32 storeu() [2/2]   | 667 |
| 16.272.2.33 stream() [2/2]   | 667 |
| 16.272.2.34 sll()            | 667 |
| 16.272.2.35 srl()            | 667 |
| 16.272.2.36 shuffle()        | 667 |
| 16.272.2.37 unpacklo_twice() | 667 |
| 16.272.2.38 unpackhi_twice() | 667 |
| 16.272.2.39 unpacklo()       | 667 |
| 16.272.2.40 unpackhi()       | 668 |
| 16.272.2.41 unpacklohi()     | 668 |
| 16.272.2.42 blend()          | 668 |
| 16.272.2.43 add()            | 668 |

|             |  |     |
|-------------|--|-----|
| 16.272.2.44 | addin()  | 668 |
| 16.272.2.45 | sub()  | 668 |
| 16.272.2.46 | subin()  | 668 |
| 16.272.2.47 | mul()  | 668 |
| 16.272.2.48 | fmadd()  | 669 |
| 16.272.2.49 | fmaddin()  | 669 |
| 16.272.2.50 | fnmadd()   | 669 |
| 16.272.2.51 | fnmaddin()   | 669 |
| 16.272.2.52 | fmsub()  | 669 |
| 16.272.2.53 | fmsubin()  | 669 |
| 16.272.2.54 | eq()   | 669 |
| 16.272.2.55 | round()  | 669 |
| 16.272.2.56 | mask_high()  | 670 |
| 16.272.2.57 | mulhi_fast()   | 670 |
| 16.272.2.58 | mod()  | 670 |
| 16.272.2.59 | signbits()   | 670 |
| 16.272.2.60 | type_string()  | 670 |
| 16.272.2.61 | zero()   | 670 |
| 16.272.3    | Field Documentation                                  | 670 |
| 16.272.3.1  | vect_size  | 670 |
| 16.272.3.2  | alignment  | 670 |
| 16.273      | Simd256_impl< true, true, true, 2 > Struct Reference | 671 |
| 16.273.1    | Member Typedef Documentation                         | 672 |
| 16.273.1.1  | vect_t   | 672 |
| 16.273.1.2  | half_t   | 672 |
| 16.273.1.3  | scalar_t   | 672 |
| 16.273.1.4  | simdHalf   | 673 |
| 16.273.2    | Member Function Documentation                        | 673 |
| 16.273.2.1  | valid()  | 673 |
| 16.273.2.2  | compliant()  | 673 |
| 16.273.2.3  | set1()   | 673 |
| 16.273.2.4  | set()  | 673 |
| 16.273.2.5  | gather()   | 673 |
| 16.273.2.6  | load()   | 673 |
| 16.273.2.7  | loadu()  | 674 |
| 16.273.2.8  | store()  | 674 |
| 16.273.2.9  | storeu()   | 674 |
| 16.273.2.10 | stream()   | 674 |
| 16.273.2.11 | sll()  | 674 |
| 16.273.2.12 | srl()  | 674 |
| 16.273.2.13 | sra()  | 674 |
| 16.273.2.14 | shuffle()  | 674 |

|   |     |
|---|-----|
| 16.273.2.15 unpacklo_twice()                                | 674 |
| 16.273.2.16 unpackhi_twice()                                | 675 |
| 16.273.2.17 unpacklo()                                      | 675 |
| 16.273.2.18 unpackhi()                                      | 675 |
| 16.273.2.19 unpacklohi()                                    | 675 |
| 16.273.2.20 blend_twice()                                   | 675 |
| 16.273.2.21 add()   | 675 |
| 16.273.2.22 addin()   | 675 |
| 16.273.2.23 sub()   | 675 |
| 16.273.2.24 subin()   | 676 |
| 16.273.2.25 mullo()   | 676 |
| 16.273.2.26 mul()   | 676 |
| 16.273.2.27 mulhi()   | 676 |
| 16.273.2.28 mulx()  | 676 |
| 16.273.2.29 fmadd()   | 676 |
| 16.273.2.30 fmaddin()                                       | 676 |
| 16.273.2.31 fmaddx()  | 676 |
| 16.273.2.32 fmaddxin()                                      | 677 |
| 16.273.2.33 fnmadd()  | 677 |
| 16.273.2.34 fnmaddin()                                      | 677 |
| 16.273.2.35 fnmaddx()                                       | 677 |
| 16.273.2.36 fnmaddxin()                                     | 677 |
| 16.273.2.37 fmsub()   | 677 |
| 16.273.2.38 fmsubin()                                       | 677 |
| 16.273.2.39 fmsubx()  | 677 |
| 16.273.2.40 fmsubxin()                                      | 678 |
| 16.273.2.41 eq()  | 678 |
| 16.273.2.42 greater()                                       | 678 |
| 16.273.2.43 lesser()  | 678 |
| 16.273.2.44 greater_eq()                                    | 678 |
| 16.273.2.45 lesser_eq()                                     | 678 |
| 16.273.2.46 hadd_to_scal()                                  | 678 |
| 16.273.2.47 round()   | 678 |
| 16.273.2.48 mod()   | 679 |
| 16.273.2.49 type_string()                                   | 679 |
| 16.273.2.50 zero()  | 679 |
| 16.273.3 Field Documentation                                | 679 |
| 16.273.3.1 vect_size  | 679 |
| 16.273.3.2 alignment  | 679 |
| 16.274 Simd256_impl< true, true, true, 4 > Struct Reference | 679 |
| 16.274.1 Member Typedef Documentation                       | 682 |
| 16.274.1.1 vect_t [1/2]                                     | 682 |

|  |     |
|--|-----|
| 16.274.1.2 half_t [1/2]                | 682 |
| 16.274.1.3 scalar_t [1/2]              | 682 |
| 16.274.1.4 simdHalf [1/2]              | 682 |
| 16.274.1.5 vect_t [2/2]                | 683 |
| 16.274.1.6 half_t [2/2]                | 683 |
| 16.274.1.7 scalar_t [2/2]              | 683 |
| 16.274.1.8 simdHalf [2/2]              | 683 |
| 16.274.2 Member Function Documentation | 683 |
| 16.274.2.1 valid() [1/2]               | 683 |
| 16.274.2.2 compliant() [1/2]           | 683 |
| 16.274.2.3 set1() [1/2]                | 683 |
| 16.274.2.4 set() [1/2]                 | 683 |
| 16.274.2.5 gather() [1/2]              | 683 |
| 16.274.2.6 load() [1/2]                | 684 |
| 16.274.2.7 loadu() [1/2]               | 684 |
| 16.274.2.8 store() [1/2]               | 684 |
| 16.274.2.9 storeu() [1/2]              | 684 |
| 16.274.2.10 stream() [1/2]             | 684 |
| 16.274.2.11 sll() [1/2]                | 684 |
| 16.274.2.12 srl() [1/2]                | 684 |
| 16.274.2.13 sra() [1/2]                | 684 |
| 16.274.2.14 shuffle_twice() [1/2]      | 684 |
| 16.274.2.15 shuffle() [1/2]            | 685 |
| 16.274.2.16 unpacklo_twice()           | 685 |
| 16.274.2.17 unpackhi_twice()           | 685 |
| 16.274.2.18 unpacklo()                 | 685 |
| 16.274.2.19 unpackhi()                 | 685 |
| 16.274.2.20 unpacklohi()               | 685 |
| 16.274.2.21 blend()                    | 685 |
| 16.274.2.22 add() [1/2]                | 685 |
| 16.274.2.23 addin() [1/2]              | 686 |
| 16.274.2.24 sub() [1/2]                | 686 |
| 16.274.2.25 subin() [1/2]              | 686 |
| 16.274.2.26 mullo() [1/2]              | 686 |
| 16.274.2.27 mul() [1/2]                | 686 |
| 16.274.2.28 mulhi() [1/2]              | 686 |
| 16.274.2.29 mulx() [1/2]               | 686 |
| 16.274.2.30 fmadd() [1/2]              | 686 |
| 16.274.2.31 fmaddin() [1/2]            | 687 |
| 16.274.2.32 fmaddx() [1/2]             | 687 |
| 16.274.2.33 fmaddxin() [1/2]           | 687 |
| 16.274.2.34 fnmadd() [1/2]             | 687 |

|                                   |     |
|-----------------------------------|-----|
| 16.274.2.35 fnmaddin() [1/2]      | 687 |
| 16.274.2.36 fnmaddx() [1/2]       | 687 |
| 16.274.2.37 fnmaddxin() [1/2]     | 687 |
| 16.274.2.38 fmsub() [1/2]         | 687 |
| 16.274.2.39 fmsubin() [1/2]       | 688 |
| 16.274.2.40 fmsubx() [1/2]        | 688 |
| 16.274.2.41 fmsubxin() [1/2]      | 688 |
| 16.274.2.42 eq() [1/2]            | 688 |
| 16.274.2.43 greater() [1/2]       | 688 |
| 16.274.2.44 lesser() [1/2]        | 688 |
| 16.274.2.45 greater_eq() [1/2]    | 688 |
| 16.274.2.46 lesser_eq() [1/2]     | 689 |
| 16.274.2.47 hadd_to_scal() [1/2]  | 689 |
| 16.274.2.48 round() [1/2]         | 689 |
| 16.274.2.49 mod() [1/2]           | 689 |
| 16.274.2.50 valid() [2/2]         | 689 |
| 16.274.2.51 compliant() [2/2]     | 689 |
| 16.274.2.52 set1() [2/2]          | 689 |
| 16.274.2.53 set() [2/2]           | 689 |
| 16.274.2.54 gather() [2/2]        | 690 |
| 16.274.2.55 load() [2/2]          | 690 |
| 16.274.2.56 loadu() [2/2]         | 690 |
| 16.274.2.57 store() [2/2]         | 690 |
| 16.274.2.58 storeu() [2/2]        | 690 |
| 16.274.2.59 stream() [2/2]        | 690 |
| 16.274.2.60 sll() [2/2]           | 690 |
| 16.274.2.61 srl() [2/2]           | 691 |
| 16.274.2.62 sra() [2/2]           | 691 |
| 16.274.2.63 shuffle_twice() [2/2] | 691 |
| 16.274.2.64 shuffle() [2/2]       | 691 |
| 16.274.2.65 add() [2/2]           | 691 |
| 16.274.2.66 addin() [2/2]         | 691 |
| 16.274.2.67 sub() [2/2]           | 691 |
| 16.274.2.68 subin() [2/2]         | 691 |
| 16.274.2.69 mullo() [2/2]         | 691 |
| 16.274.2.70 mul() [2/2]           | 692 |
| 16.274.2.71 mulhi() [2/2]         | 692 |
| 16.274.2.72 mulx() [2/2]          | 692 |
| 16.274.2.73 fmadd() [2/2]         | 692 |
| 16.274.2.74 fmaddin() [2/2]       | 692 |
| 16.274.2.75 fmaddx() [2/2]        | 692 |
| 16.274.2.76 fmaddxin() [2/2]      | 692 |

|   |        |     |
|---|--------|-----|
| 16.274.2.77 fmadd()   | [ 2/2] | 692 |
| 16.274.2.78 fmaddin()                                       | [ 2/2] | 693 |
| 16.274.2.79 fmaddx()  | [ 2/2] | 693 |
| 16.274.2.80 fmaddxin()                                      | [ 2/2] | 693 |
| 16.274.2.81 fmsub()   | [ 2/2] | 693 |
| 16.274.2.82 fmsubin()                                       | [ 2/2] | 693 |
| 16.274.2.83 fmsubx()  | [ 2/2] | 693 |
| 16.274.2.84 fmsubxin()                                      | [ 2/2] | 693 |
| 16.274.2.85 eq()  | [ 2/2] | 693 |
| 16.274.2.86 greater()                                       | [ 2/2] | 694 |
| 16.274.2.87 lesser()  | [ 2/2] | 694 |
| 16.274.2.88 greater_eq()                                    | [ 2/2] | 694 |
| 16.274.2.89 lesser_eq()                                     | [ 2/2] | 694 |
| 16.274.2.90 hadd_to_scal()                                  | [ 2/2] | 694 |
| 16.274.2.91 round()   | [ 2/2] | 694 |
| 16.274.2.92 mod()   | [ 2/2] | 694 |
| 16.274.2.93 type_string()                                   | [ 1/2] | 694 |
| 16.274.2.94 zero()  | [ 1/2] | 695 |
| 16.274.2.95 type_string()                                   | [ 2/2] | 695 |
| 16.274.2.96 zero()  | [ 2/2] | 695 |
| 16.274.2.97 vor()   |        | 695 |
| 16.274.2.98 vxor()  |        | 695 |
| 16.274.2.99 vand()  |        | 695 |
| 16.274.2.100 vandnot()                                      |        | 695 |
| 16.274.3 Field Documentation                                |        | 695 |
| 16.274.3.1 vect_size  |        | 695 |
| 16.274.3.2 alignment  |        | 695 |
| 16.275 Simd256_impl< true, true, true, 8 > Struct Reference |        | 696 |
| 16.275.1 Member Typedef Documentation                       |        | 697 |
| 16.275.1.1 vect_t   |        | 697 |
| 16.275.1.2 half_t   |        | 697 |
| 16.275.1.3 scalar_t   |        | 698 |
| 16.275.1.4 simdHalf   |        | 698 |
| 16.275.2 Member Function Documentation                      |        | 698 |
| 16.275.2.1 valid()  |        | 698 |
| 16.275.2.2 compliant()                                      |        | 698 |
| 16.275.2.3 set1()   |        | 698 |
| 16.275.2.4 set()  |        | 698 |
| 16.275.2.5 gather()   |        | 698 |
| 16.275.2.6 get()  |        | 698 |
| 16.275.2.7 load()   |        | 698 |
| 16.275.2.8 loadu()  |        | 699 |



|                              |     |
|------------------------------|-----|
| 16.275.2.9 store()           | 699 |
| 16.275.2.10 storeu()         | 699 |
| 16.275.2.11 stream()         | 699 |
| 16.275.2.12 sll()            | 699 |
| 16.275.2.13 srl()            | 699 |
| 16.275.2.14 sra()            | 699 |
| 16.275.2.15 shuffle()        | 699 |
| 16.275.2.16 unpacklo_twice() | 699 |
| 16.275.2.17 unpackhi_twice() | 700 |
| 16.275.2.18 unpacklo()       | 700 |
| 16.275.2.19 unpackhi()       | 700 |
| 16.275.2.20 unpacklohi()     | 700 |
| 16.275.2.21 blend()          | 700 |
| 16.275.2.22 add()            | 700 |
| 16.275.2.23 addin()          | 700 |
| 16.275.2.24 sub()            | 700 |
| 16.275.2.25 subin()          | 701 |
| 16.275.2.26 mullo()          | 701 |
| 16.275.2.27 mul()            | 701 |
| 16.275.2.28 mulx()           | 701 |
| 16.275.2.29 fmadd()          | 701 |
| 16.275.2.30 fmaddin()        | 701 |
| 16.275.2.31 fmaddx()         | 701 |
| 16.275.2.32 fmaddxin()       | 701 |
| 16.275.2.33 fnmadd()         | 702 |
| 16.275.2.34 fnmaddin()       | 702 |
| 16.275.2.35 fnmaddx()        | 702 |
| 16.275.2.36 fnmaddxin()      | 702 |
| 16.275.2.37 fmsub()          | 702 |
| 16.275.2.38 fmsubin()        | 702 |
| 16.275.2.39 fmsubx()         | 702 |
| 16.275.2.40 fmsubxin()       | 702 |
| 16.275.2.41 eq()             | 703 |
| 16.275.2.42 greater()        | 703 |
| 16.275.2.43 lesser()         | 703 |
| 16.275.2.44 greater_eq()     | 703 |
| 16.275.2.45 lesser_eq()      | 703 |
| 16.275.2.46 hadd_to_scal()   | 703 |
| 16.275.2.47 round()          | 703 |
| 16.275.2.48 mask_high()      | 703 |
| 16.275.2.49 mulhi_fast()     | 704 |
| 16.275.2.50 mod()            | 704 |

|   |     |
|---|-----|
| 16.275.2.51 signbits()  | 704 |
| 16.275.2.52 type_string()   | 704 |
| 16.275.2.53 zero()  | 704 |
| 16.275.3 Field Documentation  | 704 |
| 16.275.3.1 vect_size  | 704 |
| 16.275.3.2 alignment  | 704 |
| 16.276 Simd256fp_base Struct Reference  | 704 |
| 16.277 Simd256i_base Struct Reference   | 705 |
| 16.277.1 Member Typedef Documentation   | 705 |
| 16.277.1.1 vect_t   | 705 |
| 16.277.2 Member Function Documentation  | 705 |
| 16.277.2.1 type_string()  | 705 |
| 16.277.2.2 zero()   | 705 |
| 16.278 Simd512_impl< ArithType, Int, Signed, Size > Struct Template Reference | 706 |
| 16.279 Simd512_impl< true, false, true, 4 > Struct Reference                  | 706 |
| 16.279.1 Member Function Documentation  | 706 |
| 16.279.1.1 type_string()  | 706 |
| 16.280 Simd512_impl< true, false, true, 8 > Struct Reference                  | 706 |
| 16.280.1 Member Typedef Documentation   | 707 |
| 16.280.1.1 vect_t   | 707 |
| 16.280.1.2 scalar_t   | 708 |
| 16.280.2 Member Function Documentation  | 708 |
| 16.280.2.1 valid()  | 708 |
| 16.280.2.2 compliant()  | 708 |
| 16.280.2.3 zero()   | 708 |
| 16.280.2.4 set1()   | 708 |
| 16.280.2.5 set()  | 708 |
| 16.280.2.6 gather()   | 708 |
| 16.280.2.7 load()   | 708 |
| 16.280.2.8 loadu()  | 709 |
| 16.280.2.9 store()  | 709 |
| 16.280.2.10 storeu()  | 709 |
| 16.280.2.11 stream()  | 709 |
| 16.280.2.12 shuffle()   | 709 |
| 16.280.2.13 unpacklo_twice()  | 709 |
| 16.280.2.14 unpackhi_twice()  | 709 |
| 16.280.2.15 blend()   | 709 |
| 16.280.2.16 blendv()  | 709 |
| 16.280.2.17 add()   | 710 |
| 16.280.2.18 addin()   | 710 |
| 16.280.2.19 sub()   | 710 |
| 16.280.2.20 subin()   | 710 |

|  |     |
|--|-----|
| 16.280.2.21 mul()  | 710 |
| 16.280.2.22 mulin()  | 710 |
| 16.280.2.23 div()  | 710 |
| 16.280.2.24 fmadd()  | 710 |
| 16.280.2.25 fmaddin()  | 711 |
| 16.280.2.26 fnmadd()   | 711 |
| 16.280.2.27 fnmaddin()                                       | 711 |
| 16.280.2.28 fmsub()  | 711 |
| 16.280.2.29 fmsubin()  | 711 |
| 16.280.2.30 eq()   | 711 |
| 16.280.2.31 lesser()   | 711 |
| 16.280.2.32 lesser_eq()                                      | 711 |
| 16.280.2.33 greater()  | 712 |
| 16.280.2.34 greater_eq()                                     | 712 |
| 16.280.2.35 floor()  | 712 |
| 16.280.2.36 ceil()   | 712 |
| 16.280.2.37 round()  | 712 |
| 16.280.2.38 hadd()   | 712 |
| 16.280.2.39 hadd_to_scal()                                   | 712 |
| 16.280.2.40 type_string()                                    | 712 |
| 16.280.3 Field Documentation                                 | 712 |
| 16.280.3.1 vect_size   | 712 |
| 16.280.3.2 alignment   | 713 |
| 16.281 Simd512_impl< true, true, false, 8 > Struct Reference | 713 |
| 16.281.1 Member Typedef Documentation                        | 715 |
| 16.281.1.1 scalar_t  | 715 |
| 16.281.1.2 simdHalf  | 715 |
| 16.281.1.3 vect_t  | 715 |
| 16.281.1.4 half_t  | 715 |
| 16.281.2 Member Function Documentation                       | 715 |
| 16.281.2.1 set1() [1/2]                                      | 715 |
| 16.281.2.2 set() [1/3]                                       | 715 |
| 16.281.2.3 gather() [1/2]                                    | 716 |
| 16.281.2.4 load() [1/2]                                      | 716 |
| 16.281.2.5 loadu() [1/2]                                     | 716 |
| 16.281.2.6 store() [1/2]                                     | 716 |
| 16.281.2.7 maskstore() [1/2]                                 | 716 |
| 16.281.2.8 storeu() [1/2]                                    | 716 |
| 16.281.2.9 stream() [1/2]                                    | 716 |
| 16.281.2.10 sra()  | 716 |
| 16.281.2.11 greater()  | 716 |
| 16.281.2.12 lesser()   | 717 |

|                               |     |
|-------------------------------|-----|
| 16.281.2.13 greater_eq()      | 717 |
| 16.281.2.14 lesser_eq()       | 717 |
| 16.281.2.15 mullo()           | 717 |
| 16.281.2.16 mulx()            | 717 |
| 16.281.2.17 fmaddx()          | 717 |
| 16.281.2.18 fmaddxin()        | 717 |
| 16.281.2.19 fnmaddx()         | 717 |
| 16.281.2.20 fnmaddxin()       | 718 |
| 16.281.2.21 fmsubx()          | 718 |
| 16.281.2.22 fmsubxin()        | 718 |
| 16.281.2.23 hadd_to_scal()    | 718 |
| 16.281.2.24 valid()           | 718 |
| 16.281.2.25 compliant()       | 718 |
| 16.281.2.26 set1() [2/2]      | 718 |
| 16.281.2.27 set() [2/3]       | 718 |
| 16.281.2.28 set() [3/3]       | 719 |
| 16.281.2.29 gather() [2/2]    | 719 |
| 16.281.2.30 load() [2/2]      | 719 |
| 16.281.2.31 loadu() [2/2]     | 719 |
| 16.281.2.32 store() [2/2]     | 719 |
| 16.281.2.33 maskstore() [2/2] | 719 |
| 16.281.2.34 storeu() [2/2]    | 719 |
| 16.281.2.35 stream() [2/2]    | 719 |
| 16.281.2.36 sll()             | 719 |
| 16.281.2.37 srl()             | 720 |
| 16.281.2.38 shuffle()         | 720 |
| 16.281.2.39 unpacklo_twice()  | 720 |
| 16.281.2.40 unpackhi_twice()  | 720 |
| 16.281.2.41 unpacklo()        | 720 |
| 16.281.2.42 unpackhi()        | 720 |
| 16.281.2.43 unpacklohi()      | 720 |
| 16.281.2.44 blend()           | 720 |
| 16.281.2.45 add()             | 721 |
| 16.281.2.46 addin()           | 721 |
| 16.281.2.47 sub()             | 721 |
| 16.281.2.48 subin()           | 721 |
| 16.281.2.49 mul()             | 721 |
| 16.281.2.50 fmadd()           | 721 |
| 16.281.2.51 fmaddin()         | 721 |
| 16.281.2.52 fnmadd()          | 721 |
| 16.281.2.53 fnmaddin()        | 722 |
| 16.281.2.54 fmsub()           | 722 |

|   |     |
|---|-----|
| 16.281.2.55 fmsubin()                                       | 722 |
| 16.281.2.56 eq()  | 722 |
| 16.281.2.57 round()   | 722 |
| 16.281.2.58 mask_high()                                     | 722 |
| 16.281.2.59 mulhi_fast()                                    | 722 |
| 16.281.2.60 mod()   | 722 |
| 16.281.2.61 signbits()                                      | 723 |
| 16.281.2.62 type_string()                                   | 723 |
| 16.281.2.63 zero()  | 723 |
| 16.281.2.64 vor()   | 723 |
| 16.281.2.65 vxor()  | 723 |
| 16.281.2.66 vand()  | 723 |
| 16.281.2.67 vandnot()                                       | 723 |
| 16.281.3 Field Documentation                                | 723 |
| 16.281.3.1 vect_size  | 723 |
| 16.281.3.2 alignment  | 724 |
| 16.282 Simd512_impl< true, true, true, 8 > Struct Reference | 724 |
| 16.282.1 Member Typedef Documentation                       | 726 |
| 16.282.1.1 vect_t   | 726 |
| 16.282.1.2 half_t   | 726 |
| 16.282.1.3 scalar_t   | 726 |
| 16.282.1.4 simdHalf   | 726 |
| 16.282.2 Member Function Documentation                      | 726 |
| 16.282.2.1 valid()  | 726 |
| 16.282.2.2 compliant()                                      | 726 |
| 16.282.2.3 set1()   | 726 |
| 16.282.2.4 set() [1/2]                                      | 726 |
| 16.282.2.5 set() [2/2]                                      | 727 |
| 16.282.2.6 gather()   | 727 |
| 16.282.2.7 load()   | 727 |
| 16.282.2.8 loadu()  | 727 |
| 16.282.2.9 store()  | 727 |
| 16.282.2.10 maskstore()                                     | 727 |
| 16.282.2.11 storeu()  | 727 |
| 16.282.2.12 stream()  | 727 |
| 16.282.2.13 sll()   | 727 |
| 16.282.2.14 srl()   | 728 |
| 16.282.2.15 sra()   | 728 |
| 16.282.2.16 shuffle()                                       | 728 |
| 16.282.2.17 unpacklo_twice()                                | 728 |
| 16.282.2.18 unpackhi_twice()                                | 728 |
| 16.282.2.19 unpacklo()                                      | 728 |

|                              |     |
|------------------------------|-----|
| 16.282.2.20 unpackhi()       | 728 |
| 16.282.2.21 unpacklohi()     | 728 |
| 16.282.2.22 blend()          | 729 |
| 16.282.2.23 add()            | 729 |
| 16.282.2.24 addin()          | 729 |
| 16.282.2.25 sub()            | 729 |
| 16.282.2.26 subin()          | 729 |
| 16.282.2.27 mullo()          | 729 |
| 16.282.2.28 mul()            | 729 |
| 16.282.2.29 mulx()           | 729 |
| 16.282.2.30 fmadd()          | 729 |
| 16.282.2.31 fmaddin()        | 730 |
| 16.282.2.32 fmaddx()         | 730 |
| 16.282.2.33 fmaddxin()       | 730 |
| 16.282.2.34 fnmadd()         | 730 |
| 16.282.2.35 fnmaddin()       | 730 |
| 16.282.2.36 fnmaddx()        | 730 |
| 16.282.2.37 fnmaddxin()      | 730 |
| 16.282.2.38 fmsub()          | 731 |
| 16.282.2.39 fmsubin()        | 731 |
| 16.282.2.40 fmsubx()         | 731 |
| 16.282.2.41 fmsubxin()       | 731 |
| 16.282.2.42 eq()             | 731 |
| 16.282.2.43 greater()        | 731 |
| 16.282.2.44 lesser()         | 731 |
| 16.282.2.45 greater_eq()     | 731 |
| 16.282.2.46 lesser_eq()      | 732 |
| 16.282.2.47 hadd_to_scal()   | 732 |
| 16.282.2.48 round()          | 732 |
| 16.282.2.49 mask_high()      | 732 |
| 16.282.2.50 mulhi_fast()     | 732 |
| 16.282.2.51 mod()            | 732 |
| 16.282.2.52 signbits()       | 732 |
| 16.282.2.53 type_string()    | 732 |
| 16.282.2.54 zero()           | 732 |
| 16.282.2.55 vor()            | 733 |
| 16.282.2.56 vxor()           | 733 |
| 16.282.2.57 vand()           | 733 |
| 16.282.2.58 vandnot()        | 733 |
| 16.282.3 Field Documentation | 733 |
| 16.282.3.1 vect_size         | 733 |
| 16.282.3.2 alignment         | 733 |

|  |     |
|--|-----|
| 16.283 Simd512fp_base Struct Reference . . . . .                                       | 733 |
| 16.283.1 Member Function Documentation . . . . .                                       | 734 |
| 16.283.1.1 type_string() . . . . .   | 734 |
| 16.284 Simd512i_base Struct Reference . . . . .  | 734 |
| 16.284.1 Member Typedef Documentation . . . . .  | 734 |
| 16.284.1.1 vect_t . . . . .  | 734 |
| 16.284.2 Member Function Documentation . . . . .                                       | 734 |
| 16.284.2.1 type_string() . . . . .   | 734 |
| 16.284.2.2 zero() . . . . .  | 734 |
| 16.284.2.3 vor() . . . . .   | 735 |
| 16.284.2.4 vxor() . . . . .  | 735 |
| 16.284.2.5 vand() . . . . .  | 735 |
| 16.284.2.6 vandnot() . . . . .   | 735 |
| 16.285 SimdChooser< T, bool, bool > Struct Template Reference . . . . .                | 735 |
| 16.286 SimdChooser< T, false, b > Struct Template Reference . . . . .                  | 735 |
| 16.286.1 Member Typedef Documentation . . . . .  | 735 |
| 16.286.1.1 value . . . . .   | 735 |
| 16.287 SimdChooser< T, true, false > Struct Template Reference . . . . .               | 735 |
| 16.287.1 Member Typedef Documentation . . . . .  | 736 |
| 16.287.1.1 value . . . . .   | 736 |
| 16.288 SimdChooser< T, true, true > Struct Template Reference . . . . .                | 736 |
| 16.288.1 Member Typedef Documentation . . . . .  | 736 |
| 16.288.1.1 value . . . . .   | 736 |
| 16.289 simdToType< T > Struct Template Reference . . . . .                             | 736 |
| 16.290 Single Struct Reference . . . . .   | 736 |
| 16.291 Sparse< Field, SparseMatrix_t, IdxT, PtrT > Struct Template Reference . . . . . | 736 |
| 16.292 Sparse< _Field, SparseMatrix_t::COO > Struct Template Reference . . . . .       | 737 |
| 16.292.1 Member Typedef Documentation . . . . .  | 737 |
| 16.292.1.1 Field . . . . .   | 737 |
| 16.292.2 Field Documentation . . . . .   | 737 |
| 16.292.2.1 col . . . . .   | 737 |
| 16.292.2.2 row . . . . .   | 737 |
| 16.292.2.3 dat . . . . .   | 737 |
| 16.292.2.4 delayed . . . . .   | 738 |
| 16.292.2.5 kmax . . . . .  | 738 |
| 16.292.2.6 m . . . . .   | 738 |
| 16.292.2.7 n . . . . .   | 738 |
| 16.292.2.8 nnz . . . . .   | 738 |
| 16.292.2.9 nElements . . . . .   | 738 |
| 16.292.2.10 maxrow . . . . .   | 738 |
| 16.293 Sparse< _Field, SparseMatrix_t::COO_ZO > Struct Template Reference . . . . .    | 738 |
| 16.293.1 Member Typedef Documentation . . . . .  | 739 |

|  |     |
|--|-----|
| 16.293.1.1 Field   | 739 |
| 16.293.2 Field Documentation   | 739 |
| 16.293.2.1 cst   | 739 |
| 16.293.2.2 col   | 739 |
| 16.293.2.3 row   | 739 |
| 16.293.2.4 dat   | 739 |
| 16.293.2.5 delayed   | 739 |
| 16.293.2.6 kmax  | 739 |
| 16.293.2.7 m   | 739 |
| 16.293.2.8 n   | 739 |
| 16.293.2.9 nnz   | 740 |
| 16.293.2.10 nElements  | 740 |
| 16.293.2.11 maxrow   | 740 |
| 16.294 Sparse< _Field, SparseMatrix_t::CSR > Struct Template Reference     | 740 |
| 16.294.1 Member Typedef Documentation                                      | 740 |
| 16.294.1.1 Field   | 740 |
| 16.294.2 Field Documentation   | 741 |
| 16.294.2.1 delayed   | 741 |
| 16.294.2.2 kmax  | 741 |
| 16.294.2.3 m   | 741 |
| 16.294.2.4 n   | 741 |
| 16.294.2.5 nnz   | 741 |
| 16.294.2.6 nElements   | 741 |
| 16.294.2.7 maxrow  | 741 |
| 16.294.2.8 col   | 741 |
| 16.294.2.9 st  | 741 |
| 16.294.2.10 stend  | 741 |
| 16.294.2.11 dat  | 741 |
| 16.295 Sparse< _Field, SparseMatrix_t::CSR_HYB > Struct Template Reference | 742 |
| 16.295.1 Member Typedef Documentation                                      | 742 |
| 16.295.1.1 Field   | 742 |
| 16.295.2 Field Documentation   | 742 |
| 16.295.2.1 delayed   | 742 |
| 16.295.2.2 col   | 742 |
| 16.295.2.3 st  | 742 |
| 16.295.2.4 dat   | 742 |
| 16.295.2.5 kmax  | 743 |
| 16.295.2.6 m   | 743 |
| 16.295.2.7 n   | 743 |
| 16.295.2.8 nnz   | 743 |
| 16.295.2.9 nElements   | 743 |
| 16.295.2.10 maxrow   | 743 |



|   |     |
|---|-----|
| 16.295.2.11 nOnes   | 743 |
| 16.295.2.12 nMOnes  | 743 |
| 16.295.2.13 nOthers   | 743 |
| 16.296 Sparse< _Field, SparseMatrix_t::CSR_ZO > Struct Template Reference   | 743 |
| 16.296.1 Member Typedef Documentation                                       | 744 |
| 16.296.1.1 Field  | 744 |
| 16.296.2 Field Documentation  | 744 |
| 16.296.2.1 cst  | 744 |
| 16.296.2.2 delayed  | 744 |
| 16.296.2.3 kmax   | 744 |
| 16.296.2.4 m  | 744 |
| 16.296.2.5 n  | 744 |
| 16.296.2.6 nnz  | 744 |
| 16.296.2.7 nElements  | 745 |
| 16.296.2.8 maxrow   | 745 |
| 16.296.2.9 col  | 745 |
| 16.296.2.10 st  | 745 |
| 16.296.2.11 stend   | 745 |
| 16.296.2.12 dat   | 745 |
| 16.297 Sparse< _Field, SparseMatrix_t::ELL > Struct Template Reference      | 745 |
| 16.297.1 Member Typedef Documentation                                       | 746 |
| 16.297.1.1 Field  | 746 |
| 16.297.2 Field Documentation  | 746 |
| 16.297.2.1 delayed  | 746 |
| 16.297.2.2 kmax   | 746 |
| 16.297.2.3 m  | 746 |
| 16.297.2.4 n  | 746 |
| 16.297.2.5 ld   | 746 |
| 16.297.2.6 nnz  | 746 |
| 16.297.2.7 nElements  | 746 |
| 16.297.2.8 maxrow   | 746 |
| 16.297.2.9 col  | 746 |
| 16.297.2.10 dat   | 747 |
| 16.298 Sparse< _Field, SparseMatrix_t::ELL_simd > Struct Template Reference | 747 |
| 16.298.1 Field Documentation  | 747 |
| 16.298.1.1 delayed  | 747 |
| 16.298.1.2 chunk  | 747 |
| 16.298.1.3 m  | 747 |
| 16.298.1.4 n  | 747 |
| 16.298.1.5 ld   | 748 |
| 16.298.1.6 kmax   | 748 |
| 16.298.1.7 nnz  | 748 |

|  |     |
|--|-----|
| 16.298.1.8 nElements   | 748 |
| 16.298.1.9 maxrow  | 748 |
| 16.298.1.10 nChunks  | 748 |
| 16.298.1.11 col  | 748 |
| 16.298.1.12 dat  | 748 |
| 16.299 Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > Struct Template Reference | 748 |
| 16.299.1 Field Documentation   | 749 |
| 16.299.1.1 cst   | 749 |
| 16.299.1.2 delayed   | 749 |
| 16.299.1.3 chunk   | 749 |
| 16.299.1.4 m   | 749 |
| 16.299.1.5 n   | 749 |
| 16.299.1.6 ld  | 749 |
| 16.299.1.7 kmax  | 749 |
| 16.299.1.8 nnz   | 749 |
| 16.299.1.9 nElements   | 750 |
| 16.299.1.10 maxrow   | 750 |
| 16.299.1.11 nChunks  | 750 |
| 16.299.1.12 col  | 750 |
| 16.299.1.13 dat  | 750 |
| 16.300 Sparse< _Field, SparseMatrix_t::ELL_ZO > Struct Template Reference      | 750 |
| 16.300.1 Member Typedef Documentation  | 751 |
| 16.300.1.1 Field   | 751 |
| 16.300.2 Field Documentation   | 751 |
| 16.300.2.1 cst   | 751 |
| 16.300.2.2 delayed   | 751 |
| 16.300.2.3 kmax  | 751 |
| 16.300.2.4 m   | 751 |
| 16.300.2.5 n   | 751 |
| 16.300.2.6 ld  | 751 |
| 16.300.2.7 nnz   | 751 |
| 16.300.2.8 nElements   | 751 |
| 16.300.2.9 maxrow  | 751 |
| 16.300.2.10 col  | 751 |
| 16.300.2.11 dat  | 752 |
| 16.301 Sparse< _Field, SparseMatrix_t::HYB_ZO > Struct Template Reference      | 752 |
| 16.301.1 Member Typedef Documentation  | 752 |
| 16.301.1.1 Field   | 752 |
| 16.301.1.2 Self_t  | 752 |
| 16.301.2 Field Documentation   | 752 |
| 16.301.2.1 delayed   | 752 |
| 16.301.2.2 kmax  | 752 |

|  |     |
|--|-----|
| 16.301.2.3 m   | 753 |
| 16.301.2.4 n   | 753 |
| 16.301.2.5 nnz   | 753 |
| 16.301.2.6 maxrow  | 753 |
| 16.301.2.7 nElements   | 753 |
| 16.301.2.8 dat   | 753 |
| 16.301.2.9 one   | 753 |
| 16.301.2.10 mone   | 753 |
| 16.302 Sparse< _Field, SparseMatrix_t::SELL > Struct Template Reference    | 753 |
| 16.302.1 Member Typedef Documentation                                      | 754 |
| 16.302.1.1 Field   | 754 |
| 16.302.2 Field Documentation   | 754 |
| 16.302.2.1 delayed   | 754 |
| 16.302.2.2 chunk   | 754 |
| 16.302.2.3 kmax  | 754 |
| 16.302.2.4 m   | 754 |
| 16.302.2.5 n   | 754 |
| 16.302.2.6 maxrow  | 754 |
| 16.302.2.7 sigma   | 755 |
| 16.302.2.8 nChunks   | 755 |
| 16.302.2.9 nnz   | 755 |
| 16.302.2.10 nElements  | 755 |
| 16.302.2.11 perm   | 755 |
| 16.302.2.12 st   | 755 |
| 16.302.2.13 chunkSize  | 755 |
| 16.302.2.14 col  | 755 |
| 16.302.2.15 dat  | 755 |
| 16.303 Sparse< _Field, SparseMatrix_t::SELL_ZO > Struct Template Reference | 755 |
| 16.303.1 Member Typedef Documentation                                      | 756 |
| 16.303.1.1 Field   | 756 |
| 16.303.2 Field Documentation   | 756 |
| 16.303.2.1 cst   | 756 |
| 16.303.2.2 delayed   | 756 |
| 16.303.2.3 chunk   | 756 |
| 16.303.2.4 kmax  | 756 |
| 16.303.2.5 m   | 756 |
| 16.303.2.6 n   | 757 |
| 16.303.2.7 maxrow  | 757 |
| 16.303.2.8 sigma   | 757 |
| 16.303.2.9 nChunks   | 757 |
| 16.303.2.10 nnz  | 757 |
| 16.303.2.11 nElements  | 757 |

|  |     |
|--|-----|
| 16.303.2.12 perm   | 757 |
| 16.303.2.13 st   | 757 |
| 16.303.2.14 chunkSize                                      | 757 |
| 16.303.2.15 col  | 757 |
| 16.303.2.16 dat  | 757 |
| 16.304 SpMat< Field, flag > Struct Template Reference      | 757 |
| 16.304.1 Field Documentation                               | 758 |
| 16.304.1.1 _coo  | 758 |
| 16.304.1.2 _csr  | 758 |
| 16.304.1.3 _ell  | 758 |
| 16.305 Static_error_check< bool > Class Template Reference | 758 |
| 16.305.1 Constructor & Destructor Documentation            | 758 |
| 16.305.1.1 Static_error_check()                            | 758 |
| 16.306 Static_error_check< false > Class Reference         | 758 |
| 16.307 StatsMatrix Struct Reference                        | 758 |
| 16.307.1 Field Documentation                               | 759 |
| 16.307.1.1 rowdim  | 759 |
| 16.307.1.2 coldim  | 759 |
| 16.307.1.3 nOnes   | 759 |
| 16.307.1.4 nMOnes  | 759 |
| 16.307.1.5 nOthers   | 760 |
| 16.307.1.6 nnz   | 760 |
| 16.307.1.7 maxRow  | 760 |
| 16.307.1.8 minRow  | 760 |
| 16.307.1.9 averageRow                                      | 760 |
| 16.307.1.10 deviationRow                                   | 760 |
| 16.307.1.11 maxCol   | 760 |
| 16.307.1.12 minCol   | 760 |
| 16.307.1.13 averageCol                                     | 760 |
| 16.307.1.14 deviationCol                                   | 760 |
| 16.307.1.15 minColDifference                               | 760 |
| 16.307.1.16 maxColDifference                               | 760 |
| 16.307.1.17 averageColDifference                           | 761 |
| 16.307.1.18 deviationColDifference                         | 761 |
| 16.307.1.19 minRowDifference                               | 761 |
| 16.307.1.20 maxRowDifference                               | 761 |
| 16.307.1.21 averageRowDifference                           | 761 |
| 16.307.1.22 deviationRowDifference                         | 761 |
| 16.307.1.23 nDenseRows                                     | 761 |
| 16.307.1.24 nDenseCols                                     | 761 |
| 16.307.1.25 nEmptyRows                                     | 761 |
| 16.307.1.26 nEmptyCols                                     | 761 |

|  |     |
|--|-----|
| 16.307.1.27 nEmptyColsEnd . . . . .  | 761 |
| 16.307.1.28 denseRows . . . . .  | 761 |
| 16.307.1.29 denseCols . . . . .  | 762 |
| 16.308 support_fast_mod< T > Struct Template Reference . . . . .                   | 762 |
| 16.309 support_fast_mod< double > Struct Reference . . . . .                       | 762 |
| 16.310 support_fast_mod< float > Struct Reference . . . . .                        | 762 |
| 16.311 support_fast_mod< int64_t > Struct Reference . . . . .                      | 762 |
| 16.312 support_simd< T > Struct Template Reference . . . . .                       | 763 |
| 16.313 support_simd_add< T > Struct Template Reference . . . . .                   | 763 |
| 16.314 support_simd_mod< T > Struct Template Reference . . . . .                   | 763 |
| 16.315 tfn_minus Struct Reference . . . . .  | 764 |
| 16.315.1 Member Function Documentation . . . . .                                   | 764 |
| 16.315.1.1 operator>() . . . . .   | 764 |
| 16.316 tfn_minus_eq Struct Reference . . . . .                                     | 764 |
| 16.316.1 Member Function Documentation . . . . .                                   | 764 |
| 16.316.1.1 operator>() . . . . .   | 764 |
| 16.317 tfn_mul Struct Reference . . . . .  | 764 |
| 16.317.1 Member Function Documentation . . . . .                                   | 764 |
| 16.317.1.1 operator>() . . . . .   | 765 |
| 16.318 tfn_mul_eq Struct Reference . . . . .                                       | 765 |
| 16.318.1 Member Function Documentation . . . . .                                   | 765 |
| 16.318.1.1 operator>() . . . . .   | 765 |
| 16.319 tfn_plus Struct Reference . . . . .   | 765 |
| 16.319.1 Member Function Documentation . . . . .                                   | 765 |
| 16.319.1.1 operator>() . . . . .   | 765 |
| 16.320 tfn_plus_eq Struct Reference . . . . .                                      | 765 |
| 16.320.1 Member Function Documentation . . . . .                                   | 766 |
| 16.320.1.1 operator>() . . . . .   | 766 |
| 16.321 Threads Struct Reference . . . . .  | 766 |
| 16.322 ThreeD Struct Reference . . . . .   | 766 |
| 16.323 ThreeDAdaptive Struct Reference . . . . .                                   | 766 |
| 16.324 ThreeDInPlace Struct Reference . . . . .                                    | 766 |
| 16.325 TRSMHelper< ReclterTrait, ParSeqTrait > Struct Template Reference . . . . . | 766 |
| 16.325.1 Detailed Description . . . . .  | 767 |
| 16.325.2 Constructor & Destructor Documentation . . . . .                          | 767 |
| 16.325.2.1 TRSMHelper() [1/3] . . . . .  | 767 |
| 16.325.2.2 TRSMHelper() [2/3] . . . . .  | 767 |
| 16.325.2.3 TRSMHelper() [3/3] . . . . .  | 767 |
| 16.325.3 Member Function Documentation . . . . .                                   | 767 |
| 16.325.3.1 pMMH() [1/2] . . . . .  | 767 |
| 16.325.3.2 pMMH() [2/2] . . . . .  | 767 |
| 16.325.4 Field Documentation . . . . .   | 767 |

|   |            |
|---|------------|
| 16.325.4.1 parseq . . . . .                       | 768        |
| 16.326 TwoD Struct Reference . . . . .            | 768        |
| 16.327 TwoDAdaptive Struct Reference . . . . .    | 768        |
| 16.328 UnparametricTag Struct Reference . . . . . | 768        |
| 16.328.1 Detailed Description . . . . .           | 768        |
| 16.329 Winograd Struct Reference . . . . .        | 768        |
| 16.330 WinogradPar Struct Reference . . . . .     | 768        |
| <b>17 File Documentation</b>                      | <b>769</b> |
| 17.1 arithprog.C File Reference . . . . .         | 769        |
| 17.1.1 Macro Definition Documentation . . . . .   | 769        |
| 17.1.1.1 CUBE . . . . .                           | 769        |
| 17.1.1.2 GFOPS . . . . .                          | 769        |
| 17.1.2 Typedef Documentation . . . . .            | 770        |
| 17.1.2.1 TTimer . . . . .                         | 770        |
| 17.1.3 Function Documentation . . . . .           | 770        |
| 17.1.3.1 main() . . . . .                         | 770        |
| 17.2 charpoly.C File Reference . . . . .          | 770        |
| 17.2.1 Macro Definition Documentation . . . . .   | 770        |
| 17.2.1.1 CUBE . . . . .                           | 770        |
| 17.2.1.2 GFOPS . . . . .                          | 770        |
| 17.2.2 Typedef Documentation . . . . .            | 771        |
| 17.2.2.1 TTimer . . . . .                         | 771        |
| 17.2.3 Function Documentation . . . . .           | 771        |
| 17.2.3.1 main() . . . . .                         | 771        |
| 17.3 charpoly.C File Reference . . . . .          | 771        |
| 17.3.1 Function Documentation . . . . .           | 771        |
| 17.3.1.1 main() . . . . .                         | 771        |
| 17.4 fsyrk.C File Reference . . . . .             | 771        |
| 17.4.1 Macro Definition Documentation . . . . .   | 772        |
| 17.4.1.1 CUBE . . . . .                           | 772        |
| 17.4.1.2 GFOPS . . . . .                          | 772        |
| 17.4.2 Typedef Documentation . . . . .            | 772        |
| 17.4.2.1 TTimer . . . . .                         | 772        |
| 17.4.3 Function Documentation . . . . .           | 772        |
| 17.4.3.1 main() . . . . .                         | 772        |
| 17.5 fsytrf.C File Reference . . . . .            | 772        |
| 17.5.1 Macro Definition Documentation . . . . .   | 773        |
| 17.5.1.1 CUBE . . . . .                           | 773        |
| 17.5.1.2 GFOPS . . . . .                          | 773        |
| 17.5.2 Typedef Documentation . . . . .            | 773        |
| 17.5.2.1 TTimer . . . . .                         | 773        |

|   |     |
|---|-----|
| 17.5.3 Function Documentation . . . . .                   | 773 |
| 17.5.3.1 main() . . . . .                                 | 773 |
| 17.6 ftrtri.C File Reference . . . . .                    | 773 |
| 17.6.1 Macro Definition Documentation . . . . .           | 774 |
| 17.6.1.1 CUBE . . . . .                                   | 774 |
| 17.6.1.2 GFOPS . . . . .                                  | 774 |
| 17.6.2 Typedef Documentation . . . . .                    | 774 |
| 17.6.2.1 TTimer . . . . .                                 | 774 |
| 17.6.3 Function Documentation . . . . .                   | 774 |
| 17.6.3.1 main() . . . . .                                 | 774 |
| 17.7 pluq.C File Reference . . . . .                      | 774 |
| 17.7.1 Macro Definition Documentation . . . . .           | 775 |
| 17.7.1.1 CUBE . . . . .                                   | 775 |
| 17.7.1.2 GFOPS . . . . .                                  | 775 |
| 17.7.2 Typedef Documentation . . . . .                    | 775 |
| 17.7.2.1 TTimer . . . . .                                 | 775 |
| 17.7.3 Function Documentation . . . . .                   | 775 |
| 17.7.3.1 main() . . . . .                                 | 775 |
| 17.8 pluq.C File Reference . . . . .                      | 775 |
| 17.8.1 Function Documentation . . . . .                   | 776 |
| 17.8.1.1 main() . . . . .                                 | 776 |
| 17.9 winograd.C File Reference . . . . .                  | 776 |
| 17.9.1 Macro Definition Documentation . . . . .           | 776 |
| 17.9.1.1 DOUBLE_TO_FLOAT_CROSSOVER . . . . .              | 776 |
| 17.9.1.2 GFOPS . . . . .                                  | 776 |
| 17.9.2 Typedef Documentation . . . . .                    | 776 |
| 17.9.2.1 TTimer . . . . .                                 | 777 |
| 17.9.3 Function Documentation . . . . .                   | 777 |
| 17.9.3.1 balanced() [1/2] . . . . .                       | 777 |
| 17.9.3.2 balanced() [2/2] . . . . .                       | 777 |
| 17.9.3.3 main() . . . . .                                 | 777 |
| 17.10 benchmark-charpoly-mp.C File Reference . . . . .    | 777 |
| 17.10.1 Macro Definition Documentation . . . . .          | 777 |
| 17.10.1.1 __FFLASFFPACK_FORCE_SEQ . . . . .               | 777 |
| 17.10.2 Function Documentation . . . . .                  | 777 |
| 17.10.2.1 main() . . . . .                                | 778 |
| 17.11 benchmark-charpoly.C File Reference . . . . .       | 778 |
| 17.11.1 Macro Definition Documentation . . . . .          | 778 |
| 17.11.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . . | 778 |
| 17.11.2 Function Documentation . . . . .                  | 778 |
| 17.11.2.1 run_with_field() . . . . .                      | 778 |
| 17.11.2.2 main() . . . . .                                | 778 |

|   |     |
|---|-----|
| 17.12 benchmark-checkers.C File Reference . . . . . | 779 |
| 17.12.1 Macro Definition Documentation . . . . .    | 779 |
| 17.12.1.1 ENABLE_ALL_CHECKINGS . . . . .            | 779 |
| 17.12.1.2 _NR_TESTS . . . . .                       | 779 |
| 17.12.1.3 _MAX_SIZE_MATRICES . . . . .              | 779 |
| 17.12.1.4 CUBE . . . . .                            | 779 |
| 17.12.2 Function Documentation . . . . .            | 779 |
| 17.12.2.1 main() . . . . .                          | 779 |
| 17.13 benchmark-dgemm.C File Reference . . . . .    | 780 |
| 17.13.1 Macro Definition Documentation . . . . .    | 780 |
| 17.13.1.1 CBLAS_GEMM . . . . .                      | 780 |
| 17.13.2 Typedef Documentation . . . . .             | 780 |
| 17.13.2.1 TTimer . . . . .                          | 780 |
| 17.13.2.2 Floats . . . . .                          | 780 |
| 17.13.3 Function Documentation . . . . .            | 780 |
| 17.13.3.1 main() . . . . .                          | 780 |
| 17.14 benchmark-dgetrf.C File Reference . . . . .   | 781 |
| 17.14.1 Macro Definition Documentation . . . . .    | 781 |
| 17.14.1.1 __FFLASFFPACK_HAVE_DGETRF . . . . .       | 781 |
| 17.14.2 Typedef Documentation . . . . .             | 781 |
| 17.14.2.1 TTimer . . . . .                          | 781 |
| 17.14.3 Function Documentation . . . . .            | 781 |
| 17.14.3.1 main() . . . . .                          | 781 |
| 17.15 benchmark-dgetri.C File Reference . . . . .   | 781 |
| 17.15.1 Typedef Documentation . . . . .             | 782 |
| 17.15.1.1 TTimer . . . . .                          | 782 |
| 17.15.2 Function Documentation . . . . .            | 782 |
| 17.15.2.1 main() . . . . .                          | 782 |
| 17.16 benchmark-dsytrf.C File Reference . . . . .   | 782 |
| 17.16.1 Macro Definition Documentation . . . . .    | 782 |
| 17.16.1.1 EFFGFF . . . . .                          | 783 |
| 17.16.2 Typedef Documentation . . . . .             | 783 |
| 17.16.2.1 TTimer . . . . .                          | 783 |
| 17.16.3 Function Documentation . . . . .            | 783 |
| 17.16.3.1 main() . . . . .                          | 783 |
| 17.17 benchmark-dtrsm.C File Reference . . . . .    | 783 |
| 17.17.1 Typedef Documentation . . . . .             | 783 |
| 17.17.1.1 TTimer . . . . .                          | 783 |
| 17.17.2 Function Documentation . . . . .            | 783 |
| 17.17.2.1 main() . . . . .                          | 784 |
| 17.18 benchmark-dtrtri.C File Reference . . . . .   | 784 |
| 17.18.1 Macro Definition Documentation . . . . .    | 784 |



|  |     |
|--|-----|
| 17.18.1.1 <code>__FFLASFFPACK_HAVE_DTRTRI</code>             | 784 |
| 17.18.2 Typedef Documentation                                | 784 |
| 17.18.2.1 <code>TTimer</code>                                | 784 |
| 17.18.3 Function Documentation                               | 784 |
| 17.18.3.1 <code>main()</code>                                | 784 |
| 17.19 benchmark-fadd-lvl2.C File Reference                   | 785 |
| 17.19.1 Macro Definition Documentation                       | 785 |
| 17.19.1.1 <code>__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET</code> | 785 |
| 17.19.2 Function Documentation                               | 785 |
| 17.19.2.1 <code>main()</code>                                | 785 |
| 17.20 benchmark-fdot.C File Reference                        | 785 |
| 17.20.1 Macro Definition Documentation                       | 786 |
| 17.20.1.1 <code>__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET</code> | 786 |
| 17.20.2 Function Documentation                               | 786 |
| 17.20.2.1 <code>run_with_field()</code>                      | 786 |
| 17.20.2.2 <code>main()</code>                                | 786 |
| 17.21 benchmark-fgemm-mp.C File Reference                    | 786 |
| 17.21.1 Macro Definition Documentation                       | 787 |
| 17.21.1.1 <code>__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET</code> | 787 |
| 17.21.1.2 <code>MG_DEFAULT</code>                            | 787 |
| 17.21.1.3 <code>STD_RECINT_SIZE</code>                       | 787 |
| 17.21.2 Function Documentation                               | 787 |
| 17.21.2.1 <code>tmain()</code>                               | 787 |
| 17.21.2.2 <code>main()</code>                                | 787 |
| 17.22 benchmark-fgemm-rns.C File Reference                   | 787 |
| 17.22.1 Macro Definition Documentation                       | 788 |
| 17.22.1.1 <code>__FFLASFFPACK_OPENBLAS_NT_ALREADY_SET</code> | 788 |
| 17.22.2 Typedef Documentation                                | 788 |
| 17.22.2.1 <code>RNS</code>                                   | 788 |
| 17.22.2.2 <code>Field</code>                                 | 788 |
| 17.22.2.3 <code>Element_ptr</code>                           | 788 |
| 17.22.2.4 <code>ConstElement_ptr</code>                      | 788 |
| 17.22.2.5 <code>THREADS</code>                               | 788 |
| 17.22.2.6 <code>GRAIN</code>                                 | 788 |
| 17.22.2.7 <code>TWOD</code>                                  | 788 |
| 17.22.2.8 <code>TWODA</code>                                 | 789 |
| 17.22.2.9 <code>THREED</code>                                | 789 |
| 17.22.2.10 <code>THREEDA</code>                              | 789 |
| 17.22.2.11 <code>THREEDIP</code>                             | 789 |
| 17.22.2.12 <code>PSeq</code>                                 | 789 |
| 17.22.3 Function Documentation                               | 789 |
| 17.22.3.1 <code>main()</code>                                | 789 |

|   |     |
|---|-----|
| 17.23 benchmark-fgemm.C File Reference          | 789 |
| 17.23.1 Macro Definition Documentation          | 789 |
| 17.23.1.1 CLASSIC_HYBRID                        | 790 |
| 17.23.2 Function Documentation                  | 790 |
| 17.23.2.1 main()                                | 790 |
| 17.24 benchmark-fgemv-mp.C File Reference       | 790 |
| 17.24.1 Macro Definition Documentation          | 790 |
| 17.24.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET | 790 |
| 17.24.1.2 MG_DEFAULT                            | 790 |
| 17.24.1.3 STD_RECINT_SIZE                       | 791 |
| 17.24.2 Function Documentation                  | 791 |
| 17.24.2.1 write_matrix()                        | 791 |
| 17.24.2.2 tmain()                               | 791 |
| 17.24.2.3 main()                                | 791 |
| 17.25 benchmark-fgemv.C File Reference          | 791 |
| 17.25.1 Macro Definition Documentation          | 792 |
| 17.25.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET | 792 |
| 17.25.2 Function Documentation                  | 792 |
| 17.25.2.1 fill_value()                          | 792 |
| 17.25.2.2 genData()                             | 792 |
| 17.25.2.3 check_result()                        | 793 |
| 17.25.2.4 benchmark_with_timer()                | 793 |
| 17.25.2.5 benchmark_disp()                      | 793 |
| 17.25.2.6 benchmark_in_Field()                  | 794 |
| 17.25.2.7 benchmark_with_field() [1/2]          | 794 |
| 17.25.2.8 benchmark_with_field() [2/2]          | 794 |
| 17.25.2.9 main()                                | 794 |
| 17.26 benchmark-fgesv.C File Reference          | 794 |
| 17.26.1 Macro Definition Documentation          | 795 |
| 17.26.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET | 795 |
| 17.26.2 Function Documentation                  | 795 |
| 17.26.2.1 main()                                | 795 |
| 17.27 benchmark-fsyrk.C File Reference          | 795 |
| 17.27.1 Macro Definition Documentation          | 795 |
| 17.27.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET | 796 |
| 17.27.1.2 CUBE                                  | 796 |
| 17.27.2 Function Documentation                  | 796 |
| 17.27.2.1 main()                                | 796 |
| 17.28 benchmark-fsytrf.C File Reference         | 796 |
| 17.28.1 Macro Definition Documentation          | 796 |
| 17.28.1.1 __FFPACK_FSYTRF_BC_CROUT              | 796 |
| 17.28.1.2 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET | 796 |

|   |     |
|---|-----|
| 17.28.1.3 CUBE . . . . .                                  | 796 |
| 17.28.2 Function Documentation . . . . .                  | 797 |
| 17.28.2.1 main() . . . . .                                | 797 |
| 17.29 benchmark-ftsrm-mp.C File Reference . . . . .       | 797 |
| 17.29.1 Macro Definition Documentation . . . . .          | 797 |
| 17.29.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . . | 797 |
| 17.29.2 Function Documentation . . . . .                  | 797 |
| 17.29.2.1 main() . . . . .                                | 797 |
| 17.30 benchmark-ftsrm.C File Reference . . . . .          | 797 |
| 17.30.1 Macro Definition Documentation . . . . .          | 798 |
| 17.30.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . . | 798 |
| 17.30.2 Function Documentation . . . . .                  | 798 |
| 17.30.2.1 main() . . . . .                                | 798 |
| 17.31 benchmark-ftsrv.C File Reference . . . . .          | 798 |
| 17.31.1 Macro Definition Documentation . . . . .          | 798 |
| 17.31.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . . | 798 |
| 17.31.2 Function Documentation . . . . .                  | 798 |
| 17.31.2.1 main() . . . . .                                | 799 |
| 17.32 benchmark-ftsrti.C File Reference . . . . .         | 799 |
| 17.32.1 Macro Definition Documentation . . . . .          | 799 |
| 17.32.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . . | 799 |
| 17.32.1.2 CUBE . . . . .                                  | 799 |
| 17.32.2 Function Documentation . . . . .                  | 799 |
| 17.32.2.1 main() . . . . .                                | 799 |
| 17.33 benchmark-inverse.C File Reference . . . . .        | 799 |
| 17.33.1 Macro Definition Documentation . . . . .          | 800 |
| 17.33.1.1 CUBE . . . . .                                  | 800 |
| 17.33.2 Function Documentation . . . . .                  | 800 |
| 17.33.2.1 main() . . . . .                                | 800 |
| 17.34 benchmark-lqmp-mp.C File Reference . . . . .        | 800 |
| 17.34.1 Function Documentation . . . . .                  | 800 |
| 17.34.1.1 main() . . . . .                                | 800 |
| 17.35 benchmark-lqmp.C File Reference . . . . .           | 801 |
| 17.35.1 Macro Definition Documentation . . . . .          | 801 |
| 17.35.1.1 CUBE . . . . .                                  | 801 |
| 17.35.2 Function Documentation . . . . .                  | 801 |
| 17.35.2.1 main() . . . . .                                | 801 |
| 17.36 benchmark-pluq.C File Reference . . . . .           | 801 |
| 17.36.1 Macro Definition Documentation . . . . .          | 802 |
| 17.36.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET . . . . . | 802 |
| 17.36.1.2 CUBE . . . . .                                  | 802 |
| 17.36.2 Typedef Documentation . . . . .                   | 802 |

|  |     |
|--|-----|
| 17.36.2.1 Field                        | 802 |
| 17.36.3 Function Documentation         | 802 |
| 17.36.3.1 verification_PLUQ()          | 802 |
| 17.36.3.2 Rec_Initialize()             | 802 |
| 17.36.3.3 main()                       | 803 |
| 17.37 benchmark-wino.C File Reference  | 803 |
| 17.37.1 Macro Definition Documentation | 803 |
| 17.37.1.1 CUBE                         | 803 |
| 17.37.2 Function Documentation         | 803 |
| 17.37.2.1 launch_wino()                | 803 |
| 17.37.2.2 main()                       | 803 |
| 17.38 config.h File Reference          | 804 |
| 17.38.1 Macro Definition Documentation | 804 |
| 17.38.1.1 HAVE_BLAS                    | 804 |
| 17.38.1.2 HAVE_CBLAS                   | 804 |
| 17.38.1.3 HAVE_CXX11                   | 805 |
| 17.38.1.4 HAVE_DLFCN_H                 | 805 |
| 17.38.1.5 HAVE_FLOAT_H                 | 805 |
| 17.38.1.6 HAVE_INT128                  | 805 |
| 17.38.1.7 HAVE_INTPTR_T                | 805 |
| 17.38.1.8 HAVE_LAPACK                  | 805 |
| 17.38.1.9 HAVE_LIMITS_H                | 805 |
| 17.38.1.10 HAVE_LITTLE_ENDIAN          | 805 |
| 17.38.1.11 HAVE_PTHREAD_H              | 805 |
| 17.38.1.12 HAVE_STDDEF_H               | 805 |
| 17.38.1.13 HAVE_STDINT_H               | 805 |
| 17.38.1.14 HAVE_STDIO_H                | 805 |
| 17.38.1.15 HAVE_STDLIB_H               | 806 |
| 17.38.1.16 HAVE_STRINGS_H              | 806 |
| 17.38.1.17 HAVE_STRING_H               | 806 |
| 17.38.1.18 HAVE_SYS_STAT_H             | 806 |
| 17.38.1.19 HAVE_SYS_TIME_H             | 806 |
| 17.38.1.20 HAVE_SYS_TYPES_H            | 806 |
| 17.38.1.21 HAVE_UNISTD_H               | 806 |
| 17.38.1.22 LT_OBJDIR                   | 806 |
| 17.38.1.23 OPENBLAS_NUM_THREADS        | 806 |
| 17.38.1.24 PACKAGE                     | 806 |
| 17.38.1.25 PACKAGE_BUGREPORT           | 806 |
| 17.38.1.26 PACKAGE_NAME                | 806 |
| 17.38.1.27 PACKAGE_STRING              | 807 |
| 17.38.1.28 PACKAGE_TARNAME             | 807 |
| 17.38.1.29 PACKAGE_URL                 | 807 |

|   |     |
|---|-----|
| 17.38.1.30 PACKAGE_VERSION . . . . .                    | 807 |
| 17.38.1.31 SIZEOF_CHAR . . . . .                        | 807 |
| 17.38.1.32 SIZEOF_INT . . . . .                         | 807 |
| 17.38.1.33 SIZEOF_LONG . . . . .                        | 807 |
| 17.38.1.34 SIZEOF_LONG_LONG . . . . .                   | 807 |
| 17.38.1.35 SIZEOF_SHORT . . . . .                       | 807 |
| 17.38.1.36 SIZEOF___INT64 . . . . .                     | 807 |
| 17.38.1.37 STDC_HEADERS . . . . .                       | 807 |
| 17.38.1.38 USE_OPENMP . . . . .                         | 807 |
| 17.38.1.39 VERSION . . . . .                            | 808 |
| 17.39 config.h File Reference . . . . .                 | 808 |
| 17.39.1 Macro Definition Documentation . . . . .        | 808 |
| 17.39.1.1 __FFLASFFPACK_HAVE_BLAS . . . . .             | 808 |
| 17.39.1.2 __FFLASFFPACK_HAVE_CBLAS . . . . .            | 809 |
| 17.39.1.3 __FFLASFFPACK_HAVE_CXX11 . . . . .            | 809 |
| 17.39.1.4 __FFLASFFPACK_HAVE_DLFCN_H . . . . .          | 809 |
| 17.39.1.5 __FFLASFFPACK_HAVE_FLOAT_H . . . . .          | 809 |
| 17.39.1.6 __FFLASFFPACK_HAVE_INT128 . . . . .           | 809 |
| 17.39.1.7 __FFLASFFPACK_HAVE_INTPATH_T_H . . . . .      | 809 |
| 17.39.1.8 __FFLASFFPACK_HAVE_LAPACK . . . . .           | 809 |
| 17.39.1.9 __FFLASFFPACK_HAVE_LIMITS_H . . . . .         | 809 |
| 17.39.1.10 __FFLASFFPACK_HAVE_LITTLE_ENDIAN . . . . .   | 809 |
| 17.39.1.11 __FFLASFFPACK_HAVE_PTHREAD_H . . . . .       | 809 |
| 17.39.1.12 __FFLASFFPACK_HAVE_STDDEF_H . . . . .        | 809 |
| 17.39.1.13 __FFLASFFPACK_HAVE_STDINT_H . . . . .        | 809 |
| 17.39.1.14 __FFLASFFPACK_HAVE_STDIO_H . . . . .         | 810 |
| 17.39.1.15 __FFLASFFPACK_HAVE_STDLIB_H . . . . .        | 810 |
| 17.39.1.16 __FFLASFFPACK_HAVE_STRINGS_H . . . . .       | 810 |
| 17.39.1.17 __FFLASFFPACK_HAVE_STRING_H . . . . .        | 810 |
| 17.39.1.18 __FFLASFFPACK_HAVE_SYS_STAT_H . . . . .      | 810 |
| 17.39.1.19 __FFLASFFPACK_HAVE_SYS_TIME_H . . . . .      | 810 |
| 17.39.1.20 __FFLASFFPACK_HAVE_SYS_TYPES_H . . . . .     | 810 |
| 17.39.1.21 __FFLASFFPACK_HAVE_UNISTD_H . . . . .        | 810 |
| 17.39.1.22 __FFLASFFPACK_LT_OBJDIR . . . . .            | 810 |
| 17.39.1.23 __FFLASFFPACK_OPENBLAS_NUM_THREADS . . . . . | 810 |
| 17.39.1.24 __FFLASFFPACK_PACKAGE . . . . .              | 810 |
| 17.39.1.25 __FFLASFFPACK_PACKAGE_BUGREPORT . . . . .    | 810 |
| 17.39.1.26 __FFLASFFPACK_PACKAGE_NAME . . . . .         | 811 |
| 17.39.1.27 __FFLASFFPACK_PACKAGE_STRING . . . . .       | 811 |
| 17.39.1.28 __FFLASFFPACK_PACKAGE_TARNAME . . . . .      | 811 |
| 17.39.1.29 __FFLASFFPACK_PACKAGE_URL . . . . .          | 811 |
| 17.39.1.30 __FFLASFFPACK_PACKAGE_VERSION . . . . .      | 811 |

|            |   |     |
|------------|---|-----|
| 17.39.1.31 | <a href="#">__FFLASFFPACK_SIZEOF_CHAR</a>           | 811 |
| 17.39.1.32 | <a href="#">__FFLASFFPACK_SIZEOF_INT</a>            | 811 |
| 17.39.1.33 | <a href="#">__FFLASFFPACK_SIZEOF_LONG</a>           | 811 |
| 17.39.1.34 | <a href="#">__FFLASFFPACK_SIZEOF_LONG_LONG</a>      | 811 |
| 17.39.1.35 | <a href="#">__FFLASFFPACK_SIZEOF_SHORT</a>          | 811 |
| 17.39.1.36 | <a href="#">__FFLASFFPACK_SIZEOF__INT64</a>         | 811 |
| 17.39.1.37 | <a href="#">__FFLASFFPACK_STDC_HEADERS</a>          | 811 |
| 17.39.1.38 | <a href="#">__FFLASFFPACK_USE_OPENMP</a>            | 812 |
| 17.39.1.39 | <a href="#">__FFLASFFPACK_VERSION</a>               | 812 |
| 17.40      | <a href="#">mainpage.doxy File Reference</a>        | 812 |
| 17.41      | <a href="#">det.C File Reference</a>                | 812 |
| 17.41.1    | <a href="#">Function Documentation</a>              | 812 |
| 17.41.1.1  | <a href="#">main()</a>                              | 812 |
| 17.42      | <a href="#">matmul.C File Reference</a>             | 812 |
| 17.42.1    | <a href="#">Function Documentation</a>              | 812 |
| 17.42.1.1  | <a href="#">main()</a>                              | 813 |
| 17.43      | <a href="#">rank.C File Reference</a>               | 813 |
| 17.43.1    | <a href="#">Function Documentation</a>              | 813 |
| 17.43.1.1  | <a href="#">main()</a>                              | 813 |
| 17.44      | <a href="#">solve.C File Reference</a>              | 813 |
| 17.44.1    | <a href="#">Function Documentation</a>              | 813 |
| 17.44.1.1  | <a href="#">main()</a>                              | 813 |
| 17.45      | <a href="#">checker_charpoly.inl File Reference</a> | 813 |
| 17.45.1    | <a href="#">Macro Definition Documentation</a>      | 814 |
| 17.45.1.1  | <a href="#">__FFLASFFPACK_checker_charpoly_INL</a>  | 814 |
| 17.46      | <a href="#">checker_det.inl File Reference</a>      | 814 |
| 17.46.1    | <a href="#">Macro Definition Documentation</a>      | 814 |
| 17.46.1.1  | <a href="#">__FFLASFFPACK_checker_det_INL</a>       | 814 |
| 17.47      | <a href="#">checker_empty.h File Reference</a>      | 814 |
| 17.48      | <a href="#">checker_fgemm.inl File Reference</a>    | 815 |
| 17.48.1    | <a href="#">Macro Definition Documentation</a>      | 815 |
| 17.48.1.1  | <a href="#">__FFLASFFPACK_checker_fgemm_INL</a>     | 815 |
| 17.49      | <a href="#">checker_ftrsm.inl File Reference</a>    | 815 |
| 17.49.1    | <a href="#">Macro Definition Documentation</a>      | 815 |
| 17.49.1.1  | <a href="#">__FFLASFFPACK_checker_ftrsm_INL</a>     | 815 |
| 17.50      | <a href="#">checker_invert.inl File Reference</a>   | 815 |
| 17.50.1    | <a href="#">Macro Definition Documentation</a>      | 816 |
| 17.50.1.1  | <a href="#">__FFLASFFPACK_checker_invert_INL</a>    | 816 |
| 17.51      | <a href="#">checker_pluq.inl File Reference</a>     | 816 |
| 17.51.1    | <a href="#">Macro Definition Documentation</a>      | 816 |
| 17.51.1.1  | <a href="#">__FFLASFFPACK_checker_pluq_INL</a>      | 816 |
| 17.52      | <a href="#">checkers.doxy File Reference</a>        | 816 |

|   |     |
|---|-----|
| 17.53 checkers_fflas.h File Reference . . . . .       | 816 |
| 17.54 checkers_fflas.inl File Reference . . . . .     | 817 |
| 17.54.1 Macro Definition Documentation . . . . .      | 817 |
| 17.54.1.1 FFLASFFPACK_checkers_fflas_inl_H . . . . .  | 817 |
| 17.55 checkers_ffpack.h File Reference . . . . .      | 817 |
| 17.56 checkers_ffpack.inl File Reference . . . . .    | 818 |
| 17.56.1 Macro Definition Documentation . . . . .      | 818 |
| 17.56.1.1 FFLASFFPACK_checkers_ffpack_inl_H . . . . . | 818 |
| 17.57 config-blas.h File Reference . . . . .          | 819 |
| 17.57.1 Macro Definition Documentation . . . . .      | 819 |
| 17.57.1.1 CBLAS_INT . . . . .                         | 820 |
| 17.57.1.2 CBLAS_ENUM_DEFINED_H . . . . .              | 820 |
| 17.57.1.3 CBLAS_EXTERNALS . . . . .                   | 820 |
| 17.57.1.4 blas_enum . . . . .                         | 820 |
| 17.57.2 Enumeration Type Documentation . . . . .      | 820 |
| 17.57.2.1 CBLAS_ORDER . . . . .                       | 820 |
| 17.57.2.2 CBLAS_TRANSPOSE . . . . .                   | 820 |
| 17.57.2.3 CBLAS_UPLO . . . . .                        | 820 |
| 17.57.2.4 CBLAS_DIAG . . . . .                        | 821 |
| 17.57.2.5 CBLAS_SIDE . . . . .                        | 821 |
| 17.57.3 Function Documentation . . . . .              | 821 |
| 17.57.3.1 daxpy_() . . . . .                          | 821 |
| 17.57.3.2 saxpy_() . . . . .                          | 821 |
| 17.57.3.3 ddot_() . . . . .                           | 821 |
| 17.57.3.4 sdot_() . . . . .                           | 822 |
| 17.57.3.5 dasum_() . . . . .                          | 822 |
| 17.57.3.6 idamax_() . . . . .                         | 822 |
| 17.57.3.7 dnrm2_() . . . . .                          | 822 |
| 17.57.3.8 dgemv_() . . . . .                          | 822 |
| 17.57.3.9 sgemv_() . . . . .                          | 822 |
| 17.57.3.10 dger_() . . . . .                          | 823 |
| 17.57.3.11 sger_() . . . . .                          | 823 |
| 17.57.3.12 dcopy_() . . . . .                         | 823 |
| 17.57.3.13 scopy_() . . . . .                         | 823 |
| 17.57.3.14 dscal_() . . . . .                         | 823 |
| 17.57.3.15 sscal_() . . . . .                         | 824 |
| 17.57.3.16 dtrsm_() . . . . .                         | 824 |
| 17.57.3.17 strsm_() . . . . .                         | 824 |
| 17.57.3.18 dtrmm_() . . . . .                         | 824 |
| 17.57.3.19 strmm_() . . . . .                         | 825 |
| 17.57.3.20 sgemm_() . . . . .                         | 825 |
| 17.57.3.21 dgemm_() . . . . .                         | 825 |

|   |     |
|---|-----|
| 17.58 fflas-ffpack-config.h File Reference . . . . .                      | 825 |
| 17.58.1 Detailed Description . . . . .                                    | 826 |
| 17.58.2 Macro Definition Documentation . . . . .                          | 826 |
| 17.58.2.1 GCC_VERSION . . . . .   | 826 |
| 17.59 fflas-ffpack-default-thresholds.h File Reference . . . . .          | 826 |
| 17.59.1 Macro Definition Documentation . . . . .                          | 826 |
| 17.59.1.1 __FFLASFFPACK_WINOTHRESHOLD . . . . .                           | 826 |
| 17.59.1.2 __FFLASFFPACK_WINOTHRESHOLD_FLT . . . . .                       | 826 |
| 17.59.1.3 __FFLASFFPACK_WINOTHRESHOLD_BAL . . . . .                       | 827 |
| 17.59.1.4 __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT . . . . .                   | 827 |
| 17.59.1.5 __FFLASFFPACK_PLUQ_THRESHOLD . . . . .                          | 827 |
| 17.59.1.6 __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD . . . . .   | 827 |
| 17.59.1.7 __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD . . . . . | 827 |
| 17.59.1.8 __FFLASFFPACK_ARITHPROG_THRESHOLD . . . . .                     | 827 |
| 17.59.1.9 __FFLASFFPACK_FTRTRI_THRESHOLD . . . . .                        | 827 |
| 17.59.1.10 __FFLASFFPACK_FSYTRF_THRESHOLD . . . . .                       | 827 |
| 17.59.1.11 __FFLASFFPACK_FSYRK_THRESHOLD . . . . .                        | 827 |
| 17.60 fflas-ffpack-thresholds.h File Reference . . . . .                  | 827 |
| 17.61 fflas-ffpack.dox File Reference . . . . .                           | 827 |
| 17.62 fflas-ffpack.h File Reference . . . . .                             | 827 |
| 17.62.1 Detailed Description . . . . .                                    | 827 |
| 17.63 fflas.dox File Reference . . . . .                                  | 828 |
| 17.64 fflas.h File Reference . . . . .                                    | 828 |
| 17.64.1 Detailed Description . . . . .                                    | 829 |
| 17.64.2 Macro Definition Documentation . . . . .                          | 829 |
| 17.64.2.1 WINOTHRESHOLD . . . . .   | 829 |
| 17.64.2.2 DOUBLE_TO_FLOAT_CROSSOVER . . . . .                             | 829 |
| 17.65 fflas_bounds.inl File Reference . . . . .                           | 829 |
| 17.65.1 Macro Definition Documentation . . . . .                          | 830 |
| 17.65.1.1 __FFLASFFPACK_fflas_bounds_INL . . . . .                        | 830 |
| 17.65.1.2 FFLAS_INT_TYPE . . . . .  | 830 |
| 17.66 fflas_enum.h File Reference . . . . .                               | 830 |
| 17.67 fflas_fadd.h File Reference . . . . .                               | 831 |
| 17.68 fflas_fadd.inl File Reference . . . . .                             | 832 |
| 17.68.1 Macro Definition Documentation . . . . .                          | 833 |
| 17.68.1.1 __FFLASFFPACK_fadd_INL . . . . .                                | 833 |
| 17.69 fflas_fassign.h File Reference . . . . .                            | 833 |
| 17.70 fflas_fassign.inl File Reference . . . . .                          | 833 |
| 17.70.1 Macro Definition Documentation . . . . .                          | 834 |
| 17.70.1.1 __FFLASFFPACK_fassign_INL . . . . .                             | 834 |
| 17.71 fflas_faxpy.inl File Reference . . . . .                            | 834 |
| 17.71.1 Macro Definition Documentation . . . . .                          | 834 |



|   |     |
|---|-----|
| 17.71.1.1 __FflasFFPACK_faxpy_INL . . . . .                       | 835 |
| 17.72 fflas_fdot.inl File Reference . . . . .                     | 835 |
| 17.72.1 Macro Definition Documentation . . . . .                  | 835 |
| 17.72.1.1 __FflasFFPACK_fdot_INL . . . . .                        | 835 |
| 17.73 fflas_fgemm.inl File Reference . . . . .                    | 836 |
| 17.73.1 Macro Definition Documentation . . . . .                  | 838 |
| 17.73.1.1 __FflasFFPACK_fgemm_INL . . . . .                       | 838 |
| 17.74 fgemm_classical.inl File Reference . . . . .                | 838 |
| 17.74.1 Macro Definition Documentation . . . . .                  | 838 |
| 17.74.1.1 __FflasFFPACK_fflas_fflas_fgemm_classical_INL . . . . . | 838 |
| 17.75 fgemm_classical_mp.inl File Reference . . . . .             | 838 |
| 17.75.1 Detailed Description . . . . .                            | 840 |
| 17.75.2 Macro Definition Documentation . . . . .                  | 840 |
| 17.75.2.1 __FFPACK_fgemm_classical_INL . . . . .                  | 840 |
| 17.76 fgemm_winograd.inl File Reference . . . . .                 | 840 |
| 17.76.1 Macro Definition Documentation . . . . .                  | 841 |
| 17.76.1.1 __FflasFFPACK_fflas_fflas_fgemm_winograd_INL . . . . .  | 841 |
| 17.76.1.2 NEWWINO . . . . .                                       | 841 |
| 17.77 matmul.doxy File Reference . . . . .                        | 841 |
| 17.78 schedule_bini.inl File Reference . . . . .                  | 842 |
| 17.78.1 Detailed Description . . . . .                            | 842 |
| 17.78.2 Macro Definition Documentation . . . . .                  | 842 |
| 17.78.2.1 __FflasFFPACK_fgemm_bini_INL . . . . .                  | 842 |
| 17.79 schedule_winograd.inl File Reference . . . . .              | 842 |
| 17.79.1 Macro Definition Documentation . . . . .                  | 843 |
| 17.79.1.1 __FflasFFPACK_fgemm_winograd_INL . . . . .              | 843 |
| 17.80 schedule_winograd_acc.inl File Reference . . . . .          | 843 |
| 17.80.1 Macro Definition Documentation . . . . .                  | 843 |
| 17.80.1.1 __FflasFFPACK_fgemm_winograd_acc_INL . . . . .          | 843 |
| 17.81 schedule_winograd_acc_ip.inl File Reference . . . . .       | 844 |
| 17.81.1 Macro Definition Documentation . . . . .                  | 844 |
| 17.81.1.1 __FflasFFPACK_fgemm_winograd_acc_ip_INL . . . . .       | 844 |
| 17.82 schedule_winograd_ip.inl File Reference . . . . .           | 844 |
| 17.82.1 Macro Definition Documentation . . . . .                  | 845 |
| 17.82.1.1 __FflasFFPACK_fgemm_winograd_ip_INL . . . . .           | 845 |
| 17.83 fflas_fgmv.inl File Reference . . . . .                     | 845 |
| 17.83.1 Macro Definition Documentation . . . . .                  | 846 |
| 17.83.1.1 __FflasFFPACK_fgmv_INL . . . . .                        | 846 |
| 17.84 fflas_fgmv_mp.inl File Reference . . . . .                  | 847 |
| 17.84.1 Macro Definition Documentation . . . . .                  | 847 |
| 17.84.1.1 __FflasFFPACK_fgmv_mp_INL . . . . .                     | 847 |
| 17.85 fflas_fger.inl File Reference . . . . .                     | 847 |

|  |     |
|--|-----|
| 17.85.1 Macro Definition Documentation       | 848 |
| 17.85.1.1 __FFLASFFPACK_fger_INL             | 848 |
| 17.86 fflas_fger_mp.inl File Reference       | 849 |
| 17.86.1 Macro Definition Documentation       | 849 |
| 17.86.1.1 __FFPACK_fger_mp_INL               | 849 |
| 17.87 fflas_freduce.h File Reference         | 849 |
| 17.88 fflas_freduce.inl File Reference       | 850 |
| 17.88.1 Macro Definition Documentation       | 852 |
| 17.88.1.1 __FFLASFFPACK_fflas_freduce_INL    | 852 |
| 17.88.1.2 FFLASFFPACK_COPY_REDUCE            | 852 |
| 17.89 fflas_freduce_mp.inl File Reference    | 852 |
| 17.89.1 Macro Definition Documentation       | 852 |
| 17.89.1.1 __FFLASFFPACK_fflas_freduce_mp_INL | 852 |
| 17.90 fflas_freivalds.inl File Reference     | 852 |
| 17.90.1 Macro Definition Documentation       | 853 |
| 17.90.1.1 __FFLASFFPACK_freivalds_INL        | 853 |
| 17.91 fflas_fscal.h File Reference           | 853 |
| 17.92 fflas_fscal.inl File Reference         | 853 |
| 17.92.1 Macro Definition Documentation       | 854 |
| 17.92.1.1 __FFLASFFPACK_fscal_INL            | 854 |
| 17.93 fflas_fscal_mp.inl File Reference      | 855 |
| 17.93.1 Macro Definition Documentation       | 855 |
| 17.93.1.1 __FFLASFFPACK_fscal_mp_INL         | 855 |
| 17.94 fflas_fsyr2k.inl File Reference        | 855 |
| 17.94.1 Macro Definition Documentation       | 856 |
| 17.94.1.1 __FFLASFFPACK_fflas_fsyr2k_INL     | 856 |
| 17.95 fflas_fsyrk.inl File Reference         | 856 |
| 17.95.1 Macro Definition Documentation       | 857 |
| 17.95.1.1 __FFLASFFPACK_fflas_fsyrk_INL      | 857 |
| 17.96 fflas_ftrmm.inl File Reference         | 857 |
| 17.96.1 Macro Definition Documentation       | 857 |
| 17.96.1.1 __FFLASFFPACK_ftrmm_INL            | 857 |
| 17.97 fflas_ftrsm.inl File Reference         | 857 |
| 17.97.1 Macro Definition Documentation       | 858 |
| 17.97.1.1 __FFLASFFPACK_ftrsm_INL            | 858 |
| 17.98 fflas_ftrsm_mp.inl File Reference      | 858 |
| 17.98.1 Detailed Description                 | 858 |
| 17.98.2 Macro Definition Documentation       | 859 |
| 17.98.2.1 __FFPACK_ftrsm_mp_INL              | 859 |
| 17.99 fflas_ftrsv.inl File Reference         | 859 |
| 17.99.1 Macro Definition Documentation       | 859 |
| 17.99.1.1 __FFLASFFPACK_ftrsv_INL            | 859 |

|  |     |
|--|-----|
| 17.100 fflas_helpers.inl File Reference . . . . .                | 859 |
| 17.100.1 Macro Definition Documentation . . . . .                | 860 |
| 17.100.1.1 __FFLASFFPACK_fflas_fflas_mmhelper_INL . . . . .      | 860 |
| 17.101 igemm.dox File Reference . . . . .                        | 860 |
| 17.102 igemm.h File Reference . . . . .                          | 860 |
| 17.103 igemm.inl File Reference . . . . .                        | 861 |
| 17.103.1 Macro Definition Documentation . . . . .                | 861 |
| 17.103.1.1 __FFLASFFPACK_fflas_igemm_igemm_INL . . . . .         | 862 |
| 17.104 igemm_kernels.h File Reference . . . . .                  | 862 |
| 17.105 igemm_kernels.inl File Reference . . . . .                | 862 |
| 17.105.1 Macro Definition Documentation . . . . .                | 863 |
| 17.105.1.1 __FFLASFFPACK_fflas_igemm_igemm_kernels_INL . . . . . | 863 |
| 17.106 igemm_tools.h File Reference . . . . .                    | 863 |
| 17.107 igemm_tools.inl File Reference . . . . .                  | 863 |
| 17.107.1 Macro Definition Documentation . . . . .                | 864 |
| 17.107.1.1 __FFLASFFPACK_fflas_igemm_igemm_tools_INL . . . . .   | 864 |
| 17.108 fflas_level1.inl File Reference . . . . .                 | 864 |
| 17.108.1 Macro Definition Documentation . . . . .                | 866 |
| 17.108.1.1 __FFLASFFPACK_fflas_fflas_level1_INL . . . . .        | 866 |
| 17.109 fflas_level2.inl File Reference . . . . .                 | 866 |
| 17.109.1 Macro Definition Documentation . . . . .                | 869 |
| 17.109.1.1 __FFLASFFPACK_fflas_fflas_level2_INL . . . . .        | 869 |
| 17.110 fflas_level3.inl File Reference . . . . .                 | 869 |
| 17.110.1 Macro Definition Documentation . . . . .                | 871 |
| 17.110.1.1 __FFLASFFPACK_fflas_fflas_level3_INL . . . . .        | 871 |
| 17.110.1.2 __FFLAS__TRSM_READONLY . . . . .                      | 871 |
| 17.111 fflas_pfgemm.inl File Reference . . . . .                 | 871 |
| 17.111.1 Macro Definition Documentation . . . . .                | 872 |
| 17.111.1.1 __FFLASFFPACK_fflas_pfgemm_INL . . . . .              | 872 |
| 17.111.1.2 __FFLASFFPACK_SEQPARTHRESHOLD . . . . .               | 872 |
| 17.111.1.3 __FFLASFFPACK_DIMKPENALTY . . . . .                   | 872 |
| 17.112 fflas_pftrsm.inl File Reference . . . . .                 | 872 |
| 17.112.1 Macro Definition Documentation . . . . .                | 873 |
| 17.112.1.1 __FFLASFFPACK_fflas_pftrsm_INL . . . . .              | 873 |
| 17.112.1.2 PTRSM_HYBRID_THRESHOLD . . . . .                      | 873 |
| 17.113 fflas_simd.h File Reference . . . . .                     | 873 |
| 17.113.1 Macro Definition Documentation . . . . .                | 874 |
| 17.113.1.1 SIMD_INT . . . . .                                    | 874 |
| 17.113.1.2 INLINE . . . . .                                      | 874 |
| 17.113.1.3 CONST . . . . .                                       | 874 |
| 17.113.1.4 PURE . . . . .  | 874 |
| 17.113.1.5 NORML_MOD . . . . .                                   | 874 |

|  |     |
|--|-----|
| 17.113.1.6 <a href="#">FLOAT_MOD</a>   | 874 |
| 17.113.2 <a href="#">Typedef Documentation</a>                                 | 875 |
| 17.113.2.1 <a href="#">Simd</a>  | 875 |
| 17.114 <a href="#">simd.doxy File Reference</a>                                | 875 |
| 17.115 <a href="#">simd128.inl File Reference</a>                              | 875 |
| 17.115.1 <a href="#">Macro Definition Documentation</a>                        | 875 |
| 17.115.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_INL</a>        | 875 |
| 17.115.2 <a href="#">Typedef Documentation</a>                                 | 875 |
| 17.115.2.1 <a href="#">Simd128</a>   | 875 |
| 17.116 <a href="#">simd128_double.inl File Reference</a>                       | 875 |
| 17.116.1 <a href="#">Macro Definition Documentation</a>                        | 876 |
| 17.116.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL</a> | 876 |
| 17.117 <a href="#">simd128_float.inl File Reference</a>                        | 876 |
| 17.117.1 <a href="#">Macro Definition Documentation</a>                        | 876 |
| 17.117.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL</a>  | 876 |
| 17.118 <a href="#">simd128_int16.inl File Reference</a>                        | 876 |
| 17.118.1 <a href="#">Macro Definition Documentation</a>                        | 876 |
| 17.118.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL</a>  | 876 |
| 17.119 <a href="#">simd128_int32.inl File Reference</a>                        | 876 |
| 17.119.1 <a href="#">Macro Definition Documentation</a>                        | 877 |
| 17.119.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL</a>  | 877 |
| 17.120 <a href="#">simd128_int64.inl File Reference</a>                        | 877 |
| 17.120.1 <a href="#">Macro Definition Documentation</a>                        | 877 |
| 17.120.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL</a>  | 877 |
| 17.120.1.2 <a href="#">vect_t</a>  | 877 |
| 17.121 <a href="#">simd256.inl File Reference</a>                              | 877 |
| 17.121.1 <a href="#">Macro Definition Documentation</a>                        | 878 |
| 17.121.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_INL</a>        | 878 |
| 17.121.2 <a href="#">Typedef Documentation</a>                                 | 878 |
| 17.121.2.1 <a href="#">Simd256</a>   | 878 |
| 17.122 <a href="#">simd256_double.inl File Reference</a>                       | 878 |
| 17.122.1 <a href="#">Macro Definition Documentation</a>                        | 878 |
| 17.122.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL</a> | 878 |
| 17.123 <a href="#">simd256_float.inl File Reference</a>                        | 878 |
| 17.123.1 <a href="#">Macro Definition Documentation</a>                        | 878 |
| 17.123.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL</a>  | 878 |
| 17.124 <a href="#">simd256_int16.inl File Reference</a>                        | 879 |
| 17.124.1 <a href="#">Macro Definition Documentation</a>                        | 879 |
| 17.124.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL</a>  | 879 |
| 17.125 <a href="#">simd256_int32.inl File Reference</a>                        | 879 |
| 17.125.1 <a href="#">Macro Definition Documentation</a>                        | 879 |
| 17.125.1.1 <a href="#">__FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL</a>  | 879 |

|   |     |
|---|-----|
| 17.126 simd256_int64.inl File Reference . . . . .                       | 879 |
| 17.126.1 Macro Definition Documentation . . . . .                       | 880 |
| 17.126.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL . . . . . | 880 |
| 17.126.1.2 vect_t . . . . .   | 880 |
| 17.127 simd512.inl File Reference . . . . .                             | 880 |
| 17.127.1 Macro Definition Documentation . . . . .                       | 880 |
| 17.127.1.1 __FFLASFFPACK_simd512_INL . . . . .                          | 880 |
| 17.127.2 Typedef Documentation . . . . .                                | 880 |
| 17.127.2.1 Simd512 . . . . .  | 880 |
| 17.128 simd512_double.inl File Reference . . . . .                      | 880 |
| 17.128.1 Macro Definition Documentation . . . . .                       | 881 |
| 17.128.1.1 __FFLASFFPACK_simd512_double_INL . . . . .                   | 881 |
| 17.129 simd512_float.inl File Reference . . . . .                       | 881 |
| 17.129.1 Macro Definition Documentation . . . . .                       | 881 |
| 17.129.1.1 __FFLASFFPACK_simd512_float_INL . . . . .                    | 881 |
| 17.130 simd512_int32.inl File Reference . . . . .                       | 881 |
| 17.130.1 Macro Definition Documentation . . . . .                       | 881 |
| 17.130.1.1 __FFLASFFPACK_simd512_int32_INL . . . . .                    | 881 |
| 17.131 simd512_int64.inl File Reference . . . . .                       | 882 |
| 17.131.1 Macro Definition Documentation . . . . .                       | 882 |
| 17.131.1.1 _simd512_int64_INL . . . . .                                 | 882 |
| 17.131.1.2 vect_t . . . . .   | 882 |
| 17.132 simd_modular.inl File Reference . . . . .                        | 882 |
| 17.133 flas_sparse.h File Reference . . . . .                           | 882 |
| 17.133.1 Macro Definition Documentation . . . . .                       | 886 |
| 17.133.1.1 index_t . . . . .  | 886 |
| 17.133.1.2 ROUND_DOWN . . . . .   | 886 |
| 17.133.1.3 __FFLASFFPACK_CACHE_LINE_SIZE . . . . .                      | 886 |
| 17.133.1.4 assume_aligned . . . . .                                     | 886 |
| 17.133.1.5 DENSE_THRESHOLD . . . . .                                    | 887 |
| 17.134 flas_sparse.inl File Reference . . . . .                         | 887 |
| 17.134.1 Macro Definition Documentation . . . . .                       | 889 |
| 17.134.1.1 __FFLASFFPACK_fflas_fflas_sparse_INL . . . . .               | 889 |
| 17.135 coo.h File Reference . . . . .                                   | 889 |
| 17.136 coo_spmv.inl File Reference . . . . .                            | 889 |
| 17.136.1 Macro Definition Documentation . . . . .                       | 890 |
| 17.136.1.1 __FFLASFFPACK_fflas_sparse_coo_spmv_INL . . . . .            | 890 |
| 17.137 coo_spmv.inl File Reference . . . . .                            | 890 |
| 17.137.1 Macro Definition Documentation . . . . .                       | 891 |
| 17.137.1.1 __FFLASFFPACK_fflas_sparse_coo_spmv_INL . . . . .            | 891 |
| 17.138 coo_utils.inl File Reference . . . . .                           | 891 |
| 17.138.1 Macro Definition Documentation . . . . .                       | 891 |

|   |     |
|---|-----|
| 17.138.1.1 __FFLASFFPACK_fflas_sparse_coo_utils_INL . . . . .     | 892 |
| 17.139 csr.h File Reference . . . . .                             | 892 |
| 17.140 csr_pspmm.inl File Reference . . . . .                     | 892 |
| 17.140.1 Macro Definition Documentation . . . . .                 | 893 |
| 17.140.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL . . . . .     | 893 |
| 17.141 csr_pspmv.inl File Reference . . . . .                     | 893 |
| 17.141.1 Macro Definition Documentation . . . . .                 | 894 |
| 17.141.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL . . . . .     | 894 |
| 17.142 csr_spm্ম.inl File Reference . . . . .                     | 894 |
| 17.142.1 Macro Definition Documentation . . . . .                 | 895 |
| 17.142.1.1 __FFLASFFPACK_fflas_sparse_CSR_spm্ম_INL . . . . .     | 895 |
| 17.143 csr_spmv.inl File Reference . . . . .                      | 895 |
| 17.143.1 Macro Definition Documentation . . . . .                 | 896 |
| 17.143.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmv_INL . . . . .      | 896 |
| 17.144 csr_utils.inl File Reference . . . . .                     | 896 |
| 17.145 csr_hyb.h File Reference . . . . .                         | 896 |
| 17.146 csr_hyb_pspmm.inl File Reference . . . . .                 | 897 |
| 17.146.1 Macro Definition Documentation . . . . .                 | 897 |
| 17.146.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL . . . . . | 897 |
| 17.147 csr_hyb_pspmv.inl File Reference . . . . .                 | 898 |
| 17.147.1 Macro Definition Documentation . . . . .                 | 898 |
| 17.147.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL . . . . . | 898 |
| 17.148 csr_hyb_spm্ম.inl File Reference . . . . .                 | 898 |
| 17.148.1 Macro Definition Documentation . . . . .                 | 899 |
| 17.148.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spm্ম_INL . . . . . | 899 |
| 17.149 csr_hyb_spmv.inl File Reference . . . . .                  | 899 |
| 17.149.1 Macro Definition Documentation . . . . .                 | 899 |
| 17.149.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL . . . . .  | 899 |
| 17.150 csr_hyb_utils.inl File Reference . . . . .                 | 899 |
| 17.150.1 Macro Definition Documentation . . . . .                 | 900 |
| 17.150.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL . . . . . | 900 |
| 17.151 ell.h File Reference . . . . .                             | 900 |
| 17.152 ell_pspmm.inl File Reference . . . . .                     | 900 |
| 17.152.1 Macro Definition Documentation . . . . .                 | 901 |
| 17.152.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL . . . . .     | 901 |
| 17.153 ell_pspmv.inl File Reference . . . . .                     | 901 |
| 17.153.1 Macro Definition Documentation . . . . .                 | 902 |
| 17.153.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL . . . . .     | 902 |
| 17.154 ell_spm্ম.inl File Reference . . . . .                     | 902 |
| 17.154.1 Macro Definition Documentation . . . . .                 | 903 |
| 17.154.1.1 __FFLASFFPACK_fflas_sparse_ELL_spm্ম_INL . . . . .     | 903 |
| 17.155 ell_spmv.inl File Reference . . . . .                      | 903 |

|  |     |
|--|-----|
| 17.155.1 Macro Definition Documentation                  | 904 |
| 17.155.1.1 __FFLASFFPACK_fflas_sparse_ELL_spmv_INL       | 904 |
| 17.156 ell_utils.inl File Reference                      | 904 |
| 17.156.1 Macro Definition Documentation                  | 904 |
| 17.156.1.1 __FFLASFFPACK_fflas_sparse_ELL_utils_INL      | 904 |
| 17.157 ell_simd.h File Reference                         | 904 |
| 17.158 ell_simd_pspmv.inl File Reference                 | 905 |
| 17.158.1 Macro Definition Documentation                  | 906 |
| 17.158.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL | 906 |
| 17.159 ell_simd_spmv.inl File Reference                  | 906 |
| 17.159.1 Macro Definition Documentation                  | 906 |
| 17.159.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL  | 907 |
| 17.160 ell_simd_utils.inl File Reference                 | 907 |
| 17.160.1 Macro Definition Documentation                  | 907 |
| 17.160.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL | 907 |
| 17.161 hyb_zo.h File Reference                           | 907 |
| 17.162 hyb_zo_pspmm.inl File Reference                   | 907 |
| 17.162.1 Macro Definition Documentation                  | 908 |
| 17.162.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL   | 908 |
| 17.163 hyb_zo_pspmv.inl File Reference                   | 908 |
| 17.163.1 Macro Definition Documentation                  | 908 |
| 17.163.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL   | 908 |
| 17.164 hyb_zo_spm্ম.inl File Reference                   | 909 |
| 17.164.1 Macro Definition Documentation                  | 909 |
| 17.164.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spm্ম_INL   | 909 |
| 17.165 hyb_zo_spmv.inl File Reference                    | 909 |
| 17.165.1 Macro Definition Documentation                  | 909 |
| 17.165.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL    | 910 |
| 17.166 hyb_zo_utils.inl File Reference                   | 910 |
| 17.166.1 Macro Definition Documentation                  | 910 |
| 17.166.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL   | 910 |
| 17.167 read_sparse.h File Reference                      | 910 |
| 17.167.1 Macro Definition Documentation                  | 911 |
| 17.167.1.1 DNS_BIN_VER                                   | 911 |
| 17.167.1.2 mask_t  | 911 |
| 17.168 sell.h File Reference                             | 911 |
| 17.169 sell_pspmv.inl File Reference                     | 912 |
| 17.169.1 Macro Definition Documentation                  | 912 |
| 17.169.1.1 __FFLASFFPACK_fflas_sparse_sell_pspmv_INL     | 912 |
| 17.170 sell_spmv.inl File Reference                      | 912 |
| 17.170.1 Macro Definition Documentation                  | 913 |
| 17.170.1.1 __FFLASFFPACK_fflas_sparse_sell_spmv_INL      | 913 |

|            |  |     |
|------------|--|-----|
| 17.171     | <a href="#">sell_utils.inl File Reference</a>                        | 913 |
| 17.171.1   | <a href="#">Macro Definition Documentation</a>                       | 914 |
| 17.171.1.1 | <a href="#">__FFLASFFPACK_fflas_sparse_sell_utils_INL</a>            | 914 |
| 17.172     | <a href="#">sparse_matrix_traits.h File Reference</a>                | 914 |
| 17.173     | <a href="#">utils.h File Reference</a>                               | 915 |
| 17.174     | <a href="#">ffpack.dox File Reference</a>                            | 916 |
| 17.175     | <a href="#">ffpack.h File Reference</a>                              | 916 |
| 17.175.1   | <a href="#">Detailed Description</a>                                 | 924 |
| 17.175.2   | <a href="#">Macro Definition Documentation</a>                       | 924 |
| 17.175.2.1 | <a href="#">__FFLASFFPACK_FTRSTR_THRESHOLD</a>                       | 924 |
| 17.175.2.2 | <a href="#">__FFLASFFPACK_FTRSSYR2K_THRESHOLD</a>                    | 924 |
| 17.176     | <a href="#">ffpack.inl File Reference</a>                            | 924 |
| 17.176.1   | <a href="#">Macro Definition Documentation</a>                       | 925 |
| 17.176.1.1 | <a href="#">__FFLASFFPACK_ffpack_INL</a>                             | 925 |
| 17.177     | <a href="#">ffpack_charpoly.inl File Reference</a>                   | 925 |
| 17.177.1   | <a href="#">Macro Definition Documentation</a>                       | 926 |
| 17.177.1.1 | <a href="#">__FFLASFFPACK_charpoly_INL</a>                           | 926 |
| 17.178     | <a href="#">ffpack_charpoly_danilevski.inl File Reference</a>        | 926 |
| 17.178.1   | <a href="#">Macro Definition Documentation</a>                       | 926 |
| 17.178.1.1 | <a href="#">__FFLASFFPACK_ffpack_charpoly_danilveski_INL</a>         | 927 |
| 17.179     | <a href="#">ffpack_charpoly_kgfast.inl File Reference</a>            | 927 |
| 17.179.1   | <a href="#">Macro Definition Documentation</a>                       | 927 |
| 17.179.1.1 | <a href="#">__FFLASFFPACK_ffpack_charpoly_kgfast_INL</a>             | 927 |
| 17.180     | <a href="#">ffpack_charpoly_kgfastgeneralized.inl File Reference</a> | 927 |
| 17.180.1   | <a href="#">Macro Definition Documentation</a>                       | 928 |
| 17.180.1.1 | <a href="#">__FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL</a>  | 928 |
| 17.181     | <a href="#">ffpack_charpoly_kglu.inl File Reference</a>              | 928 |
| 17.181.1   | <a href="#">Macro Definition Documentation</a>                       | 928 |
| 17.181.1.1 | <a href="#">__FFLASFFPACK_ffpack_charpoly_kglu_INL</a>               | 928 |
| 17.182     | <a href="#">ffpack_charpoly_mp.inl File Reference</a>                | 928 |
| 17.182.1   | <a href="#">Macro Definition Documentation</a>                       | 929 |
| 17.182.1.1 | <a href="#">__FFPACK_charpoly_mp_INL</a>                             | 929 |
| 17.183     | <a href="#">ffpack_det_mp.inl File Reference</a>                     | 929 |
| 17.183.1   | <a href="#">Macro Definition Documentation</a>                       | 929 |
| 17.183.1.1 | <a href="#">__FFPACK_det_mp_INL</a>                                  | 929 |
| 17.184     | <a href="#">ffpack_echelonforms.inl File Reference</a>               | 930 |
| 17.184.1   | <a href="#">Macro Definition Documentation</a>                       | 931 |
| 17.184.1.1 | <a href="#">__FFLASFFPACK_ffpack_echelon_forms_INL</a>               | 931 |
| 17.184.1.2 | <a href="#">__FFLASFFPACK_GAUSSJORDAN_BASECASE</a>                   | 931 |
| 17.185     | <a href="#">ffpack_fgesv.inl File Reference</a>                      | 931 |
| 17.185.1   | <a href="#">Macro Definition Documentation</a>                       | 931 |
| 17.185.1.1 | <a href="#">__FFLASFFPACK_ffpack_fgesv_INL</a>                       | 931 |



|  |     |
|--|-----|
| 17.186 fpack_fgetrs.inl File Reference . . . . .         | 932 |
| 17.186.1 Macro Definition Documentation . . . . .        | 932 |
| 17.186.1.1 __FFLASFFPACK_fpack_fgetrs_INL . . . . .      | 932 |
| 17.187 fpack_frobenius.inl File Reference . . . . .      | 932 |
| 17.188 fpack_fsytrf.inl File Reference . . . . .         | 933 |
| 17.188.1 Macro Definition Documentation . . . . .        | 934 |
| 17.188.1.1 __FFLASFFPACK_fpack_fsytrf_INL . . . . .      | 934 |
| 17.189 fpack_ftrssyr2k.inl File Reference . . . . .      | 934 |
| 17.189.1 Macro Definition Documentation . . . . .        | 934 |
| 17.189.1.1 __FFLASFFPACK_fpack_ftrssyr2k_INL . . . . .   | 935 |
| 17.190 fpack_ftrstr.inl File Reference . . . . .         | 935 |
| 17.190.1 Macro Definition Documentation . . . . .        | 935 |
| 17.190.1.1 __FFLASFFPACK_fpack_ftrstr_INL . . . . .      | 935 |
| 17.191 fpack_ftrtr.inl File Reference . . . . .          | 935 |
| 17.191.1 Macro Definition Documentation . . . . .        | 936 |
| 17.191.1.1 ENABLE_ALL_CHECKINGS . . . . .                | 936 |
| 17.191.1.2 __FFLASFFPACK_fpack_ftrtr_INL . . . . .       | 936 |
| 17.192 fpack_invert.inl File Reference . . . . .         | 936 |
| 17.192.1 Macro Definition Documentation . . . . .        | 936 |
| 17.192.1.1 __FFLASFFPACK_fpack_invert_INL . . . . .      | 936 |
| 17.193 fpack_krylovelim.inl File Reference . . . . .     | 936 |
| 17.193.1 Macro Definition Documentation . . . . .        | 937 |
| 17.193.1.1 __FFLASFFPACK_fpack_krylovelim_INL . . . . .  | 937 |
| 17.194 fpack_ludivine.inl File Reference . . . . .       | 937 |
| 17.194.1 Macro Definition Documentation . . . . .        | 937 |
| 17.194.1.1 __FFLASFFPACK_fpack_ludivine_INL . . . . .    | 937 |
| 17.195 fpack_ludivine_mp.inl File Reference . . . . .    | 938 |
| 17.195.1 Macro Definition Documentation . . . . .        | 938 |
| 17.195.1.1 __FFPACK_ludivine_mp_INL . . . . .            | 938 |
| 17.196 fpack_minpoly.inl File Reference . . . . .        | 938 |
| 17.196.1 Macro Definition Documentation . . . . .        | 939 |
| 17.196.1.1 __FFLASFFPACK_fpack_minpoly_INL . . . . .     | 939 |
| 17.197 fpack_permutation.inl File Reference . . . . .    | 939 |
| 17.197.1 Macro Definition Documentation . . . . .        | 941 |
| 17.197.1.1 __FFLASFFPACK_fpack_permutation_INL . . . . . | 941 |
| 17.197.1.2 FFLASFFPACK_PERM_BKSIZE . . . . .             | 941 |
| 17.198 fpack_pluq.inl File Reference . . . . .           | 941 |
| 17.198.1 Macro Definition Documentation . . . . .        | 942 |
| 17.198.1.1 __FFLASFFPACK_fpack_pluq_INL . . . . .        | 942 |
| 17.198.1.2 CROUT . . . . .                               | 942 |
| 17.199 fpack_pluq_mp.inl File Reference . . . . .        | 942 |
| 17.199.1 Macro Definition Documentation . . . . .        | 943 |

|  |     |
|--|-----|
| 17.199.1.1 __FFPACK_pluq_mp_INL . . . . .                      | 943 |
| 17.200 ffpack_ppluq.inl File Reference . . . . .               | 943 |
| 17.200.1 Macro Definition Documentation . . . . .              | 943 |
| 17.200.1.1 __FFLASFFPACK_ffpack_ppluq_INL . . . . .            | 943 |
| 17.200.1.2 __FFLAS__TRSM_READONLY . . . . .                    | 943 |
| 17.200.1.3 PBASECASE_K . . . . .                               | 943 |
| 17.201 ffpack_rankprofiles.inl File Reference . . . . .        | 944 |
| 17.201.1 Macro Definition Documentation . . . . .              | 945 |
| 17.201.1.1 __FFLASFFPACK_ffpack_rank_profiles_INL . . . . .    | 945 |
| 17.202 field-traits.h File Reference . . . . .                 | 945 |
| 17.202.1 Detailed Description . . . . .                        | 947 |
| 17.203 field.doxy File Reference . . . . .                     | 947 |
| 17.204 rns-double-elt.h File Reference . . . . .               | 947 |
| 17.204.1 Detailed Description . . . . .                        | 947 |
| 17.205 rns-double-recint.inl File Reference . . . . .          | 948 |
| 17.205.1 Macro Definition Documentation . . . . .              | 948 |
| 17.205.1.1 __FFLASFFPACK_field_rns_double_recint_INL . . . . . | 948 |
| 17.206 rns-double.h File Reference . . . . .                   | 948 |
| 17.206.1 Detailed Description . . . . .                        | 949 |
| 17.206.2 Macro Definition Documentation . . . . .              | 949 |
| 17.206.2.1 ROUND_DOWN . . . . .                                | 949 |
| 17.207 rns-double.inl File Reference . . . . .                 | 949 |
| 17.207.1 Macro Definition Documentation . . . . .              | 949 |
| 17.207.1.1 __FFLASFFPACK_field_rns_double_INL . . . . .        | 949 |
| 17.208 rns-integer-mod.h File Reference . . . . .              | 949 |
| 17.208.1 Detailed Description . . . . .                        | 950 |
| 17.209 rns-integer.h File Reference . . . . .                  | 950 |
| 17.209.1 Detailed Description . . . . .                        | 951 |
| 17.210 rns.h File Reference . . . . .                          | 951 |
| 17.211 rns.inl File Reference . . . . .                        | 951 |
| 17.211.1 Macro Definition Documentation . . . . .              | 951 |
| 17.211.1.1 __FFLASFFPACK_field_rns_INL . . . . .               | 951 |
| 17.212 interfaces.doxy File Reference . . . . .                | 951 |
| 17.213 fflas_c.h File Reference . . . . .                      | 951 |
| 17.213.1 Macro Definition Documentation . . . . .              | 953 |
| 17.213.1.1 FFLAS_COMPILED . . . . .                            | 954 |
| 17.213.2 Enumeration Type Documentation . . . . .              | 954 |
| 17.213.2.1 FFLAS_C_ORDER . . . . .                             | 954 |
| 17.213.2.2 FFLAS_C_TRANSPOSE . . . . .                         | 954 |
| 17.213.2.3 FFLAS_C_UPLO . . . . .                              | 954 |
| 17.213.2.4 FFLAS_C_DIAG . . . . .                              | 954 |
| 17.213.2.5 FFLAS_C_SIDE . . . . .                              | 955 |

|  |     |
|--|-----|
| 17.213.2.6 FFLAS_C_BASE . . . . .                  | 955 |
| 17.213.3 Function Documentation . . . . .          | 955 |
| 17.213.3.1 freducein_1_modular_double() . . . . .  | 955 |
| 17.213.3.2 freduce_1_modular_double() . . . . .    | 955 |
| 17.213.3.3 fnegin_1_modular_double() . . . . .     | 956 |
| 17.213.3.4 fneg_1_modular_double() . . . . .       | 956 |
| 17.213.3.5 fzero_1_modular_double() . . . . .      | 956 |
| 17.213.3.6 fiszero_1_modular_double() . . . . .    | 956 |
| 17.213.3.7 fequal_1_modular_double() . . . . .     | 956 |
| 17.213.3.8 fassign_1_modular_double() . . . . .    | 957 |
| 17.213.3.9 fscaln_1_modular_double() . . . . .     | 957 |
| 17.213.3.10 fscal_1_modular_double() . . . . .     | 957 |
| 17.213.3.11 faxpy_1_modular_double() . . . . .     | 957 |
| 17.213.3.12 fdot_1_modular_double() . . . . .      | 957 |
| 17.213.3.13 fswap_1_modular_double() . . . . .     | 958 |
| 17.213.3.14 fadd_1_modular_double() . . . . .      | 958 |
| 17.213.3.15 fsub_1_modular_double() . . . . .      | 958 |
| 17.213.3.16 faddin_1_modular_double() . . . . .    | 958 |
| 17.213.3.17 fsubin_1_modular_double() . . . . .    | 958 |
| 17.213.3.18 fassign_2_modular_double() . . . . .   | 959 |
| 17.213.3.19 fzero_2_modular_double() . . . . .     | 959 |
| 17.213.3.20 fequal_2_modular_double() . . . . .    | 959 |
| 17.213.3.21 fiszero_2_modular_double() . . . . .   | 959 |
| 17.213.3.22 fidentity_2_modular_double() . . . . . | 959 |
| 17.213.3.23 freducein_2_modular_double() . . . . . | 960 |
| 17.213.3.24 freduce_2_modular_double() . . . . .   | 960 |
| 17.213.3.25 fnegin_2_modular_double() . . . . .    | 960 |
| 17.213.3.26 fneg_2_modular_double() . . . . .      | 960 |
| 17.213.3.27 fscaln_2_modular_double() . . . . .    | 960 |
| 17.213.3.28 fscal_2_modular_double() . . . . .     | 961 |
| 17.213.3.29 faxpy_2_modular_double() . . . . .     | 961 |
| 17.213.3.30 fmove_2_modular_double() . . . . .     | 961 |
| 17.213.3.31 fadd_2_modular_double() . . . . .      | 961 |
| 17.213.3.32 fsub_2_modular_double() . . . . .      | 962 |
| 17.213.3.33 fsubin_2_modular_double() . . . . .    | 962 |
| 17.213.3.34 faddin_2_modular_double() . . . . .    | 962 |
| 17.213.3.35 fgemv_2_modular_double() . . . . .     | 962 |
| 17.213.3.36 fger_2_modular_double() . . . . .      | 963 |
| 17.213.3.37 ftrsv_2_modular_double() . . . . .     | 963 |
| 17.213.3.38 ftrsm_3_modular_double() . . . . .     | 963 |
| 17.213.3.39 ftrmm_3_modular_double() . . . . .     | 963 |
| 17.213.3.40 fgemm_3_modular_double() . . . . .     | 964 |

|  |     |
|--|-----|
| 17.213.3.41 fsquare_3_modular_double()         | 964 |
| 17.214 fflas_L1_inst.C File Reference          | 964 |
| 17.214.1 Macro Definition Documentation        | 965 |
| 17.214.1.1 __FFLAS_L1_INST_C                   | 965 |
| 17.214.1.2 INST_OR_DECL                        | 965 |
| 17.214.1.3 FFLAS_FIELD [1/2]                   | 965 |
| 17.214.1.4 FFLAS_ELT [1/6]                     | 965 |
| 17.214.1.5 FFLAS_ELT [2/6]                     | 965 |
| 17.214.1.6 FFLAS_ELT [3/6]                     | 965 |
| 17.214.1.7 FFLAS_FIELD [2/2]                   | 965 |
| 17.214.1.8 FFLAS_ELT [4/6]                     | 965 |
| 17.214.1.9 FFLAS_ELT [5/6]                     | 966 |
| 17.214.1.10 FFLAS_ELT [6/6]                    | 966 |
| 17.215 fflas_L1_inst.h File Reference          | 966 |
| 17.215.1 Macro Definition Documentation        | 966 |
| 17.215.1.1 INST_OR_DECL                        | 966 |
| 17.215.1.2 FFLAS_FIELD [1/2]                   | 966 |
| 17.215.1.3 FFLAS_ELT [1/6]                     | 966 |
| 17.215.1.4 FFLAS_ELT [2/6]                     | 966 |
| 17.215.1.5 FFLAS_ELT [3/6]                     | 966 |
| 17.215.1.6 FFLAS_FIELD [2/2]                   | 967 |
| 17.215.1.7 FFLAS_ELT [4/6]                     | 967 |
| 17.215.1.8 FFLAS_ELT [5/6]                     | 967 |
| 17.215.1.9 FFLAS_ELT [6/6]                     | 967 |
| 17.216 fflas_L1_inst_implem.inl File Reference | 967 |
| 17.217 fflas_L2_inst.C File Reference          | 968 |
| 17.217.1 Macro Definition Documentation        | 968 |
| 17.217.1.1 __FFLAS_L2_INST_C                   | 969 |
| 17.217.1.2 INST_OR_DECL                        | 969 |
| 17.217.1.3 FFLAS_FIELD [1/2]                   | 969 |
| 17.217.1.4 FFLAS_ELT [1/6]                     | 969 |
| 17.217.1.5 FFLAS_ELT [2/6]                     | 969 |
| 17.217.1.6 FFLAS_ELT [3/6]                     | 969 |
| 17.217.1.7 FFLAS_FIELD [2/2]                   | 969 |
| 17.217.1.8 FFLAS_ELT [4/6]                     | 969 |
| 17.217.1.9 FFLAS_ELT [5/6]                     | 969 |
| 17.217.1.10 FFLAS_ELT [6/6]                    | 969 |
| 17.218 fflas_L2_inst.h File Reference          | 969 |
| 17.218.1 Macro Definition Documentation        | 970 |
| 17.218.1.1 INST_OR_DECL                        | 970 |
| 17.218.1.2 FFLAS_FIELD [1/2]                   | 970 |
| 17.218.1.3 FFLAS_ELT [1/6]                     | 970 |

|  |     |
|--|-----|
| 17.218.1.4 FFLAS_ELT [2/6]                     | 970 |
| 17.218.1.5 FFLAS_ELT [3/6]                     | 970 |
| 17.218.1.6 FFLAS_FIELD [2/2]                   | 970 |
| 17.218.1.7 FFLAS_ELT [4/6]                     | 970 |
| 17.218.1.8 FFLAS_ELT [5/6]                     | 970 |
| 17.218.1.9 FFLAS_ELT [6/6]                     | 970 |
| 17.219 fflas_L2_inst_implem.inl File Reference | 971 |
| 17.220 fflas_L3_inst.C File Reference          | 972 |
| 17.220.1 Macro Definition Documentation        | 973 |
| 17.220.1.1 __FFLAS_L3_INST_C                   | 973 |
| 17.220.1.2 INST_OR_DECL                        | 973 |
| 17.220.1.3 FFLAS_FIELD [1/2]                   | 973 |
| 17.220.1.4 FFLAS_ELT [1/6]                     | 973 |
| 17.220.1.5 FFLAS_ELT [2/6]                     | 973 |
| 17.220.1.6 FFLAS_ELT [3/6]                     | 973 |
| 17.220.1.7 FFLAS_FIELD [2/2]                   | 973 |
| 17.220.1.8 FFLAS_ELT [4/6]                     | 973 |
| 17.220.1.9 FFLAS_ELT [5/6]                     | 973 |
| 17.220.1.10 FFLAS_ELT [6/6]                    | 973 |
| 17.221 fflas_L3_inst.h File Reference          | 973 |
| 17.221.1 Macro Definition Documentation        | 974 |
| 17.221.1.1 INST_OR_DECL                        | 974 |
| 17.221.1.2 FFLAS_FIELD [1/2]                   | 974 |
| 17.221.1.3 FFLAS_ELT [1/6]                     | 974 |
| 17.221.1.4 FFLAS_ELT [2/6]                     | 974 |
| 17.221.1.5 FFLAS_ELT [3/6]                     | 974 |
| 17.221.1.6 FFLAS_FIELD [2/2]                   | 974 |
| 17.221.1.7 FFLAS_ELT [4/6]                     | 974 |
| 17.221.1.8 FFLAS_ELT [5/6]                     | 974 |
| 17.221.1.9 FFLAS_ELT [6/6]                     | 975 |
| 17.222 fflas_L3_inst_implem.inl File Reference | 975 |
| 17.222.1 Macro Definition Documentation        | 975 |
| 17.222.1.1 __FFLAS__TRSM_READONLY              | 975 |
| 17.223 fflas_lvl1.C File Reference             | 976 |
| 17.223.1 Detailed Description                  | 976 |
| 17.223.2 Function Documentation                | 976 |
| 17.223.2.1 freducein_1_modular_double()        | 977 |
| 17.223.2.2 freduce_1_modular_double()          | 977 |
| 17.223.2.3 fnegin_1_modular_double()           | 977 |
| 17.223.2.4 fneg_1_modular_double()             | 977 |
| 17.223.2.5 fzero_1_modular_double()            | 977 |
| 17.223.2.6 fiszero_1_modular_double()          | 977 |

|   |     |
|---|-----|
| 17.223.2.7 fequal_1_modular_double()    | 978 |
| 17.223.2.8 fassign_1_modular_double()   | 978 |
| 17.223.2.9 fscal_1_modular_double()     | 978 |
| 17.223.2.10 fscal_1_modular_double()    | 978 |
| 17.223.2.11 faxpy_1_modular_double()    | 978 |
| 17.223.2.12 fdot_1_modular_double()     | 979 |
| 17.223.2.13 fswap_1_modular_double()    | 979 |
| 17.223.2.14 fadd_1_modular_double()     | 979 |
| 17.223.2.15 fsub_1_modular_double()     | 979 |
| 17.223.2.16 faddin_1_modular_double()   | 980 |
| 17.223.2.17 fsubin_1_modular_double()   | 980 |
| 17.224 fflas_lvl2.C File Reference      | 980 |
| 17.224.1 Detailed Description           | 981 |
| 17.224.2 Function Documentation         | 981 |
| 17.224.2.1 fassign_2_modular_double()   | 981 |
| 17.224.2.2 fzero_2_modular_double()     | 981 |
| 17.224.2.3 fequal_2_modular_double()    | 982 |
| 17.224.2.4 fiszero_2_modular_double()   | 982 |
| 17.224.2.5 fidentity_2_modular_double() | 982 |
| 17.224.2.6 freducein_2_modular_double() | 982 |
| 17.224.2.7 freduce_2_modular_double()   | 982 |
| 17.224.2.8 fnegin_2_modular_double()    | 983 |
| 17.224.2.9 fneg_2_modular_double()      | 983 |
| 17.224.2.10 fscal_2_modular_double()    | 983 |
| 17.224.2.11 fscal_2_modular_double()    | 983 |
| 17.224.2.12 faxpy_2_modular_double()    | 983 |
| 17.224.2.13 fmove_2_modular_double()    | 984 |
| 17.224.2.14 fadd_2_modular_double()     | 984 |
| 17.224.2.15 fsub_2_modular_double()     | 984 |
| 17.224.2.16 fsubin_2_modular_double()   | 984 |
| 17.224.2.17 faddin_2_modular_double()   | 985 |
| 17.224.2.18 fgemv_2_modular_double()    | 985 |
| 17.224.2.19 fger_2_modular_double()     | 985 |
| 17.224.2.20 ftrsv_2_modular_double()    | 985 |
| 17.225 fflas_lvl3.C File Reference      | 986 |
| 17.225.1 Detailed Description           | 986 |
| 17.225.2 Function Documentation         | 986 |
| 17.225.2.1 ftrsm_3_modular_double()     | 986 |
| 17.225.2.2 ftrmm_3_modular_double()     | 987 |
| 17.225.2.3 fgemm_3_modular_double()     | 987 |
| 17.225.2.4 fsquare_3_modular_double()   | 987 |
| 17.226 fflas_sparse.C File Reference    | 988 |

|  |      |
|--|------|
| 17.226.1 Detailed Description . . . . .                          | 988  |
| 17.227 fpack.C File Reference . . . . .                          | 988  |
| 17.227.1 Detailed Description . . . . .                          | 991  |
| 17.227.2 Function Documentation . . . . .                        | 991  |
| 17.227.2.1 LAPACKPerm2MathPerm() . . . . .                       | 991  |
| 17.227.2.2 MathPerm2LAPACKPerm() . . . . .                       | 991  |
| 17.227.2.3 MatrixApplyS_modular_double() . . . . .               | 992  |
| 17.227.2.4 PermApplyS_double() . . . . .                         | 992  |
| 17.227.2.5 MatrixApplyT_modular_double() . . . . .               | 992  |
| 17.227.2.6 PermApplyT_double() . . . . .                         | 992  |
| 17.227.2.7 composePermutationsLLM() . . . . .                    | 992  |
| 17.227.2.8 composePermutationsLLL() . . . . .                    | 993  |
| 17.227.2.9 composePermutationsMLM() . . . . .                    | 993  |
| 17.227.2.10 cyclic_shift_mathPerm() . . . . .                    | 993  |
| 17.227.2.11 cyclic_shift_row_modular_double() . . . . .          | 993  |
| 17.227.2.12 cyclic_shift_col_modular_double() . . . . .          | 993  |
| 17.227.2.13 applyP_modular_double() . . . . .                    | 993  |
| 17.227.2.14 fgetrsin_modular_double() . . . . .                  | 994  |
| 17.227.2.15 fgetrsv_modular_double() . . . . .                   | 994  |
| 17.227.2.16 fgesvin_modular_double() . . . . .                   | 994  |
| 17.227.2.17 fgesv_modular_double() . . . . .                     | 995  |
| 17.227.2.18 ftrtri_modular_double() . . . . .                    | 995  |
| 17.227.2.19 trinv_left_modular_double() . . . . .                | 995  |
| 17.227.2.20 ftrtm_modular_double() . . . . .                     | 995  |
| 17.227.2.21 PLUQ_modular_double() . . . . .                      | 996  |
| 17.227.2.22 LUdivine_modular_double() . . . . .                  | 996  |
| 17.227.2.23 ColumnEchelonForm_modular_double() . . . . .         | 996  |
| 17.227.2.24 RowEchelonForm_modular_double() . . . . .            | 996  |
| 17.227.2.25 ReducedColumnEchelonForm_modular_double() . . . . .  | 997  |
| 17.227.2.26 ReducedRowEchelonForm_modular_double() . . . . .     | 997  |
| 17.227.2.27 ColumnEchelonForm_modular_float() . . . . .          | 997  |
| 17.227.2.28 RowEchelonForm_modular_float() . . . . .             | 997  |
| 17.227.2.29 ReducedColumnEchelonForm_modular_float() . . . . .   | 998  |
| 17.227.2.30 ReducedRowEchelonForm_modular_float() . . . . .      | 998  |
| 17.227.2.31 ColumnEchelonForm_modular_int32_t() . . . . .        | 998  |
| 17.227.2.32 RowEchelonForm_modular_int32_t() . . . . .           | 998  |
| 17.227.2.33 ReducedColumnEchelonForm_modular_int32_t() . . . . . | 999  |
| 17.227.2.34 ReducedRowEchelonForm_modular_int32_t() . . . . .    | 999  |
| 17.227.2.35 pColumnEchelonForm_modular_double() . . . . .        | 999  |
| 17.227.2.36 pRowEchelonForm_modular_double() . . . . .           | 999  |
| 17.227.2.37 pReducedColumnEchelonForm_modular_double() . . . . . | 1000 |
| 17.227.2.38 pReducedRowEchelonForm_modular_double() . . . . .    | 1000 |

|             |   |      |
|-------------|---|------|
| 17.227.2.39 | pColumnEchelonForm_modular_float()              | 1000 |
| 17.227.2.40 | pRowEchelonForm_modular_float()                 | 1000 |
| 17.227.2.41 | pReducedColumnEchelonForm_modular_float()       | 1001 |
| 17.227.2.42 | pReducedRowEchelonForm_modular_float()          | 1001 |
| 17.227.2.43 | pColumnEchelonForm_modular_int32_t()            | 1001 |
| 17.227.2.44 | pRowEchelonForm_modular_int32_t()               | 1001 |
| 17.227.2.45 | pReducedColumnEchelonForm_modular_int32_t()     | 1002 |
| 17.227.2.46 | pReducedRowEchelonForm_modular_int32_t()        | 1002 |
| 17.227.2.47 | Invertin_modular_double()                       | 1002 |
| 17.227.2.48 | Invert_modular_double()                         | 1002 |
| 17.227.2.49 | Invert2_modular_double()                        | 1003 |
| 17.227.2.50 | KrylovElim_modular_double()                     | 1003 |
| 17.227.2.51 | SpecRankProfile_modular_double()                | 1003 |
| 17.227.2.52 | Rank_modular_double()                           | 1003 |
| 17.227.2.53 | IsSingular_modular_double()                     | 1004 |
| 17.227.2.54 | Det_modular_double()                            | 1004 |
| 17.227.2.55 | Solve_modular_double()                          | 1004 |
| 17.227.2.56 | solveLB_modular_double()                        | 1004 |
| 17.227.2.57 | solveLB2_modular_double()                       | 1004 |
| 17.227.2.58 | RandomNullSpaceVector_modular_double()          | 1005 |
| 17.227.2.59 | NullSpaceBasis_modular_double()                 | 1005 |
| 17.227.2.60 | RowRankProfile_modular_double()                 | 1005 |
| 17.227.2.61 | ColumnRankProfile_modular_double()              | 1005 |
| 17.227.2.62 | RankProfileFromLU()                             | 1006 |
| 17.227.2.63 | LeadingSubmatrixRankProfiles()                  | 1006 |
| 17.227.2.64 | RowRankProfileSubmatrixIndices_modular_double() | 1006 |
| 17.227.2.65 | ColRankProfileSubmatrixIndices_modular_double() | 1006 |
| 17.227.2.66 | RowRankProfileSubmatrix_modular_double()        | 1006 |
| 17.227.2.67 | ColRankProfileSubmatrix_modular_double()        | 1007 |
| 17.227.2.68 | getTriangular_modular_double()                  | 1007 |
| 17.227.2.69 | getTriangularin_modular_double()                | 1007 |
| 17.227.2.70 | getEchelonForm_modular_double()                 | 1007 |
| 17.227.2.71 | getEchelonFormin_modular_double()               | 1008 |
| 17.227.2.72 | getEchelonTransform_modular_double()            | 1008 |
| 17.227.2.73 | getReducedEchelonForm_modular_double()          | 1008 |
| 17.227.2.74 | getReducedEchelonFormin_modular_double()        | 1009 |
| 17.227.2.75 | getReducedEchelonTransform_modular_double()     | 1009 |
| 17.227.2.76 | PLUQtoEchelonPermutation()                      | 1009 |
| 17.228      | ffpack_c.h File Reference                       | 1009 |
| 17.228.1    | Macro Definition Documentation                  | 1013 |
| 17.228.1.1  | FFPACK_COMPILED                                 | 1013 |
| 17.228.2    | Enumeration Type Documentation                  | 1013 |



|   |      |
|---|------|
| 17.228.2.1 FFLAS_C_ORDER . . . . .                              | 1013 |
| 17.228.2.2 FFLAS_C_TRANSPOSE . . . . .                          | 1013 |
| 17.228.2.3 FFLAS_C_UPLO . . . . .                               | 1013 |
| 17.228.2.4 FFLAS_C_DIAG . . . . .                               | 1013 |
| 17.228.2.5 FFLAS_C_SIDE . . . . .                               | 1014 |
| 17.228.2.6 FFPACK_C_LU_TAG . . . . .                            | 1014 |
| 17.228.2.7 FFPACK_C_CHARPOLY_TAG . . . . .                      | 1014 |
| 17.228.2.8 FFPACK_C_MINPOLY_TAG . . . . .                       | 1014 |
| 17.228.3 Function Documentation . . . . .                       | 1015 |
| 17.228.3.1 LAPACKPerm2MathPerm() . . . . .                      | 1015 |
| 17.228.3.2 MathPerm2LAPACKPerm() . . . . .                      | 1015 |
| 17.228.3.3 MatrixApplyS_modular_double() . . . . .              | 1015 |
| 17.228.3.4 PermApplyS_double() . . . . .                        | 1015 |
| 17.228.3.5 MatrixApplyT_modular_double() . . . . .              | 1015 |
| 17.228.3.6 PermApplyT_double() . . . . .                        | 1016 |
| 17.228.3.7 composePermutationsLLM() . . . . .                   | 1016 |
| 17.228.3.8 composePermutationsLLL() . . . . .                   | 1016 |
| 17.228.3.9 composePermutationsMLM() . . . . .                   | 1016 |
| 17.228.3.10 cyclic_shift_mathPerm() . . . . .                   | 1016 |
| 17.228.3.11 cyclic_shift_row_modular_double() . . . . .         | 1017 |
| 17.228.3.12 cyclic_shift_col_modular_double() . . . . .         | 1017 |
| 17.228.3.13 applyP_modular_double() . . . . .                   | 1017 |
| 17.228.3.14 fgetrsin_modular_double() . . . . .                 | 1017 |
| 17.228.3.15 fgetrs_modular_double() . . . . .                   | 1017 |
| 17.228.3.16 fgesvin_modular_double() . . . . .                  | 1018 |
| 17.228.3.17 fgesv_modular_double() . . . . .                    | 1018 |
| 17.228.3.18 ftrtri_modular_double() . . . . .                   | 1018 |
| 17.228.3.19 trinv_left_modular_double() . . . . .               | 1019 |
| 17.228.3.20 ftrtrm_modular_double() . . . . .                   | 1019 |
| 17.228.3.21 PLUQ_modular_double() . . . . .                     | 1019 |
| 17.228.3.22 LUdivine_modular_double() . . . . .                 | 1019 |
| 17.228.3.23 LUdivine_small_modular_double() . . . . .           | 1019 |
| 17.228.3.24 LUdivine_gauss_modular_double() . . . . .           | 1020 |
| 17.228.3.25 ColumnEchelonForm_modular_double() . . . . .        | 1020 |
| 17.228.3.26 RowEchelonForm_modular_double() . . . . .           | 1020 |
| 17.228.3.27 ColumnEchelonForm_modular_float() . . . . .         | 1021 |
| 17.228.3.28 RowEchelonForm_modular_float() . . . . .            | 1021 |
| 17.228.3.29 ColumnEchelonForm_modular_int32_t() . . . . .       | 1021 |
| 17.228.3.30 RowEchelonForm_modular_int32_t() . . . . .          | 1021 |
| 17.228.3.31 ReducedColumnEchelonForm_modular_double() . . . . . | 1022 |
| 17.228.3.32 ReducedRowEchelonForm_modular_double() . . . . .    | 1022 |
| 17.228.3.33 ReducedColumnEchelonForm_modular_float() . . . . .  | 1022 |

|             |   |      |
|-------------|---|------|
| 17.228.3.34 | ReducedRowEchelonForm_modular_float()           | 1022 |
| 17.228.3.35 | ReducedColumnEchelonForm_modular_int32_t()      | 1023 |
| 17.228.3.36 | ReducedRowEchelonForm_modular_int32_t()         | 1023 |
| 17.228.3.37 | ReducedRowEchelonForm2_modular_double()         | 1023 |
| 17.228.3.38 | REF_modular_double()                            | 1023 |
| 17.228.3.39 | Invertin_modular_double()                       | 1024 |
| 17.228.3.40 | Invert_modular_double()                         | 1024 |
| 17.228.3.41 | Invert2_modular_double()                        | 1024 |
| 17.228.3.42 | KrylovElim_modular_double()                     | 1024 |
| 17.228.3.43 | SpecRankProfile_modular_double()                | 1024 |
| 17.228.3.44 | Rank_modular_double()                           | 1025 |
| 17.228.3.45 | IsSingular_modular_double()                     | 1025 |
| 17.228.3.46 | Det_modular_double()                            | 1025 |
| 17.228.3.47 | Solve_modular_double()                          | 1025 |
| 17.228.3.48 | solveLB_modular_double()                        | 1025 |
| 17.228.3.49 | solveLB2_modular_double()                       | 1026 |
| 17.228.3.50 | RandomNullSpaceVector_modular_double()          | 1026 |
| 17.228.3.51 | NullSpaceBasis_modular_double()                 | 1026 |
| 17.228.3.52 | RowRankProfile_modular_double()                 | 1026 |
| 17.228.3.53 | ColumnRankProfile_modular_double()              | 1027 |
| 17.228.3.54 | RankProfileFromLU()                             | 1027 |
| 17.228.3.55 | LeadingSubmatrixRankProfiles()                  | 1027 |
| 17.228.3.56 | RowRankProfileSubmatrixIndices_modular_double() | 1027 |
| 17.228.3.57 | ColRankProfileSubmatrixIndices_modular_double() | 1028 |
| 17.228.3.58 | RowRankProfileSubmatrix_modular_double()        | 1028 |
| 17.228.3.59 | ColRankProfileSubmatrix_modular_double()        | 1028 |
| 17.228.3.60 | getTriangular_modular_double()                  | 1028 |
| 17.228.3.61 | getTriangularin_modular_double()                | 1029 |
| 17.228.3.62 | getEchelonForm_modular_double()                 | 1029 |
| 17.228.3.63 | getEchelonFormin_modular_double()               | 1029 |
| 17.228.3.64 | getEchelonTransform_modular_double()            | 1029 |
| 17.228.3.65 | getReducedEchelonForm_modular_double()          | 1030 |
| 17.228.3.66 | getReducedEchelonFormin_modular_double()        | 1030 |
| 17.228.3.67 | getReducedEchelonTransform_modular_double()     | 1030 |
| 17.228.3.68 | PLUQtoEchelonPermutation()                      | 1031 |
| 17.229      | ffpack_inst.C File Reference                    | 1031 |
| 17.229.1    | Macro Definition Documentation                  | 1031 |
| 17.229.1.1  | __FFPACK_INST_C                                 | 1031 |
| 17.229.1.2  | FFLAS_COMPILED                                  | 1031 |
| 17.229.1.3  | INST_OR_DECL                                    | 1031 |
| 17.229.1.4  | FFLAS_FIELD [1/2]                               | 1031 |
| 17.229.1.5  | FFLAS_ELT [1/6]                                 | 1031 |

|  |      |
|--|------|
| 17.229.1.6 FFLAS_ELT [2/6]                   | 1032 |
| 17.229.1.7 FFLAS_ELT [3/6]                   | 1032 |
| 17.229.1.8 FFLAS_FIELD [2/2]                 | 1032 |
| 17.229.1.9 FFLAS_ELT [4/6]                   | 1032 |
| 17.229.1.10 FFLAS_ELT [5/6]                  | 1032 |
| 17.229.1.11 FFLAS_ELT [6/6]                  | 1032 |
| 17.230 fpack_inst.h File Reference           | 1032 |
| 17.230.1 Macro Definition Documentation      | 1032 |
| 17.230.1.1 FFLAS_COMPILED                    | 1032 |
| 17.230.1.2 INST_OR_DECL                      | 1033 |
| 17.230.1.3 FFLAS_FIELD [1/2]                 | 1033 |
| 17.230.1.4 FFLAS_ELT [1/6]                   | 1033 |
| 17.230.1.5 FFLAS_ELT [2/6]                   | 1033 |
| 17.230.1.6 FFLAS_ELT [3/6]                   | 1033 |
| 17.230.1.7 FFLAS_FIELD [2/2]                 | 1033 |
| 17.230.1.8 FFLAS_ELT [4/6]                   | 1033 |
| 17.230.1.9 FFLAS_ELT [5/6]                   | 1033 |
| 17.230.1.10 FFLAS_ELT [6/6]                  | 1033 |
| 17.231 fpack_inst_implem.inl File Reference  | 1033 |
| 17.232 blockcuts.inl File Reference          | 1037 |
| 17.232.1 Macro Definition Documentation      | 1038 |
| 17.232.1.1 __FFLASFFPACK_fflas_blockcuts_INL | 1038 |
| 17.232.1.2 __FFLASFFPACK_MINBLOCKCUTS        | 1038 |
| 17.233 fflas_plevel1.h File Reference        | 1038 |
| 17.234 kaapi_routines.inl File Reference     | 1039 |
| 17.234.1 Macro Definition Documentation      | 1039 |
| 17.234.1.1 __FFLASFFPACK_KAAPI_ROUTINES_INL  | 1039 |
| 17.235 parallel.h File Reference             | 1039 |
| 17.235.1 Macro Definition Documentation      | 1040 |
| 17.235.1.1 __FFLASFFPACK_SEQUENTIAL          | 1040 |
| 17.235.1.2 index_t                           | 1040 |
| 17.235.1.3 TASK                              | 1040 |
| 17.235.1.4 WAIT                              | 1040 |
| 17.235.1.5 CHECK_DEPENDENCIES                | 1040 |
| 17.235.1.6 BARRIER                           | 1040 |
| 17.235.1.7 PAR_BLOCK                         | 1040 |
| 17.235.1.8 SYNCH_GROUP                       | 1040 |
| 17.235.1.9 NUM_THREADS                       | 1040 |
| 17.235.1.10 MAX_THREADS                      | 1041 |
| 17.235.1.11 READ                             | 1041 |
| 17.235.1.12 WRITE                            | 1041 |
| 17.235.1.13 READWRITE                        | 1041 |

|   |      |
|---|------|
| 17.235.1.14 CONSTREFERENCE . . . . .                | 1041 |
| 17.235.1.15 VALUE . . . . .                         | 1041 |
| 17.235.1.16 BEGIN_PARALLEL_MAIN . . . . .           | 1041 |
| 17.235.1.17 END_PARALLEL_MAIN . . . . .             | 1041 |
| 17.235.1.18 FORBLOCK1D . . . . .                    | 1041 |
| 17.235.1.19 FOR1D . . . . .                         | 1042 |
| 17.235.1.20 PARFORBLOCK1D . . . . .                 | 1042 |
| 17.235.1.21 PARFOR1D . . . . .                      | 1042 |
| 17.235.1.22 FORBLOCK2D . . . . .                    | 1042 |
| 17.235.1.23 FOR2D . . . . .                         | 1042 |
| 17.235.1.24 PARFORBLOCK2D . . . . .                 | 1043 |
| 17.235.1.25 PARFOR2D . . . . .                      | 1043 |
| 17.235.1.26 COMMA . . . . .                         | 1043 |
| 17.235.1.27 MODE . . . . .                          | 1043 |
| 17.235.1.28 RETURNPARAM . . . . .                   | 1043 |
| 17.235.1.29 NUMARGS . . . . .                       | 1043 |
| 17.235.1.30 PP_NARG_ . . . . .                      | 1043 |
| 17.235.1.31 PP_ARG_N . . . . .                      | 1043 |
| 17.235.1.32 PP_RSEQ_N . . . . .                     | 1045 |
| 17.235.1.33 NOSPLIT . . . . .                       | 1045 |
| 17.235.1.34 splitting_0 . . . . .                   | 1045 |
| 17.235.1.35 splitting_1 . . . . .                   | 1045 |
| 17.235.1.36 splitting_2 . . . . .                   | 1045 |
| 17.235.1.37 splitting_3 . . . . .                   | 1045 |
| 17.235.1.38 splitt . . . . .                        | 1045 |
| 17.235.1.39 SPLITTER . . . . .                      | 1046 |
| 17.236 pfgemm_variants.inl File Reference . . . . . | 1046 |
| 17.237 pfgemv.inl File Reference . . . . .          | 1047 |
| 17.238 align-allocator.h File Reference . . . . .   | 1047 |
| 17.239 args-parser.h File Reference . . . . .       | 1047 |
| 17.239.1 Macro Definition Documentation . . . . .   | 1048 |
| 17.239.1.1 TYPE_BOOL . . . . .                      | 1048 |
| 17.239.1.2 END_OF_ARGUMENTS . . . . .               | 1048 |
| 17.239.1.3 type_integer . . . . .                   | 1048 |
| 17.239.2 Enumeration Type Documentation . . . . .   | 1048 |
| 17.239.2.1 ArgumentType . . . . .                   | 1048 |
| 17.239.3 Function Documentation . . . . .           | 1048 |
| 17.239.3.1 printHelpMessage() . . . . .             | 1049 |
| 17.239.3.2 findArgument() . . . . .                 | 1049 |
| 17.239.3.3 getListArgs() . . . . .                  | 1049 |
| 17.240 bit_manipulation.h File Reference . . . . .  | 1049 |
| 17.240.1 Macro Definition Documentation . . . . .   | 1049 |

|   |      |
|---|------|
| 17.240.1.1 <code>__has_builtin</code>                   | 1049 |
| 17.240.2 Function Documentation                         | 1050 |
| 17.240.2.1 <code>clz()</code> [1/2]                     | 1050 |
| 17.240.2.2 <code>clz()</code> [2/2]                     | 1050 |
| 17.240.2.3 <code>ctz()</code> [1/2]                     | 1050 |
| 17.240.2.4 <code>ctz()</code> [2/2]                     | 1050 |
| 17.241 <code>cast.h</code> File Reference               | 1050 |
| 17.242 <code>debug.h</code> File Reference              | 1050 |
| 17.242.1 Detailed Description                           | 1051 |
| 17.242.2 Macro Definition Documentation                 | 1051 |
| 17.242.2.1 <code>FFLASFFPACK_check</code>               | 1051 |
| 17.242.2.2 <code>FFLASFFPACK_abort</code>               | 1051 |
| 17.243 <code>fflas_intrinsic.h</code> File Reference    | 1051 |
| 17.244 <code>fflas_io.h</code> File Reference           | 1051 |
| 17.245 <code>fflas_memory.h</code> File Reference       | 1052 |
| 17.246 <code>fflas_randommatrix.h</code> File Reference | 1053 |
| 17.247 <code>flimits.h</code> File Reference            | 1055 |
| 17.247.1 Function Documentation                         | 1055 |
| 17.247.1.1 <code>in_range()</code> [1/3]                | 1055 |
| 17.247.1.2 <code>in_range()</code> [2/3]                | 1056 |
| 17.247.1.3 <code>in_range()</code> [3/3]                | 1056 |
| 17.248 <code>Matio.h</code> File Reference              | 1056 |
| 17.248.1 Function Documentation                         | 1056 |
| 17.248.1.1 <code>read_field()</code>                    | 1056 |
| 17.248.1.2 <code>write_field()</code>                   | 1056 |
| 17.249 <code>test-utils.h</code> File Reference         | 1056 |
| 17.250 <code>timer.h</code> File Reference              | 1057 |
| 17.251 <code>cblas.C</code> File Reference              | 1057 |
| 17.251.1 Macro Definition Documentation                 | 1058 |
| 17.251.1.1 <code>__FFLASFFPACK_CONFIGURATION</code>     | 1058 |
| 17.251.1.2 <code>__FFLASFFPACK_HAVE_CBLAS</code>        | 1058 |
| 17.251.2 Function Documentation                         | 1058 |
| 17.251.2.1 <code>main()</code>                          | 1058 |
| 17.252 <code>clapack.C</code> File Reference            | 1058 |
| 17.252.1 Macro Definition Documentation                 | 1058 |
| 17.252.1.1 <code>__FFLASFFPACK_CONFIGURATION</code>     | 1058 |
| 17.252.1.2 <code>__FFLASFFPACK_HAVE_LAPACK</code>       | 1058 |
| 17.252.1.3 <code>__FFLASFFPACK_HAVE_CLAPACK</code>      | 1059 |
| 17.252.2 Function Documentation                         | 1059 |
| 17.252.2.1 <code>main()</code>                          | 1059 |
| 17.253 <code>cuda.C</code> File Reference               | 1059 |
| 17.253.1 Function Documentation                         | 1059 |

|   |      |
|---|------|
| 17.253.1.1 main()                         | 1059 |
| 17.254 fblas.C File Reference             | 1059 |
| 17.254.1 Macro Definition Documentation   | 1059 |
| 17.254.1.1 __FFLASFFPACK_CONFIGURATION    | 1059 |
| 17.254.2 Function Documentation           | 1060 |
| 17.254.2.1 dgemm_()                       | 1060 |
| 17.254.2.2 main()                         | 1060 |
| 17.255 gmp.C File Reference               | 1060 |
| 17.255.1 Function Documentation           | 1060 |
| 17.255.1.1 main()                         | 1060 |
| 17.256 instrset.h File Reference          | 1060 |
| 17.256.1 Macro Definition Documentation   | 1061 |
| 17.256.1.1 INSTRSET_H                     | 1061 |
| 17.256.1.2 INSTRSET                       | 1061 |
| 17.256.1.3 const_int                      | 1061 |
| 17.256.1.4 const_uint                     | 1061 |
| 17.256.2 Typedef Documentation            | 1061 |
| 17.256.2.1 int8_t                         | 1062 |
| 17.256.2.2 uint8_t                        | 1062 |
| 17.256.2.3 int16_t                        | 1062 |
| 17.256.2.4 uint16_t                       | 1062 |
| 17.256.2.5 int32_t                        | 1062 |
| 17.256.2.6 uint32_t                       | 1062 |
| 17.256.2.7 int64_t                        | 1062 |
| 17.256.2.8 uint64_t                       | 1062 |
| 17.256.2.9 intptr_t                       | 1062 |
| 17.256.3 Function Documentation           | 1062 |
| 17.256.3.1 instrset_detect()              | 1062 |
| 17.256.3.2 hasFMA3()                      | 1062 |
| 17.256.3.3 hasFMA4()                      | 1063 |
| 17.256.3.4 hasXOP()                       | 1063 |
| 17.256.3.5 hasAVX512ER()                  | 1063 |
| 17.257 instrset_detect.cpp File Reference | 1063 |
| 17.257.1 Function Documentation           | 1063 |
| 17.257.1.1 instrset_detect()              | 1063 |
| 17.257.1.2 hasFMA3()                      | 1063 |
| 17.257.1.3 hasFMA4()                      | 1063 |
| 17.257.1.4 hasXOP()                       | 1063 |
| 17.257.1.5 hasF16C()                      | 1064 |
| 17.257.1.6 hasAVX512ER()                  | 1064 |
| 17.258 lapack.C File Reference            | 1064 |
| 17.258.1 Macro Definition Documentation   | 1064 |

|   |      |
|---|------|
| 17.258.1.1 __FFLASFFPACK_CONFIGURATION . . . . .      | 1064 |
| 17.258.1.2 __FFLASFFPACK_HAVE_LAPACK . . . . .        | 1064 |
| 17.258.2 Function Documentation . . . . .             | 1064 |
| 17.258.2.1 main() . . . . .                           | 1064 |
| 17.259 regression-check.C File Reference . . . . .    | 1064 |
| 17.259.1 Function Documentation . . . . .             | 1065 |
| 17.259.1.1 check1() . . . . .                         | 1065 |
| 17.259.1.2 check2() . . . . .                         | 1065 |
| 17.259.1.3 check3() . . . . .                         | 1065 |
| 17.259.1.4 check4() . . . . .                         | 1065 |
| 17.259.1.5 checkZeroDimCharpoly() . . . . .           | 1065 |
| 17.259.1.6 checkZeroDimMinPoly() . . . . .            | 1065 |
| 17.259.1.7 gf2ModularBalanced() . . . . .             | 1065 |
| 17.259.1.8 main() . . . . .                           | 1065 |
| 17.260 test-charpoly-check.C File Reference . . . . . | 1066 |
| 17.260.1 Macro Definition Documentation . . . . .     | 1066 |
| 17.260.1.1 ENABLE_CHECKER_charpoly . . . . .          | 1066 |
| 17.260.1.2 TIME_CHECKER_CHARPOLY . . . . .            | 1066 |
| 17.260.2 Function Documentation . . . . .             | 1066 |
| 17.260.2.1 printPolynomial() . . . . .                | 1066 |
| 17.260.2.2 main() . . . . .                           | 1066 |
| 17.261 test-charpoly.C File Reference . . . . .       | 1066 |
| 17.261.1 Function Documentation . . . . .             | 1067 |
| 17.261.1.1 launch_test() . . . . .                    | 1067 |
| 17.261.1.2 run_with_field() . . . . .                 | 1067 |
| 17.261.1.3 main() . . . . .                           | 1067 |
| 17.262 test-compressQ.C File Reference . . . . .      | 1067 |
| 17.262.1 Typedef Documentation . . . . .              | 1068 |
| 17.262.1.1 Field . . . . .                            | 1068 |
| 17.262.2 Function Documentation . . . . .             | 1068 |
| 17.262.2.1 printvect() . . . . .                      | 1068 |
| 17.262.2.2 main() . . . . .                           | 1068 |
| 17.263 test-det-check.C File Reference . . . . .      | 1068 |
| 17.263.1 Macro Definition Documentation . . . . .     | 1069 |
| 17.263.1.1 ENABLE_CHECKER_Det . . . . .               | 1069 |
| 17.263.1.2 TIME_CHECKER_Det . . . . .                 | 1069 |
| 17.263.2 Function Documentation . . . . .             | 1069 |
| 17.263.2.1 main() . . . . .                           | 1069 |
| 17.264 test-det.C File Reference . . . . .            | 1069 |
| 17.264.1 Function Documentation . . . . .             | 1069 |
| 17.264.1.1 test_det() . . . . .                       | 1070 |
| 17.264.1.2 main() . . . . .                           | 1070 |

|   |      |
|---|------|
| 17.265 test-echelon.C File Reference          | 1070 |
| 17.265.1 Macro Definition Documentation       | 1071 |
| 17.265.1.1 __FFLASFFPACK_SEQUENTIAL           | 1071 |
| 17.265.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE | 1071 |
| 17.265.1.3 __FFLASFFPACK_PLUQ_THRESHOLD       | 1071 |
| 17.265.2 Function Documentation               | 1071 |
| 17.265.2.1 test_colechelon()                  | 1071 |
| 17.265.2.2 test_rowechelon()                  | 1071 |
| 17.265.2.3 test_redcoechelon()                | 1071 |
| 17.265.2.4 test_redrowechelon()               | 1072 |
| 17.265.2.5 run_with_field()                   | 1072 |
| 17.265.2.6 main()                             | 1072 |
| 17.266 test-fadd.C File Reference             | 1072 |
| 17.266.1 Function Documentation               | 1073 |
| 17.266.1.1 test_fadd()                        | 1073 |
| 17.266.1.2 test_faddin()                      | 1073 |
| 17.266.1.3 test_fsub()                        | 1073 |
| 17.266.1.4 test_fsubin()                      | 1073 |
| 17.266.1.5 main()                             | 1074 |
| 17.267 test-fdot.C File Reference             | 1074 |
| 17.267.1 Macro Definition Documentation       | 1074 |
| 17.267.1.1 ENABLE_ALL_CHECKINGS               | 1074 |
| 17.267.2 Function Documentation               | 1074 |
| 17.267.2.1 check_fdot()                       | 1074 |
| 17.267.2.2 run_with_field()                   | 1075 |
| 17.267.2.3 run_with_Integer()                 | 1075 |
| 17.267.2.4 main()                             | 1075 |
| 17.268 test-fgemm-check.C File Reference      | 1075 |
| 17.268.1 Macro Definition Documentation       | 1075 |
| 17.268.1.1 ENABLE_ALL_CHECKINGS               | 1075 |
| 17.268.2 Function Documentation               | 1075 |
| 17.268.2.1 launch_MM_dispatch()               | 1076 |
| 17.268.2.2 run_with_field()                   | 1076 |
| 17.268.2.3 main()                             | 1076 |
| 17.269 test-fgemm.C File Reference            | 1076 |
| 17.269.1 Macro Definition Documentation       | 1077 |
| 17.269.1.1 ENABLE_CHECKER_fgemm               | 1077 |
| 17.269.2 Function Documentation               | 1077 |
| 17.269.2.1 check_MM()                         | 1077 |
| 17.269.2.2 launch_MM()                        | 1077 |
| 17.269.2.3 launch_MM_dispatch()               | 1078 |
| 17.269.2.4 run_with_field()                   | 1078 |



|   |      |
|---|------|
| 17.269.2.5 main()                       | 1078 |
| 17.270 test-fgemv.C File Reference      | 1079 |
| 17.270.1 Function Documentation         | 1079 |
| 17.270.1.1 check_MV()                   | 1079 |
| 17.270.1.2 launch_MV()                  | 1079 |
| 17.270.1.3 launch_MV_dispatch()         | 1080 |
| 17.270.1.4 run_with_field()             | 1080 |
| 17.270.1.5 main()                       | 1080 |
| 17.271 test-fger.C File Reference       | 1080 |
| 17.271.1 Macro Definition Documentation | 1081 |
| 17.271.1.1 TIME                         | 1081 |
| 17.271.2 Function Documentation         | 1081 |
| 17.271.2.1 check_fger()                 | 1081 |
| 17.271.2.2 launch_fger()                | 1081 |
| 17.271.2.3 launch_fger_dispatch()       | 1082 |
| 17.271.2.4 run_with_field()             | 1082 |
| 17.271.2.5 main()                       | 1082 |
| 17.272 test-fgesv.C File Reference      | 1082 |
| 17.272.1 Function Documentation         | 1083 |
| 17.272.1.1 test_square_fgesv()          | 1083 |
| 17.272.1.2 test_rect_fgesv()            | 1083 |
| 17.272.1.3 run_with_field()             | 1083 |
| 17.272.1.4 main()                       | 1083 |
| 17.273 test-finit.C File Reference      | 1084 |
| 17.273.1 Function Documentation         | 1084 |
| 17.273.1.1 test_freduce()               | 1084 |
| 17.273.1.2 run_with_field()             | 1084 |
| 17.273.1.3 main()                       | 1084 |
| 17.274 test-fscal.C File Reference      | 1085 |
| 17.274.1 Function Documentation         | 1085 |
| 17.274.1.1 test_fscal() [1/2]           | 1085 |
| 17.274.1.2 test_fscal() [2/2]           | 1085 |
| 17.274.1.3 test_fscalin() [1/2]         | 1085 |
| 17.274.1.4 test_fscalin() [2/2]         | 1086 |
| 17.274.1.5 main()                       | 1086 |
| 17.275 test-fsyr2k.C File Reference     | 1086 |
| 17.275.1 Macro Definition Documentation | 1086 |
| 17.275.1.1 ENABLE_ALL_CHECKINGS         | 1086 |
| 17.275.2 Function Documentation         | 1087 |
| 17.275.2.1 check_fsyr2k()               | 1087 |
| 17.275.2.2 run_with_field()             | 1087 |
| 17.275.2.3 main()                       | 1087 |

|   |      |
|---|------|
| 17.276 test-fsyk.C File Reference       | 1087 |
| 17.276.1 Macro Definition Documentation | 1088 |
| 17.276.1.1 ENABLE_ALL_CHECKINGS         | 1088 |
| 17.276.2 Function Documentation         | 1088 |
| 17.276.2.1 check_fsyk()                 | 1088 |
| 17.276.2.2 check_fsyk_diag()            | 1088 |
| 17.276.2.3 check_fsyk_bkdiag()          | 1088 |
| 17.276.2.4 run_with_field()             | 1089 |
| 17.276.2.5 main()                       | 1089 |
| 17.277 test-fsytrf.C File Reference     | 1089 |
| 17.277.1 Function Documentation         | 1089 |
| 17.277.1.1 operator<<()                 | 1090 |
| 17.277.1.2 test_RPM_fsytrf()            | 1090 |
| 17.277.1.3 test_generic_fsytrf()        | 1090 |
| 17.277.1.4 run_with_field()             | 1090 |
| 17.277.1.5 main()                       | 1090 |
| 17.278 test-frm.C File Reference        | 1090 |
| 17.278.1 Macro Definition Documentation | 1091 |
| 17.278.1.1 __FFLASFFPACK_SEQUENTIAL     | 1091 |
| 17.278.2 Function Documentation         | 1091 |
| 17.278.2.1 check_frm()                  | 1091 |
| 17.278.2.2 run_with_field()             | 1091 |
| 17.278.2.3 main()                       | 1092 |
| 17.279 test-frm.C File Reference        | 1092 |
| 17.279.1 Macro Definition Documentation | 1092 |
| 17.279.1.1 __FFLASFFPACK_SEQUENTIAL     | 1092 |
| 17.279.1.2 ENABLE_ALL_CHECKINGS         | 1092 |
| 17.279.2 Function Documentation         | 1092 |
| 17.279.2.1 check_frm()                  | 1092 |
| 17.279.2.2 run_with_field()             | 1093 |
| 17.279.2.3 main()                       | 1093 |
| 17.280 test-frm-check.C File Reference  | 1093 |
| 17.280.1 Macro Definition Documentation | 1093 |
| 17.280.1.1 ENABLE_ALL_CHECKINGS         | 1093 |
| 17.280.2 Function Documentation         | 1093 |
| 17.280.2.1 main()                       | 1093 |
| 17.281 test-frm.C File Reference        | 1094 |
| 17.281.1 Macro Definition Documentation | 1094 |
| 17.281.1.1 __FFLASFFPACK_SEQUENTIAL     | 1094 |
| 17.281.1.2 ENABLE_ALL_CHECKINGS         | 1094 |
| 17.281.2 Function Documentation         | 1094 |
| 17.281.2.1 check_frm()                  | 1094 |

|   |      |
|---|------|
| 17.281.2.2 run_with_field()               | 1095 |
| 17.281.2.3 main()                         | 1095 |
| 17.282 test-ftsyr2k.C File Reference      | 1095 |
| 17.282.1 Macro Definition Documentation   | 1095 |
| 17.282.1.1 ENABLE_ALL_CHECKINGS           | 1095 |
| 17.282.2 Function Documentation           | 1096 |
| 17.282.2.1 check_ftsyr2k()                | 1096 |
| 17.282.2.2 run_with_field()               | 1096 |
| 17.282.2.3 main()                         | 1096 |
| 17.283 test-ftsstr.C File Reference       | 1096 |
| 17.283.1 Macro Definition Documentation   | 1097 |
| 17.283.1.1 ENABLE_ALL_CHECKINGS           | 1097 |
| 17.283.2 Function Documentation           | 1097 |
| 17.283.2.1 check_ftsstr()                 | 1097 |
| 17.283.2.2 run_with_field()               | 1097 |
| 17.283.2.3 main()                         | 1097 |
| 17.284 test-ftsrv.C File Reference        | 1097 |
| 17.284.1 Macro Definition Documentation   | 1098 |
| 17.284.1.1 __FFLASFFPACK_SEQUENTIAL       | 1098 |
| 17.284.1.2 ENABLE_ALL_CHECKINGS           | 1098 |
| 17.284.2 Function Documentation           | 1098 |
| 17.284.2.1 check_ftsrv()                  | 1098 |
| 17.284.2.2 run_with_field()               | 1098 |
| 17.284.2.3 main()                         | 1098 |
| 17.285 test-ftsrti.C File Reference       | 1098 |
| 17.285.1 Macro Definition Documentation   | 1099 |
| 17.285.1.1 __FFLASFFPACK_SEQUENTIAL       | 1099 |
| 17.285.1.2 ENABLE_ALL_CHECKINGS           | 1099 |
| 17.285.2 Function Documentation           | 1099 |
| 17.285.2.1 check_ftsrti()                 | 1099 |
| 17.285.2.2 run_with_field()               | 1099 |
| 17.285.2.3 main()                         | 1100 |
| 17.286 test-interfaces-c.c File Reference | 1100 |
| 17.286.1 Function Documentation           | 1100 |
| 17.286.1.1 main()                         | 1100 |
| 17.287 test-invert-check.C File Reference | 1100 |
| 17.287.1 Macro Definition Documentation   | 1100 |
| 17.287.1.1 ENABLE_ALL_CHECKINGS           | 1100 |
| 17.287.2 Function Documentation           | 1100 |
| 17.287.2.1 main()                         | 1101 |
| 17.288 test-io.C File Reference           | 1101 |
| 17.288.1 Function Documentation           | 1101 |

|   |      |
|---|------|
| 17.288.1.1 run_with_field()                         | 1101 |
| 17.288.1.2 main()                                   | 1101 |
| 17.289 test-lu.C File Reference                     | 1102 |
| 17.289.1 Macro Definition Documentation             | 1102 |
| 17.289.1.1 BASECASE_K                               | 1102 |
| 17.289.1.2 __FFLASFFPACK_SEQUENTIAL                 | 1103 |
| 17.289.1.3 __LUDIVINE_CUTOFF                        | 1103 |
| 17.289.2 Function Documentation                     | 1103 |
| 17.289.2.1 test_LUdivine()                          | 1103 |
| 17.289.2.2 verifPLUQ()                              | 1103 |
| 17.289.2.3 test_pluq()                              | 1104 |
| 17.289.2.4 launch_test()                            | 1105 |
| 17.289.2.5 run_with_field()                         | 1105 |
| 17.289.2.6 main()                                   | 1105 |
| 17.289.3 Variable Documentation                     | 1105 |
| 17.289.3.1 tperm                                    | 1105 |
| 17.289.3.2 tgemm                                    | 1105 |
| 17.289.3.3 tBC                                      | 1105 |
| 17.289.3.4 ttrsm                                    | 1105 |
| 17.289.3.5 trest                                    | 1105 |
| 17.289.3.6 timtot                                   | 1106 |
| 17.289.3.7 mvcnt                                    | 1106 |
| 17.290 test-maxdelayeddim.C File Reference          | 1106 |
| 17.290.1 Macro Definition Documentation             | 1106 |
| 17.290.1.1 MAX_WITH_SIZE_T                          | 1106 |
| 17.290.2 Function Documentation                     | 1106 |
| 17.290.2.1 test()                                   | 1106 |
| 17.290.2.2 main()                                   | 1106 |
| 17.291 test-minpoly.C File Reference                | 1107 |
| 17.291.1 Function Documentation                     | 1107 |
| 17.291.1.1 check_minpoly()                          | 1107 |
| 17.291.1.2 run_with_field()                         | 1107 |
| 17.291.1.3 main()                                   | 1107 |
| 17.292 test-multifile1.C File Reference             | 1108 |
| 17.293 test-multifile2.C File Reference             | 1108 |
| 17.293.1 Function Documentation                     | 1108 |
| 17.293.1.1 main()                                   | 1108 |
| 17.294 test-nullspace.C File Reference              | 1108 |
| 17.294.1 Function Documentation                     | 1108 |
| 17.294.1.1 checkingMessage()                        | 1109 |
| 17.294.1.2 readOrRandomMatrixWithRankAndRandomRPM() | 1109 |
| 17.294.1.3 test_nullspace()                         | 1109 |

|   |      |
|---|------|
| 17.294.1.4 run_with_field()               | 1109 |
| 17.294.1.5 main()                         | 1109 |
| 17.295 test-permutations.C File Reference | 1109 |
| 17.295.1 Function Documentation           | 1110 |
| 17.295.1.1 checkMonotonicApplyP()         | 1110 |
| 17.295.1.2 main()                         | 1110 |
| 17.295.2 Variable Documentation           | 1110 |
| 17.295.2.1 tperm                          | 1110 |
| 17.295.2.2 tgemm                          | 1110 |
| 17.295.2.3 tBC                            | 1110 |
| 17.295.2.4 ttrsm                          | 1110 |
| 17.295.2.5 trest                          | 1111 |
| 17.295.2.6 timtot                         | 1111 |
| 17.296 test-pluq-check.C File Reference   | 1111 |
| 17.296.1 Macro Definition Documentation   | 1111 |
| 17.296.1.1 ENABLE_ALL_CHECKINGS           | 1111 |
| 17.296.2 Function Documentation           | 1111 |
| 17.296.2.1 main()                         | 1111 |
| 17.297 test-rankprofiles.C File Reference | 1111 |
| 17.297.1 Macro Definition Documentation   | 1112 |
| 17.297.1.1 __FFLASFFPACK_SEQUENTIAL       | 1112 |
| 17.297.2 Function Documentation           | 1112 |
| 17.297.2.1 run_with_field()               | 1112 |
| 17.297.2.2 main()                         | 1112 |
| 17.298 test-rpm.C File Reference          | 1112 |
| 17.298.1 Function Documentation           | 1113 |
| 17.298.1.1 checkRPM()                     | 1113 |
| 17.298.1.2 checkSymmetricRPM()            | 1113 |
| 17.298.1.3 main()                         | 1113 |
| 17.299 test-simd.C File Reference         | 1113 |
| 17.299.1 Macro Definition Documentation   | 1114 |
| 17.299.1.1 REGISTER_TYPE_NAME             | 1114 |
| 17.299.1.2 TEST_ONE_OP                    | 1114 |
| 17.299.2 Typedef Documentation            | 1114 |
| 17.299.2.1 integer                        | 1115 |
| 17.299.3 Function Documentation           | 1115 |
| 17.299.3.1 TypeName()                     | 1115 |
| 17.299.3.2 REGISTER_TYPE_NAME() [1/8]     | 1115 |
| 17.299.3.3 REGISTER_TYPE_NAME() [2/8]     | 1115 |
| 17.299.3.4 REGISTER_TYPE_NAME() [3/8]     | 1115 |
| 17.299.3.5 REGISTER_TYPE_NAME() [4/8]     | 1115 |
| 17.299.3.6 REGISTER_TYPE_NAME() [5/8]     | 1115 |

|  |      |
|--|------|
| 17.299.3.7 REGISTER_TYPE_NAME() [6/8]      | 1115 |
| 17.299.3.8 REGISTER_TYPE_NAME() [7/8]      | 1115 |
| 17.299.3.9 REGISTER_TYPE_NAME() [8/8]      | 1115 |
| 17.299.3.10 generate_random_vector() [1/2] | 1116 |
| 17.299.3.11 generate_random_vector() [2/2] | 1116 |
| 17.299.3.12 check_eq() [1/2]               | 1116 |
| 17.299.3.13 check_eq() [2/2]               | 1116 |
| 17.299.3.14 eval_func_on_array() [1/2]     | 1116 |
| 17.299.3.15 eval_func_on_array() [2/2]     | 1116 |
| 17.299.3.16 test_op()                      | 1116 |
| 17.299.3.17 test_impl() [1/2]              | 1116 |
| 17.299.3.18 test_impl() [2/2]              | 1116 |
| 17.299.3.19 test() [1/2]                   | 1117 |
| 17.299.3.20 test() [2/2]                   | 1117 |
| 17.299.3.21 main()                         | 1117 |
| 17.300 test-solve.C File Reference         | 1117 |
| 17.300.1 Function Documentation            | 1117 |
| 17.300.1.1 check_solve()                   | 1117 |
| 17.300.1.2 run_with_field()                | 1117 |
| 17.300.1.3 main()                          | 1118 |
| 17.301 101-fgemv.C File Reference          | 1118 |
| 17.301.1 Function Documentation            | 1118 |
| 17.301.1.1 main()                          | 1118 |
| 17.302 2x2-fgemv.C File Reference          | 1118 |
| 17.302.1 Function Documentation            | 1118 |
| 17.302.1.1 main()                          | 1118 |
| 17.303 2x2-ftsrv.C File Reference          | 1119 |
| 17.303.1 Function Documentation            | 1119 |
| 17.303.1.1 main()                          | 1119 |
| 17.304 2x2-pluq.C File Reference           | 1119 |
| 17.304.1 Function Documentation            | 1119 |
| 17.304.1.1 main()                          | 1119 |
| 17.305 fflas-101_1.C File Reference        | 1119 |
| 17.305.1 Function Documentation            | 1120 |
| 17.305.1.1 main()                          | 1120 |
| 17.306 fflas-101_3.C File Reference        | 1120 |
| 17.306.1 Function Documentation            | 1120 |
| 17.306.1.1 main()                          | 1120 |
| 17.307 fflas_101.C File Reference          | 1120 |
| 17.307.1 Function Documentation            | 1120 |
| 17.307.1.1 main()                          | 1121 |
| 17.308 fflas_101_lvl1.C File Reference     | 1121 |

---

|   |             |
|---|-------------|
| 17.308.1 Function Documentation . . . . .     | 1121        |
| 17.308.1.1 main() . . . . .                   | 1121        |
| 17.309 fpack-fgesv.C File Reference . . . . . | 1121        |
| 17.309.1 Function Documentation . . . . .     | 1121        |
| 17.309.1.1 main() . . . . .                   | 1121        |
| 17.310 fpack-solve.C File Reference . . . . . | 1121        |
| 17.310.1 Function Documentation . . . . .     | 1122        |
| 17.310.1.1 main() . . . . .                   | 1122        |
| <b>Index</b>                                  | <b>1123</b> |





# Chapter 1

## FFLAS-FFPACK Documentation.

### 1.1 Introduction

FFLAS-FFPACK is a LGPL-2.1+ source code library for basic linear algebra operations over a finite field. It is inspired by BLAS interface (Basic Linear Algebra Subprograms) and the LAPACK library for numerical linear algebra, and shares part of their design. Yet it differs in many aspects due to the specificities of computing over a finite field:

- it is generic with respect to the finite field, so as to accomodate a large variety of field sizes and implementations;
- it is a pure source code library, to be included and compiled in the user's software. Its build system is only used for tests and benchmarks.

### 1.2 Goals

### 1.3 Design

### 1.4 Using FFLAS-FFPACK.

- [Copying and Licence](#).
- [Tutorial](#). This is a brief introduction to FFLAS-FFPACK capabilities.
- [Configuring and Installing FFLAS-FFPACK](#). Explains how to configure/install from sources or from the latest svn version.
- [Architecture of the library](#).. Describes how FFLAS-FFPACK is organized
- [Documentation for Users](#). If everything around is blue, then you are reading the lighter, user-oriented, documentation.
- [Documentation for Developers](#). If everything around is green, then you can get to everything (not necessarily yet) documented.

### 1.5 Contributing to fflas-ffpack, getting assistance.

Version

2.3.0



## Chapter 2

# Configuring and Installing FFLAS-FFPACK

FFLAS-FFPACK is a header-only package.

Howver configuration process can be tweaked a lot. Configure looks for BLAS routines and [Givaro](#) library which are both mandatory dependencies. See the output of `./configure -help` for information about the LAPACK/↵ BLAS discovering strategies.



## Chapter 3

# Copying and Licence

The FFLAS-FFPACK library is licensed under the terms of the GNU LGPL v2.1 or later.

See <https://www.gnu.org/licenses/lgpl-2.1.html>



## Chapter 4

# Tutorial

no doc.





## Chapter 5

# Architecture of the library.

no doc.



## Chapter 6

# Bug List

Global **DOUBLE\_TO\_FLOAT\_CROSSOVER**

to be benchmarked.

Global **FFLAS::details::pack\_lhs** (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)

this is fassign

this is fassign

Global **FFLAS::details::pack\_rhs** (int64\_t \*XX, const int64\_t \*X, size\_t ldx, size\_t rows, size\_t cols)

this is fassign

this is fassign

Global **FFLAS::fconvert** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, FFLAS\_ELT \*X, const size\_t incX, const FFLAS\_ELT \*Y, const size\_t incY)

use cblas\_(d)scal when possible

Global **FFLAS::fconvert** (const Field &F, const size\_t n, OtherElement\_ptr X, const size\_t incX, typename Field::ConstElement\_ptr Y, const size\_t incY)

use cblas\_(d)scal when possible

Global **FFLAS::finit** (const Field &F, const size\_t n, const OtherElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::finit** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fneg** (const Field &F, const size\_t n, typename Field::ConstElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fneg** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fnegin** (const Field &F, const size\_t n, typename Field::Element\_ptr X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::fnegin** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, FFLAS\_ELT \*X, const size\_t incX)

use cblas\_(d)scal when possible

Global **FFLAS::freduce** (const Field &F, const size\_t n, typename Field::ConstElement\_ptr Y, const size\_t incY, typename Field::Element\_ptr X, const size\_t incX)

use cblas\_(d)scal when possible

- Global **FFLAS::reduce** (const Field &F, const size\_t n, typename **Field::Element\_ptr** X, const size\_t incX)  
use cblas\_(d)scal when possible
- Global **FFLAS::reduce** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, FFLAS\_ELT \*X, const size\_t incX)  
use cblas\_(d)scal when possible
- Global **FFLAS::reduce** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)  
use cblas\_(d)scal when possible
- Global **FFLAS::fscal** (const Field &F, const size\_t n, const typename **Field::Element** alpha, typename **Field::ConstElement\_ptr** X, const size\_t incX, typename **Field::Element\_ptr** Y, const size\_t incY)  
use cblas\_(d)scal when possible
- Global **FFLAS::fscal** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, const FFLAS\_ELT \*X, const size\_t incX, FFLAS\_ELT \*Y, const size\_t incY)  
use cblas\_(d)scal when possible
- Global **FFLAS::fscaln** (const Field &F, const size\_t n, const typename **Field::Element** alpha, typename **Field::Element\_ptr** X, const size\_t incX)  
use cblas\_(d)scal when possible
- Global **FFLAS::fscaln** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, FFLAS\_ELT \*X, const size\_t incX)  
use cblas\_(d)scal when possible
- Global **FFLAS::fsquare** (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const typename **Field::Element** alpha, typename **Field::ConstElement\_ptr** A, const size\_t lda, const typename **Field::Element** beta, typename **Field::Element\_ptr** C, const size\_t ldc)  
why double ?
- Global **FFLAS::fswap** (const Field &F, const size\_t N, typename **Field::Element\_ptr** X, const size\_t incX, typename **Field::Element\_ptr** Y, const size\_t incY)  
use cblas\_dswap when double
- Global **FFLAS::fswap** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t N, FFLAS\_ELT \*X, const size\_t incX, FFLAS\_ELT \*Y, const size\_t incY)  
use cblas\_dswap when double
- Global **FFLAS::ftrsm** (const Field &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename **Field::Element** alpha, typename **Field::ConstElement\_ptr** A, const size\_t lda, typename **Field::Element\_ptr** B, const size\_t ldb)  
 $\alpha$  must be non zero.
- Global **FFLAS::ftrsm** (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const FFLAS\_ELT alpha, const FFLAS\_ELT \*A, const size\_t lda, FFLAS\_ELT \*B, const size\_t ldb)  
 $\alpha$  must be non zero.
- Global **FFPACK::buildMatrix** (const Field &F, typename **Field::ConstElement\_ptr** E, typename **Field::ConstElement\_ptr** C, const size\_t lda, const size\_t \*B, const size\_t \*T, const size\_t me, const size\_t mc, const size\_t lambda, const size\_t mu)  
is this :
- Global **FFPACK::invert2** (const Field &F, const size\_t M, typename **Field::Element\_ptr** A, const size\_t lda, typename **Field::Element\_ptr** X, const size\_t idx, int &nullity)  
not tested.
- Global **launch\_fger\_dispatch** (const Field &F, const size\_t nn, const typename **Field::Element** alpha, const size\_t iters, RandIter &G)  
test for transpo  
test for incx equal

---

Global `launch_MM_dispatch` (const Field &F, const int mm, const int nn, const int kk, const typename `Field::Element` alpha, const typename `Field::Element` beta, const size\_t iters, RandIter &G)

Global `launch_MM_dispatch` (const Field &F, const int mm, const int nn, const int kk, const typename `Field::Element` alpha, const typename `Field::Element` beta, const size\_t iters, const int nbw, const bool par, RandIter &G)

test for IdX equal

test for transpo

Global `launch_MM_dispatch` (const Field &F, const int mm, const int nn, const int kk, const typename `Field::Element` alpha, const typename `Field::Element` beta, const size\_t iters, RandIter &G)

test for transpo

test for IdX equal

Global `printvect` (std::ostream &o, vector< T > &vect)

does not belong here



## Chapter 7

# Bibliography

- Global **FFLAS::Protected::TRSMBound** (const Givaro::ModularBalanced< Element > &F) .  
Dumas Giorgi Pernet 06, arXiv:cs/0601133
- Global **FFPACK::LeadingSubmatrixRankProfiles** (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP) .  
Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13.
- Global **FFPACK::LUdivine** (const Field &F, const **FFLAS::FFLAS\_DIAG** Diag, const **FFLAS::FFLAS\_TRANSPOSE** trans, const size\_t M, const size\_t N, typename **Field::Element\_ptr** A, const size\_t lda, size\_t \*P, size\_t \*Qt, const FFPACK\_LU\_TAG LuTag=FfpackSlabRecursive, const size\_t cutoff=\_\_FFLASFFPACK\_LUDIVINE\_THRESHOLD) .  
Jeannerod C-P, Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013  
• Pernet C, Brassel M *LUdivine, une divine factorisation LU*, 2002
- Global **FFPACK::PLUQ** (const Field &F, const **FFLAS::FFLAS\_DIAG** Diag, const size\_t M, const size\_t N, typename **Field::Element\_ptr** A, const size\_t lda, size\_t \*P, size\_t \*Q) .  
Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13, 2013
- Global **FFPACK::Protected::GaussJordan** (const Field &F, const size\_t M, const size\_t N, typename **Field::Element\_ptr** A, const size\_t lda, const size\_t colbeg, const size\_t rowbeg, const size\_t colsize, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag) .  
Algorithm 2.8 of A. Storjohann Thesis 2000,  
• Algorithm 11 of Jeannerod C-P., Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013
- Class **ftsmLeftUpperNoTransNonUnit**< Element > .  
Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.





## Chapter 8

# Todo List

### File `debug.h`

we should put vector printing elsewhere.

Global `FFLAS::fadd` (const Field &F, const size\_t N, typename `Field::ConstElement_ptr` A, const size\_t incA, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` B, const size\_t incB, typename `Field::Element_ptr` C, const size\_t incC)

optimise here

Global `FFLAS::fassign` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t N, const FFLAS\_ELT \*Y, const size\_t incY, FFLAS\_ELT \*X, const size\_t incX)

variant for triangular matrix

Global `FFLAS::fassign` (const Field &F, const size\_t N, typename `Field::ConstElement_ptr` Y, const size\_t incY, typename `Field::Element_ptr` X, const size\_t incX)

variant for triangular matrix

Global `FFLAS::fconvert` (const Field &F, const size\_t m, const size\_t n, OtherElement\_ptr A, const size\_t lda, typename `Field::ConstElement_ptr` B, const size\_t ldb)

check if n == lda

Global `FFLAS::fneg` (const Field &F, const size\_t m, const size\_t n, typename `Field::ConstElement_ptr` B, const size\_t ldb, typename `Field::Element_ptr` A, const size\_t lda)

check if n == lda

Global `FFLAS::fnegin` (const Field &F, const size\_t m, const size\_t n, typename `Field::Element_ptr` A, const size\_t lda)

check if n == lda

Global `FFLAS::fscal` (const Field &F, const size\_t n, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` X, const size\_t incX, typename `Field::Element_ptr` Y, const size\_t incY)

check if comparison with +/-1,0 is necessary.

Global `FFLAS::fscal` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, const FFLAS\_ELT \*X, const size\_t incX, FFLAS\_ELT \*Y, const size\_t incY)

check if comparison with +/-1,0 is necessary.

Global `FFLAS::fscaln` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t n, const FFLAS\_ELT alpha, FFLAS\_ELT \*X, const size\_t incX)

check if comparison with +/-1,0 is necessary.

Global `FFLAS::fscaln` (const Field &F, const size\_t n, const typename `Field::Element` alpha, typename `Field::Element_ptr` X, const size\_t incX)

check if comparison with +/-1,0 is necessary.

Global **FFLAS::Protected::igemm** (const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, size\_t rows, size\_t cols, size\_t depth, const int64\_t alpha, const int64\_t \*A, size\_t lda, const int64\_t \*B, size\_t ldb, const int64\_t beta, int64\_t \*C, size\_t ldc)

use primitive (no [Field\(\)](#)) and specialise for int64.

Global **FFLAS::Protected::MatF2MatFI\_Triangular** (const Field &F, Givaro::FloatDomain::Element\_ptr S, const size\_t lds, typename [Field::ConstElement\\_ptr](#) const E, const size\_t lde, const size\_t m, const size\_t n)

do finit(...,FFLAS\_TRANS,FFLAS\_DIAG)

do fconvert(...,FFLAS\_TRANS,FFLAS\_DIAG)

Global **FFPACK::getTriangular** (const Field &F, const **FFLAS::FFLAS\_UPLO** Uplo, const **FFLAS::FFLAS\_DIAG** diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false)

just one triangular fzero+fassign ?

Global **FFPACK::getTriangular** (const Field &F, const **FFLAS::FFLAS\_UPLO** Uplo, const **FFLAS::FFLAS\_DIAG** diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda)

just one triangular fzero+fassign ?

Global **FFPACK::invert2** (const Field &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, int &>nullity)

this init is not all necessary (done after ftrtri)

Global **FFPACK::LUdivine** (const Field &F, const **FFLAS::FFLAS\_DIAG** Diag, const **FFLAS::FFLAS\_TRANSPOSE** trans, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const **FFPACK::FFPACK\_LU\_TAG** LuTag, const size\_t cutoff)

std::swap ?

Global **FFPACK::Protected::RandomKrylovPrecond** (const PolRing &PR, std::list< typename PolRing::Element > &completedFactors, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t lda, size\_t &Nb, typename PolRing::Domain\_t::Element\_ptr &B, size\_t &ldb, typename PolRing::Domain\_t::RandIter &g, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)

don't assing K2 c\*noc x N but only mas (c,noc) x N and store each one after the other

swap to save space ??

## Module [field](#)

biblio

Global **launch\_fger\_dispatch** (const Field &F, const size\_t nn, const typename [Field::Element](#) alpha, const size\_t iters, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

Global **launch\_MM\_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size\_t iters, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

Global **launch\_MM\_dispatch** (const Field &F, const int mm, const int nn, const int kk, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size\_t iters, const int nbw, const bool par, RandIter &G)

does nbw actually do nbw recursive calls and then call blas (check ?) ?

## Module [MMalgos](#)

biblio

## Module [simd](#)

biblio

Global **test\_colechelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, **FFPACK::FFPACK\_LU\_TAG** LuTag, RandIter &G, bool par)

check lda

---

Global **test\_det** (Field &F, size\_t n, int iter, RandIter &G)

test with stride

Global **test\_redcochelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, FFPACK::FFPACK\_LU\_TAG LuTag, RandIter &G, bool par)

check Ida

Global **test\_redrowechelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, FFPACK::FFPACK\_LU\_TAG LuTag, RandIter &G, bool par)

check Ida

Global **test\_rowechelon** (Field &F, size\_t m, size\_t n, size\_t r, size\_t iters, FFPACK::FFPACK\_LU\_TAG LuTag, RandIter &G, bool par)

check Ida



## Chapter 9

# Module Index

### 9.1 Modules

Here is a list of all modules:

|  |    |
|--|----|
| CHECKER . . . . .                          | 49 |
| FFLAS-FFPACK . . . . .                     | 49 |
| FFLAS . . . . .                            | 50 |
| Interfaces . . . . .                       | 51 |
| Matrix Multiplication Algorithms . . . . . | 50 |
| SIMD wrapper . . . . .                     | 50 |
| FFPACK . . . . .                           | 50 |
| FFLAS-FFPACK fields . . . . .              | 51 |
| RNS . . . . .                              | 51 |



## Chapter 10

# Namespace Index

### 10.1 Namespace List

Here is a list of all namespaces with brief descriptions:

|   |     |
|---|-----|
| FFLAS   | 53  |
| FFLAS::BLAS3  | 199 |
| FFLAS::csr_hyb_details  | 205 |
| FFLAS::CuttingStrategy  | 205 |
| FFLAS::details  | 206 |
| FFLAS::details_spmv   | 214 |
| FFLAS::ElementCategories  | 214 |
| FFLAS::FieldCategories  |     |
| Traits and categories will need to be placed in a proper file later                 | 215 |
| FFLAS::MMHelperAlgo   | 215 |
| FFLAS::ModeCategories   |     |
| Specifies the mode of action for an algorithm w.r.t                                 | 215 |
| FFLAS::ParSeqHelper   |     |
| ParSeqHelper for both fgemm and ftrsm   | 216 |
| FFLAS::Protected  | 216 |
| FFLAS::sell_details   | 229 |
| FFLAS::sparse_details   | 230 |
| FFLAS::sparse_details_impl  | 244 |
| FFLAS::StrategyParameter  | 286 |
| FFLAS::StructureHelper  |     |
| StructureHelper for ftrsm   | 286 |
| FFLAS::vectorised   | 286 |
| FFLAS::vectorised::unswitch   | 292 |
| FFPACK  |     |
| Finite Field <b>PACK</b> Set of elimination based routines for dense linear algebra | 293 |
| FFPACK::Protected   | 395 |
| Givaro  | 403 |
| MKL_CONFIG  | 403 |
| RecInt  | 403 |





## Chapter 11

# Hierarchical Index

### 11.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

|  |     |
|--|-----|
| AlgoChooser< ModeT, ParSeq > . . . . .   | 405 |
| AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > . . . . . | 405 |
| ArbitraryPrecIntTag . . . . .  | 405 |
| AreEqual< X, Y > . . . . .   | 406 |
| AreEqual< X, X > . . . . .   | 406 |
| Argument . . . . .   | 406 |
| associatedDelayedField< Field > . . . . .  | 407 |
| associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > . . . . .                         | 407 |
| associatedDelayedField< const Givaro::Modular< T, X > > . . . . .                              | 408 |
| associatedDelayedField< const Givaro::ModularBalanced< T > > . . . . .                         | 408 |
| associatedDelayedField< const Givaro::ZRing< T > > . . . . .                                   | 409 |
| Auto . . . . .   | 409 |
| Bini . . . . .   | 409 |
| Block . . . . .  | 409 |
| callLUdivine_small< Element > . . . . .  | 410 |
| callLUdivine_small< double > . . . . .   | 410 |
| callLUdivine_small< float > . . . . .  | 411 |
| CharpolyFailed . . . . .   | 411 |
| Checker_Empty< Field > . . . . .   | 411 |
| CheckerImplem_charpoly< Field, Polynomial > . . . . .  | 412 |
| CheckerImplem_Det< Field > . . . . .   | 413 |
| CheckerImplem_fgemm< Field > . . . . .   | 414 |
| CheckerImplem_ftdsm< Field > . . . . .   | 415 |
| CheckerImplem_invert< Field > . . . . .  | 416 |
| CheckerImplem_PLUQ< Field > . . . . .  | 417 |
| Classic . . . . .  | 418 |
| Column . . . . .   | 418 |
| CompactElement< Element > . . . . .  | 418 |
| CompactElement< double > . . . . .   | 418 |
| CompactElement< float > . . . . .  | 419 |
| CompactElement< int16_t > . . . . .  | 419 |
| CompactElement< int32_t > . . . . .  | 419 |
| CompactElement< int64_t > . . . . .  | 420 |
| compatible_data_type< Field > . . . . .  | 420 |
| compatible_data_type< Givaro::ZRing< double > > . . . . .                                      | 420 |

|   |     |
|---|-----|
| compatible_data_type< Givaro::ZRing< float > > . . . . .  | 420 |
| Compose< H1, H2 > . . . . .                               | 421 |
| Const_int_t< n > . . . . .                                | 422 |
| Const_uint_t< n > . . . . .                               | 422 |
| Simd128_impl< true, true, false, 2 >::Converter . . . . . | 422 |
| Simd128_impl< true, true, false, 4 >::Converter . . . . . | 423 |
| Simd128_impl< true, true, false, 8 >::Converter . . . . . | 423 |
| Simd128_impl< true, true, true, 2 >::Converter . . . . .  | 423 |
| Simd128_impl< true, true, true, 4 >::Converter . . . . .  | 424 |
| Simd128_impl< true, true, true, 8 >::Converter . . . . .  | 424 |
| Simd256_impl< true, true, false, 2 >::Converter . . . . . | 425 |
| Simd256_impl< true, true, false, 4 >::Converter . . . . . | 425 |
| Simd256_impl< true, true, false, 8 >::Converter . . . . . | 425 |
| Simd256_impl< true, true, true, 2 >::Converter . . . . .  | 426 |
| Simd256_impl< true, true, true, 4 >::Converter . . . . .  | 426 |
| Simd256_impl< true, true, true, 8 >::Converter . . . . .  | 427 |
| Simd512_impl< true, true, false, 8 >::Converter . . . . . | 427 |
| Simd512_impl< true, true, true, 8 >::Converter . . . . .  | 427 |
| ConvertTo< T > . . . . .                                  | 428 |
| Coo< ValT, IdxT > . . . . .                               | 428 |
| Coo< Field > . . . . .                                    | 430 |
| Coo< ValT, IdxT > . . . . .                               | 431 |
| CooMat< Field > . . . . .                                 | 433 |
| CooMat< FFPACK::RNSInteger > . . . . .                    | 433 |
| CsrMat< Field > . . . . .                                 | 434 |
| CsrMat< FFPACK::RNSInteger > . . . . .                    | 434 |
| DefaultBoundedTag . . . . .                               | 434 |
| DefaultTag . . . . .                                      | 435 |
| DelayedTag . . . . .                                      | 435 |
| ElementTraits< Element > . . . . .                        | 435 |
| ElementTraits< double > . . . . .                         | 435 |
| ElementTraits< FFPACK::rns_double_elt > . . . . .         | 436 |
| ElementTraits< float > . . . . .                          | 436 |
| ElementTraits< Givaro::Integer > . . . . .                | 436 |
| ElementTraits< int16_t > . . . . .                        | 437 |
| ElementTraits< int32_t > . . . . .                        | 437 |
| ElementTraits< int64_t > . . . . .                        | 437 |
| ElementTraits< int8_t > . . . . .                         | 438 |
| ElementTraits< Reclnt::rint< K > > . . . . .              | 438 |
| ElementTraits< Reclnt::rmint< K, MG > > . . . . .         | 438 |
| ElementTraits< Reclnt::ruint< K > > . . . . .             | 439 |
| ElementTraits< uint16_t > . . . . .                       | 439 |
| ElementTraits< uint32_t > . . . . .                       | 439 |
| ElementTraits< uint64_t > . . . . .                       | 440 |
| ElementTraits< uint8_t > . . . . .                        | 440 |
| EllMat< Field > . . . . .                                 | 440 |
| EllMat< FFPACK::RNSInteger > . . . . .                    | 440 |
| Failure . . . . .   | 441 |
| FailureCharpolyCheck . . . . .                            | 443 |
| FailureDetCheck . . . . .                                 | 443 |
| FailureFgemmCheck . . . . .                               | 443 |
| FailureInvertCheck . . . . .                              | 443 |
| FailurePLUQCheck . . . . .                                | 444 |
| FailureTrsmCheck . . . . .                                | 444 |
| false_type  |     |
| isSparseMatrix< Field, M > . . . . .                      | 479 |
| isSparseMatrixMKLFormat< F, M > . . . . .                 | 483 |
| isSparseMatrixSimdFormat< F, M > . . . . .                | 483 |

|  |     |
|--|-----|
| isZOSparseMatrix< F, M > . . . . .                           | 484 |
| support_fast_mod< T > . . . . .                              | 762 |
| support_simd< T > . . . . .                                  | 763 |
| support_simd_add< T > . . . . .                              | 763 |
| support_simd_mod< T > . . . . .                              | 763 |
| FieldSimd< _Field > . . . . .                                | 444 |
| FieldTraits< Field > . . . . .                               | 450 |
| FieldTraits< FFPACK::RNSInteger< T > > . . . . .             | 451 |
| FieldTraits< FFPACK::RNSIntegerMod< T > > . . . . .          | 451 |
| FieldTraits< Givaro::Modular< Element > > . . . . .          | 452 |
| FieldTraits< Givaro::ModularBalanced< Element > > . . . . .  | 452 |
| FieldTraits< Givaro::ZRing< double > > . . . . .             | 453 |
| FieldTraits< Givaro::ZRing< float > > . . . . .              | 453 |
| FieldTraits< Givaro::ZRing< Givaro::Integer > > . . . . .    | 454 |
| FieldTraits< Givaro::ZRing< int16_t > > . . . . .            | 455 |
| FieldTraits< Givaro::ZRing< int32_t > > . . . . .            | 455 |
| FieldTraits< Givaro::ZRing< int64_t > > . . . . .            | 456 |
| FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > > . . . . . | 456 |
| FieldTraits< Givaro::ZRing< uint16_t > > . . . . .           | 457 |
| FieldTraits< Givaro::ZRing< uint32_t > > . . . . .           | 457 |
| FieldTraits< Givaro::ZRing< uint64_t > > . . . . .           | 458 |
| Fixed . . . . .  | 458 |
| FixedPrecIntTag . . . . .                                    | 458 |
| ForStrategy1D< blocksize_t, Cut, Param > . . . . .           | 459 |
| ForStrategy2D< blocksize_t, Cut, Param > . . . . .           | 461 |
| ftmmLeftLowerNoTransNonUnit< Element > . . . . .             | 464 |
| ftmmLeftLowerNoTransUnit< Element > . . . . .                | 464 |
| ftmmLeftLowerTransNonUnit< Element > . . . . .               | 465 |
| ftmmLeftLowerTransUnit< Element > . . . . .                  | 465 |
| ftmmLeftUpperNoTransNonUnit< Element > . . . . .             | 465 |
| ftmmLeftUpperNoTransUnit< Element > . . . . .                | 465 |
| ftmmLeftUpperTransNonUnit< Element > . . . . .               | 465 |
| ftmmLeftUpperTransUnit< Element > . . . . .                  | 465 |
| ftmmRightLowerNoTransNonUnit< Element > . . . . .            | 465 |
| ftmmRightLowerNoTransUnit< Element > . . . . .               | 465 |
| ftmmRightLowerTransNonUnit< Element > . . . . .              | 466 |
| ftmmRightLowerTransUnit< Element > . . . . .                 | 466 |
| ftmmRightUpperNoTransNonUnit< Element > . . . . .            | 466 |
| ftmmRightUpperNoTransUnit< Element > . . . . .               | 466 |
| ftmmRightUpperTransNonUnit< Element > . . . . .              | 466 |
| ftmmRightUpperTransUnit< Element > . . . . .                 | 466 |
| ftsmLeftLowerNoTransNonUnit< Element > . . . . .             | 466 |
| ftsmLeftLowerNoTransUnit< Element > . . . . .                | 466 |
| ftsmLeftLowerTransNonUnit< Element > . . . . .               | 467 |
| ftsmLeftLowerTransUnit< Element > . . . . .                  | 467 |
| ftsmLeftUpperNoTransNonUnit< Element > . . . . .             | 467 |
| ftsmLeftUpperNoTransUnit< Element > . . . . .                | 467 |
| ftsmLeftUpperTransNonUnit< Element > . . . . .               | 467 |
| ftsmLeftUpperTransUnit< Element > . . . . .                  | 467 |
| ftsmRightLowerNoTransNonUnit< Element > . . . . .            | 468 |
| ftsmRightLowerNoTransUnit< Element > . . . . .               | 468 |
| ftsmRightLowerTransNonUnit< Element > . . . . .              | 468 |
| ftsmRightLowerTransUnit< Element > . . . . .                 | 468 |
| ftsmRightUpperNoTransNonUnit< Element > . . . . .            | 468 |
| ftsmRightUpperNoTransUnit< Element > . . . . .               | 468 |
| ftsmRightUpperTransNonUnit< Element > . . . . .              | 468 |
| ftsmRightUpperTransUnit< Element > . . . . .                 | 468 |
| GenericTag . . . . .   | 469 |

|  |     |
|--|-----|
| GenericTag   | 469 |
| Grain  | 469 |
| has_minus_eq_impl< C >   | 469 |
| has_minus_impl< C >  | 469 |
| has_mul_eq_impl< C >   | 470 |
| has_mul_impl< C >  | 470 |
| has_operation< T >   | 470 |
| has_plus_eq_impl< C >  | 471 |
| has_plus_impl< C >   | 471 |
| HelperFlag   | 471 |
| HelperMod< Field, ElementTraits >  | 472 |
| HelperMod< Field, ElementCategories::MachineIntTag >   | 472 |
| HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >  | 474 |
| HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >  | 474 |
| HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >  | 475 |
| Hybrid   | 476 |
| Info   | 476 |
| Info   | 477 |
| is_simd< T >   | 479 |
| Iterative  | 485 |
| LazyTag  | 486 |
| limits< T >  | 486 |
| limits< char >   | 486 |
| limits< double >   | 487 |
| limits< float >  | 487 |
| limits< Givaro::Integer >  | 488 |
| limits< int >  | 489 |
| limits< long >   | 489 |
| limits< long long >  | 490 |
| limits< Reclnt::rint< K > >  | 491 |
| limits< Reclnt::ruint< K > >   | 492 |
| limits< short int >  | 492 |
| limits< signed char >  | 493 |
| limits< unsigned char >  | 494 |
| limits< unsigned int >   | 494 |
| limits< unsigned long >  | 495 |
| limits< unsigned long long >   | 496 |
| limits< unsigned short int >   | 497 |
| MachineFloatTag  | 497 |
| MachineIntTag  | 498 |
| MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >   | 498 |
| MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >                  | 503 |
| MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >               | 505 |
| MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >                             | 507 |
| MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > | 509 |
| MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >                                    | 511 |
| ModeTraits< Field >  | 513 |
| ModeTraits< Givaro::Modular< Element, Compute > >  | 513 |
| ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >  | 513 |
| ModeTraits< Givaro::Modular< int16_t, Compute > >  | 514 |
| ModeTraits< Givaro::Modular< int32_t, Compute > >  | 514 |
| ModeTraits< Givaro::Modular< int8_t, Compute > >   | 514 |
| ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >   | 515 |
| ModeTraits< Givaro::Modular< uint16_t, Compute > >   | 515 |
| ModeTraits< Givaro::Modular< uint32_t, Compute > >   | 516 |
| ModeTraits< Givaro::Modular< uint8_t, Compute > >  | 516 |
| ModeTraits< Givaro::ModularBalanced< Element > >   | 516 |

|   |     |
|---|-----|
| ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > . . . . .                                  | 517 |
| ModeTraits< Givaro::ModularBalanced< int16_t > > . . . . .  | 517 |
| ModeTraits< Givaro::ModularBalanced< int32_t > > . . . . .  | 517 |
| ModeTraits< Givaro::ModularBalanced< int8_t > > . . . . .   | 518 |
| ModeTraits< Givaro::Montgomery< T > > . . . . .   | 518 |
| ModeTraits< Givaro::ZRing< double > > . . . . .   | 518 |
| ModeTraits< Givaro::ZRing< float > > . . . . .  | 519 |
| ModeTraits< Givaro::ZRing< Givaro::Integer > > . . . . .  | 519 |
| ModularBalanced< T > . . . . .  | 519 |
| ModularTag . . . . .  | 520 |
| Montgomery< T > . . . . .   | 520 |
| need_field_characteristic< Field > . . . . .  | 520 |
| need_field_characteristic< Givaro::Modular< Field > > . . . . .                                     | 520 |
| need_field_characteristic< Givaro::ModularBalanced< Field > > . . . . .                             | 520 |
| NoSimd< T > . . . . .   | 521 |
| Parallel< C, P > . . . . .  | 522 |
| readMyMachineType< Field, T > . . . . .   | 525 |
| readMyMachineType< Field, mpz_t > . . . . .   | 526 |
| Recursive . . . . .   | 527 |
| Recursive . . . . .   | 527 |
| rint< K > . . . . .   | 527 |
| rns_double . . . . .  | 527 |
| rns_double_elt . . . . .  | 532 |
| rns_double_elt_cstptr . . . . .   | 534 |
| rns_double_elt_ptr . . . . .  | 537 |
| rns_double_extended . . . . .   | 540 |
| RNSElementTag . . . . .   | 544 |
| RNSInteger< RNS > . . . . .   | 545 |
| RNSInteger< FFPACK::rns_double > . . . . .  | 545 |
| RNSIntegerMod< RNS > . . . . .  | 548 |
| RNSIntegerMod< FFPACK::rns_double > . . . . .   | 548 |
| rnsRandIter< RNS > . . . . .  | 555 |
| RNSInteger< RNS >::RandIter . . . . .   | 523 |
| RNSIntegerMod< RNS >::RandIter . . . . .  | 524 |
| Row . . . . .   | 556 |
| ruint< K > . . . . .  | 556 |
| ScalFunctions< Element, Enable > . . . . .  | 556 |
| ScalFunctions< Element, typename enable_if< is_floating_point< Element >::value >::type > . . . . . | 557 |
| ScalFunctions< Element, typename enable_if< is_integral< Element >::value >::type > . . . . .       | 560 |
| Sequential . . . . .  | 566 |
| Simd128_impl< ArithType, Int, Signed, Size > . . . . .  | 567 |
| Simd128fp_base . . . . .  | 623 |
| Simd128_impl< true, false, true, 4 > . . . . .  | 567 |
| Simd128_impl< true, false, true, 8 > . . . . .  | 567 |
| Simd128i_base . . . . .   | 624 |
| Simd128_impl< true, true, true, 2 > . . . . .   | 597 |
| Simd128_impl< true, true, false, 2 > . . . . .  | 568 |
| Simd128_impl< true, true, true, 4 > . . . . .   | 606 |
| Simd128_impl< true, true, false, 4 > . . . . .  | 577 |
| Simd128_impl< true, true, true, 8 > . . . . .   | 615 |
| Simd128_impl< true, true, false, 8 > . . . . .  | 587 |
| Simd256_impl< ArithType, Int, Signed, Size > . . . . .  | 626 |
| Simd256fp_base . . . . .  | 704 |
| Simd256_impl< true, false, true, 4 > . . . . .  | 626 |
| Simd256_impl< true, false, true, 8 > . . . . .  | 626 |
| Simd256i_base . . . . .   | 705 |

|   |     |
|---|-----|
| Simd256_impl< true, true, true, 2 > . . . . .                           | 671 |
| Simd256_impl< true, true, false, 2 > . . . . .                          | 633 |
| Simd256_impl< true, true, true, 4 > . . . . .                           | 679 |
| Simd256_impl< true, true, false, 4 > . . . . .                          | 643 |
| Simd256_impl< true, true, false, 4 > . . . . .                          | 643 |
| Simd256_impl< true, true, true, 8 > . . . . .                           | 696 |
| Simd256_impl< true, true, false, 8 > . . . . .                          | 661 |
| Simd512_impl< ArithType, Int, Signed, Size > . . . . .                  | 706 |
| Simd512fp_base . . . . .  | 733 |
| Simd512_impl< true, false, true, 4 > . . . . .                          | 706 |
| Simd512_impl< true, false, true, 8 > . . . . .                          | 706 |
| Simd512i_base . . . . .   | 734 |
| Simd256_impl< true, true, true, 4 > . . . . .                           | 679 |
| Simd512_impl< true, true, true, 8 > . . . . .                           | 724 |
| Simd512_impl< true, true, false, 8 > . . . . .                          | 713 |
| SimdChooser< T, bool, bool > . . . . .                                  | 735 |
| SimdChooser< T, false, b > . . . . .                                    | 735 |
| SimdChooser< T, true, false > . . . . .                                 | 735 |
| SimdChooser< T, true, true > . . . . .                                  | 736 |
| simdToType< T > . . . . .   | 736 |
| Single . . . . .  | 736 |
| Sparse< Field, SparseMatrix_t, IdxT, PtrT > . . . . .                   | 736 |
| Sparse< _Field, SparseMatrix_t::COO > . . . . .                         | 737 |
| Sparse< _Field, SparseMatrix_t::COO_ZO > . . . . .                      | 738 |
| Sparse< _Field, SparseMatrix_t::CSR > . . . . .                         | 740 |
| Sparse< _Field, SparseMatrix_t::CSR_ZO > . . . . .                      | 743 |
| Sparse< _Field, SparseMatrix_t::CSR_HYB > . . . . .                     | 742 |
| Sparse< _Field, SparseMatrix_t::ELL > . . . . .                         | 745 |
| Sparse< _Field, SparseMatrix_t::ELL_ZO > . . . . .                      | 750 |
| Sparse< _Field, SparseMatrix_t::ELL_simd > . . . . .                    | 747 |
| Sparse< _Field, SparseMatrix_t::ELL_simd_ZO > . . . . .                 | 748 |
| Sparse< _Field, SparseMatrix_t::HYB_ZO > . . . . .                      | 752 |
| Sparse< _Field, SparseMatrix_t::SELL > . . . . .                        | 753 |
| Sparse< _Field, SparseMatrix_t::SELL_ZO > . . . . .                     | 755 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int16_t > . . . . .    | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int32_t > . . . . .    | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO, int64_t > . . . . .    | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int16_t > . . . . . | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int32_t > . . . . . | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::COO_ZO, int64_t > . . . . . | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int16_t > . . . . .    | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int32_t > . . . . .    | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR, int64_t > . . . . .    | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int16_t > . . . . . | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int32_t > . . . . . | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::CSR_ZO, int64_t > . . . . . | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int16_t > . . . . .    | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int32_t > . . . . .    | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL, int64_t > . . . . .    | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int16_t > . . . . . | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int32_t > . . . . . | 736 |
| Sparse< FFPACK::RNSInteger, SparseMatrix_t::ELL_ZO, int64_t > . . . . . | 736 |
| SpMat< Field, flag > . . . . .  | 757 |
| Static_error_check< bool > . . . . .                                    | 758 |
| Static_error_check< false > . . . . .                                   | 758 |
| StatsMatrix . . . . .   | 758 |

|   |     |
|---|-----|
| tfn_minus   | 764 |
| tfn_minus_eq  | 764 |
| tfn_mul   | 764 |
| tfn_mul_eq  | 765 |
| tfn_plus  | 765 |
| tfn_plus_eq   | 765 |
| Threads   | 766 |
| ThreeD  | 766 |
| ThreeDAdaptive  | 766 |
| ThreeDInPlace   | 766 |
| TRSMHelper< ReclterTrait, ParSeqTrait >                                 | 766 |
| true_type   |     |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >           | 480 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >        | 480 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >           | 480 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >       | 480 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >        | 481 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >           | 481 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >        | 482 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >      | 481 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >   | 482 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >        | 482 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >          | 483 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >       | 483 |
| isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >      | 484 |
| isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >      | 484 |
| isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >      | 485 |
| isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > > | 485 |
| isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >     | 485 |
| support_fast_mod< double >  | 762 |
| support_fast_mod< float >   | 762 |
| support_fast_mod< int64_t >   | 762 |
| TwoD  | 768 |
| TwoDAdaptive  | 768 |
| UnparametricTag   | 768 |
| Winograd  | 768 |
| WinogradPar   | 768 |





## Chapter 12

# Data Structure Index

### 12.1 Data Structures

Here are the data structures with brief descriptions:

|  |     |
|--|-----|
| <a href="#">AlgoChooser&lt; ModeT, ParSeq &gt;</a>   | 405 |
| <a href="#">AlgoChooser&lt; ModeCategories::ConvertTo&lt; ElementCategories::RNSElementTag &gt;, ParSeq &gt;</a> | 405 |
| <a href="#">ArbitraryPrecIntTag</a>  |     |
| Arbitrary precision integers: GMP  | 405 |
| <a href="#">AreEqual&lt; X, Y &gt;</a>   | 406 |
| <a href="#">AreEqual&lt; X, X &gt;</a>   | 406 |
| <a href="#">Argument</a>   | 406 |
| <a href="#">associatedDelayedField&lt; Field &gt;</a>  | 407 |
| <a href="#">associatedDelayedField&lt; const FFPACK::RNSIntegerMod&lt; RNS &gt; &gt;</a>                         | 407 |
| <a href="#">associatedDelayedField&lt; const Givaro::Modular&lt; T, X &gt; &gt;</a>                              | 408 |
| <a href="#">associatedDelayedField&lt; const Givaro::ModularBalanced&lt; T &gt; &gt;</a>                         | 408 |
| <a href="#">associatedDelayedField&lt; const Givaro::ZRing&lt; T &gt; &gt;</a>                                   | 409 |
| <a href="#">Auto</a>   | 409 |
| <a href="#">Bini</a>   | 409 |
| <a href="#">Block</a>  | 409 |
| <a href="#">callLUdivine_small&lt; Element &gt;</a>  | 410 |
| <a href="#">callLUdivine_small&lt; double &gt;</a>   | 410 |
| <a href="#">callLUdivine_small&lt; float &gt;</a>  | 411 |
| <a href="#">CharpolyFailed</a>   | 411 |
| <a href="#">Checker_Empty&lt; Field &gt;</a>   | 411 |
| <a href="#">CheckerImplem_charpoly&lt; Field, Polynomial &gt;</a>  | 412 |
| <a href="#">CheckerImplem_Det&lt; Field &gt;</a>   | 413 |
| <a href="#">CheckerImplem_fgemm&lt; Field &gt;</a>   | 414 |
| <a href="#">CheckerImplem_ftdsm&lt; Field &gt;</a>   | 415 |
| <a href="#">CheckerImplem_invert&lt; Field &gt;</a>  | 416 |
| <a href="#">CheckerImplem_PLUQ&lt; Field &gt;</a>  | 417 |
| <a href="#">Classic</a>  | 418 |
| <a href="#">Column</a>   | 418 |
| <a href="#">CompactElement&lt; Element &gt;</a>  | 418 |
| <a href="#">CompactElement&lt; double &gt;</a>   | 418 |
| <a href="#">CompactElement&lt; float &gt;</a>  | 419 |
| <a href="#">CompactElement&lt; int16_t &gt;</a>  | 419 |
| <a href="#">CompactElement&lt; int32_t &gt;</a>  | 419 |
| <a href="#">CompactElement&lt; int64_t &gt;</a>  | 420 |
| <a href="#">compatible_data_type&lt; Field &gt;</a>  | 420 |

|  |     |
|--|-----|
| <a href="#">compatible_data_type&lt; Givaro::ZRing&lt; double &gt; &gt;</a>      | 420 |
| <a href="#">compatible_data_type&lt; Givaro::ZRing&lt; float &gt; &gt;</a>       | 420 |
| <a href="#">Compose&lt; H1, H2 &gt;</a>  | 421 |
| <a href="#">Const_int_t&lt; n &gt;</a>   | 422 |
| <a href="#">Const_uint_t&lt; n &gt;</a>  | 422 |
| <a href="#">Simd128_impl&lt; true, true, false, 2 &gt;::Converter</a>            | 422 |
| <a href="#">Simd128_impl&lt; true, true, false, 4 &gt;::Converter</a>            | 423 |
| <a href="#">Simd128_impl&lt; true, true, false, 8 &gt;::Converter</a>            | 423 |
| <a href="#">Simd128_impl&lt; true, true, true, 2 &gt;::Converter</a>             | 423 |
| <a href="#">Simd128_impl&lt; true, true, true, 4 &gt;::Converter</a>             | 424 |
| <a href="#">Simd128_impl&lt; true, true, true, 8 &gt;::Converter</a>             | 424 |
| <a href="#">Simd256_impl&lt; true, true, false, 2 &gt;::Converter</a>            | 425 |
| <a href="#">Simd256_impl&lt; true, true, false, 4 &gt;::Converter</a>            | 425 |
| <a href="#">Simd256_impl&lt; true, true, false, 8 &gt;::Converter</a>            | 425 |
| <a href="#">Simd256_impl&lt; true, true, true, 2 &gt;::Converter</a>             | 426 |
| <a href="#">Simd256_impl&lt; true, true, true, 4 &gt;::Converter</a>             | 426 |
| <a href="#">Simd256_impl&lt; true, true, true, 8 &gt;::Converter</a>             | 427 |
| <a href="#">Simd512_impl&lt; true, true, false, 8 &gt;::Converter</a>            | 427 |
| <a href="#">Simd512_impl&lt; true, true, true, 8 &gt;::Converter</a>             | 427 |
| <a href="#">ConvertTo&lt; T &gt;</a>   |     |
| Force conversion to appropriate element type of ElementCategory T                | 428 |
| <a href="#">Coo&lt; ValT, IdxT &gt;</a>  | 428 |
| <a href="#">Coo&lt; Field &gt;</a>   | 430 |
| <a href="#">Coo&lt; ValT, IdxT &gt;</a>  | 431 |
| <a href="#">CooMat&lt; Field &gt;</a>  | 433 |
| <a href="#">CsrMat&lt; Field &gt;</a>  | 434 |
| <a href="#">DefaultBoundedTag</a>  |     |
| Use standard field operations, but keeps track of bounds on input and output     | 434 |
| <a href="#">DefaultTag</a>   |     |
| No specific mode of action: use standard field operations                        | 435 |
| <a href="#">DelayedTag</a>   |     |
| Performs field operations with delayed mod reductions. Ensures result is reduced | 435 |
| <a href="#">ElementTraits&lt; Element &gt;</a>                                   |     |
| <a href="#">ElementTraits</a>  | 435 |
| <a href="#">ElementTraits&lt; double &gt;</a>                                    | 435 |
| <a href="#">ElementTraits&lt; FFPACK::rns_double_elt &gt;</a>                    | 436 |
| <a href="#">ElementTraits&lt; float &gt;</a>                                     | 436 |
| <a href="#">ElementTraits&lt; Givaro::Integer &gt;</a>                           | 436 |
| <a href="#">ElementTraits&lt; int16_t &gt;</a>                                   | 437 |
| <a href="#">ElementTraits&lt; int32_t &gt;</a>                                   | 437 |
| <a href="#">ElementTraits&lt; int64_t &gt;</a>                                   | 437 |
| <a href="#">ElementTraits&lt; int8_t &gt;</a>                                    | 438 |
| <a href="#">ElementTraits&lt; Reclnt::rint&lt; K &gt; &gt;</a>                   | 438 |
| <a href="#">ElementTraits&lt; Reclnt::rmint&lt; K, MG &gt; &gt;</a>              | 438 |
| <a href="#">ElementTraits&lt; Reclnt::ruint&lt; K &gt; &gt;</a>                  | 439 |
| <a href="#">ElementTraits&lt; uint16_t &gt;</a>                                  | 439 |
| <a href="#">ElementTraits&lt; uint32_t &gt;</a>                                  | 439 |
| <a href="#">ElementTraits&lt; uint64_t &gt;</a>                                  | 440 |
| <a href="#">ElementTraits&lt; uint8_t &gt;</a>                                   | 440 |
| <a href="#">EllMat&lt; Field &gt;</a>  | 440 |
| <a href="#">Failure</a>  |     |
| A precondition failed  | 441 |
| <a href="#">FailureCharpolyCheck</a>   | 443 |
| <a href="#">FailureDetCheck</a>  | 443 |
| <a href="#">FailureFgemmCheck</a>  | 443 |
| <a href="#">FailureInvertCheck</a>   | 443 |
| <a href="#">FailurePLUQCheck</a>   | 444 |
| <a href="#">FailureTrsmCheck</a>   | 444 |

|  |     |
|--|-----|
| FieldSimd< _Field > . . . . .  | 444 |
| FieldTraits< Field >   |     |
| FieldTrait . . . . .   | 450 |
| FieldTraits< FFPACK::RNSInteger< T > > . . . . .   | 451 |
| FieldTraits< FFPACK::RNSIntegerMod< T > > . . . . .  | 451 |
| FieldTraits< Givaro::Modular< Element > > . . . . .  | 452 |
| FieldTraits< Givaro::ModularBalanced< Element > > . . . . .                                    | 452 |
| FieldTraits< Givaro::ZRing< double > > . . . . .   | 453 |
| FieldTraits< Givaro::ZRing< float > > . . . . .  | 453 |
| FieldTraits< Givaro::ZRing< Givaro::Integer > > . . . . .                                      | 454 |
| FieldTraits< Givaro::ZRing< int16_t > > . . . . .  | 455 |
| FieldTraits< Givaro::ZRing< int32_t > > . . . . .  | 455 |
| FieldTraits< Givaro::ZRing< int64_t > > . . . . .  | 456 |
| FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > > . . . . .                                   | 456 |
| FieldTraits< Givaro::ZRing< uint16_t > > . . . . .   | 457 |
| FieldTraits< Givaro::ZRing< uint32_t > > . . . . .   | 457 |
| FieldTraits< Givaro::ZRing< uint64_t > > . . . . .   | 458 |
| Fixed . . . . .  | 458 |
| FixedPrecIntTag  |     |
| Fixed precision integers above machine precision: Givaro::reclnt . . . . .                     | 458 |
| ForStrategy1D< blocksize_t, Cut, Param > . . . . .   | 459 |
| ForStrategy2D< blocksize_t, Cut, Param > . . . . .   | 461 |
| ftmmLeftLowerNoTransNonUnit< Element > . . . . .   | 464 |
| ftmmLeftLowerNoTransUnit< Element > . . . . .  | 464 |
| ftmmLeftLowerTransNonUnit< Element > . . . . .   | 465 |
| ftmmLeftLowerTransUnit< Element > . . . . .  | 465 |
| ftmmLeftUpperNoTransNonUnit< Element > . . . . .   | 465 |
| ftmmLeftUpperNoTransUnit< Element > . . . . .  | 465 |
| ftmmLeftUpperTransNonUnit< Element > . . . . .   | 465 |
| ftmmLeftUpperTransUnit< Element > . . . . .  | 465 |
| ftmmRightLowerNoTransNonUnit< Element > . . . . .  | 465 |
| ftmmRightLowerNoTransUnit< Element > . . . . .   | 465 |
| ftmmRightLowerTransNonUnit< Element > . . . . .  | 466 |
| ftmmRightLowerTransUnit< Element > . . . . .   | 466 |
| ftmmRightUpperNoTransNonUnit< Element > . . . . .  | 466 |
| ftmmRightUpperNoTransUnit< Element > . . . . .   | 466 |
| ftmmRightUpperTransNonUnit< Element > . . . . .  | 466 |
| ftmmRightUpperTransUnit< Element > . . . . .   | 466 |
| ftsmLeftLowerNoTransNonUnit< Element > . . . . .   | 466 |
| ftsmLeftLowerNoTransUnit< Element > . . . . .  | 466 |
| ftsmLeftLowerTransNonUnit< Element > . . . . .   | 467 |
| ftsmLeftLowerTransUnit< Element > . . . . .  | 467 |
| ftsmLeftUpperNoTransNonUnit< Element > . . . . .   |     |
| Computes the maximal size for delaying the modular reduction in a triangular system resolution | 467 |
| ftsmLeftUpperNoTransUnit< Element > . . . . .  | 467 |
| ftsmLeftUpperTransNonUnit< Element > . . . . .   | 467 |
| ftsmLeftUpperTransUnit< Element > . . . . .  | 467 |
| ftsmRightLowerNoTransNonUnit< Element > . . . . .  | 468 |
| ftsmRightLowerNoTransUnit< Element > . . . . .   | 468 |
| ftsmRightLowerTransNonUnit< Element > . . . . .  | 468 |
| ftsmRightLowerTransUnit< Element > . . . . .   | 468 |
| ftsmRightUpperNoTransNonUnit< Element > . . . . .  | 468 |
| ftsmRightUpperNoTransUnit< Element > . . . . .   | 468 |
| ftsmRightUpperTransNonUnit< Element > . . . . .  | 468 |
| ftsmRightUpperTransUnit< Element > . . . . .   | 468 |
| GenericTag   |     |
| Default is generic . . . . .   | 469 |

|   |     |
|---|-----|
| GenericTag  |     |
| Generic ring  | 469 |
| Grain   | 469 |
| has_minus_eq_impl< C >  | 469 |
| has_minus_impl< C >   | 469 |
| has_mul_eq_impl< C >  | 470 |
| has_mul_impl< C >   | 470 |
| has_operation< T >  | 470 |
| has_plus_eq_impl< C >   | 471 |
| has_plus_impl< C >  | 471 |
| HelperFlag  | 471 |
| HelperMod< Field, ElementTraits >   | 472 |
| HelperMod< Field, ElementCategories::MachineIntTag >                                      | 472 |
| HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >                         | 474 |
| HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >                             | 474 |
| HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >                             | 475 |
| Hybrid  | 476 |
| Info  | 476 |
| Info  | 477 |
| is_simd< T >  | 479 |
| isSparseMatrix< Field, M >  | 479 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >                             | 480 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >                          | 480 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >                             | 480 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >                         | 480 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >                          | 481 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >                             | 481 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >                        | 481 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >                     | 482 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >                          | 482 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >                          | 482 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >                            | 483 |
| isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >                         | 483 |
| isSparseMatrixMKLFormat< F, M >   | 483 |
| isSparseMatrixSimdFormat< F, M >  | 483 |
| isZOSparseMatrix< F, M >  | 484 |
| isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >                        | 484 |
| isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >                        | 484 |
| isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >                   | 485 |
| isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >                        | 485 |
| isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >                       | 485 |
| Iterative   | 485 |
| LazyTag   |     |
| Performs field operations with delayed mod only when necessary. Result may not be reduced | 486 |
| limits< T >   | 486 |
| limits< char >  | 486 |
| limits< double >  | 487 |
| limits< float >   | 487 |
| limits< Givaro::Integer >   | 488 |
| limits< int >   | 489 |
| limits< long >  | 489 |
| limits< long long >   | 490 |
| limits< Reclnt::rint< K > >   | 491 |
| limits< Reclnt::ruint< K > >  | 492 |
| limits< short int >   | 492 |
| limits< signed char >   | 493 |
| limits< unsigned char >   | 494 |
| limits< unsigned int >  | 494 |

|  |     |
|--|-----|
| limits< unsigned long > . . . . .  | 495 |
| limits< unsigned long long > . . . . .   | 496 |
| limits< unsigned short int > . . . . .   | 497 |
| MachineFloatTag  |     |
| Float or double . . . . .  | 497 |
| MachineIntTag  |     |
| Short, int, long, long long, and unsigned variants . . . . .   | 498 |
| MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > . . . . .   | 498 |
| MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > . . . . .                  | 503 |
| MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > . . . . .               | 505 |
| MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait > . . . . .                             | 507 |
| MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > . . . . . | 509 |
| MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > . . . . .                                    | 511 |
| FGEMM Helper for Default and ConvertTo modes of operation . . . . .  | 511 |
| ModeTraits< Field > . . . . .  | 513 |
| ModeTraits . . . . .   | 513 |
| ModeTraits< Givaro::Modular< Element, Compute > > . . . . .  | 513 |
| ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > . . . . .  | 513 |
| ModeTraits< Givaro::Modular< int16_t, Compute > > . . . . .  | 514 |
| ModeTraits< Givaro::Modular< int32_t, Compute > > . . . . .  | 514 |
| ModeTraits< Givaro::Modular< int8_t, Compute > > . . . . .   | 514 |
| ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > > . . . . .   | 515 |
| ModeTraits< Givaro::Modular< uint16_t, Compute > > . . . . .   | 515 |
| ModeTraits< Givaro::Modular< uint32_t, Compute > > . . . . .   | 516 |
| ModeTraits< Givaro::Modular< uint8_t, Compute > > . . . . .  | 516 |
| ModeTraits< Givaro::ModularBalanced< Element > > . . . . .   | 516 |
| ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > . . . . .   | 517 |
| ModeTraits< Givaro::ModularBalanced< int16_t > > . . . . .   | 517 |
| ModeTraits< Givaro::ModularBalanced< int32_t > > . . . . .   | 517 |
| ModeTraits< Givaro::ModularBalanced< int8_t > > . . . . .  | 518 |
| ModeTraits< Givaro::Montgomery< T > > . . . . .  | 518 |
| ModeTraits< Givaro::ZRing< double > > . . . . .  | 518 |
| ModeTraits< Givaro::ZRing< float > > . . . . .   | 519 |
| ModeTraits< Givaro::ZRing< Givaro::Integer > > . . . . .   | 519 |
| ModularBalanced< T > . . . . .   | 519 |
| ModularTag   |     |
| This is a modular field like e.g. Modular<T> or ModularBalanced<T> . . . . .                                       | 520 |
| Montgomery< T > . . . . .  | 520 |
| need_field_characteristic< Field > . . . . .   | 520 |
| need_field_characteristic< Givaro::Modular< Field > > . . . . .  | 520 |
| need_field_characteristic< Givaro::ModularBalanced< Field > > . . . . .  | 520 |
| NoSimd< T > . . . . .  | 521 |
| Parallel< C, P > . . . . .   | 522 |
| RNSInteger< RNS >::RandIter . . . . .  | 523 |
| RNSIntegerMod< RNS >::RandIter . . . . .   | 524 |
| readMyMachineType< Field, T > . . . . .  | 525 |
| readMyMachineType< Field, mpz_t > . . . . .  | 526 |
| Recursive . . . . .  | 527 |
| Recursive . . . . .  | 527 |
| rint< K > . . . . .  | 527 |
| rns_double . . . . .   | 527 |
| rns_double_elt . . . . .   | 532 |
| rns_double_elt_cstptr . . . . .  | 534 |
| rns_double_elt_ptr . . . . .   | 537 |
| rns_double_extended . . . . .  | 540 |
| RNSElementTag  |     |
| Representation in a Residue Number System . . . . .  | 544 |

|   |     |
|---|-----|
| RNSInteger< RNS >   | 545 |
| RNSIntegerMod< RNS >  | 548 |
| rnsRandIter< RNS >  | 555 |
| Row   | 556 |
| ruInt< K >  | 556 |
| ScalFunctions< Element, Enable >  | 556 |
| ScalFunctions< Element, typename enable_if< is_floating_point< Element >::value >::type > | 557 |
| ScalFunctions< Element, typename enable_if< is_integral< Element >::value >::type >       | 560 |
| Sequential  | 566 |
| Simd128_impl< ArithType, Int, Signed, Size >  | 567 |
| Simd128_impl< true, false, true, 4 >  | 567 |
| Simd128_impl< true, false, true, 8 >  | 567 |
| Simd128_impl< true, true, false, 2 >  | 568 |
| Simd128_impl< true, true, false, 4 >  | 577 |
| Simd128_impl< true, true, false, 8 >  | 587 |
| Simd128_impl< true, true, true, 2 >   | 597 |
| Simd128_impl< true, true, true, 4 >   | 606 |
| Simd128_impl< true, true, true, 8 >   | 615 |
| Simd128fp_base  | 623 |
| Simd128i_base   | 624 |
| Simd256_impl< ArithType, Int, Signed, Size >  | 626 |
| Simd256_impl< true, false, true, 4 >  | 626 |
| Simd256_impl< true, false, true, 8 >  | 626 |
| Simd256_impl< true, true, false, 2 >  | 633 |
| Simd256_impl< true, true, false, 4 >  | 643 |
| Simd256_impl< true, true, false, 8 >  | 661 |
| Simd256_impl< true, true, true, 2 >   | 671 |
| Simd256_impl< true, true, true, 4 >   | 679 |
| Simd256_impl< true, true, true, 8 >   | 696 |
| Simd256fp_base  | 704 |
| Simd256i_base   | 705 |
| Simd512_impl< ArithType, Int, Signed, Size >  | 706 |
| Simd512_impl< true, false, true, 4 >  | 706 |
| Simd512_impl< true, false, true, 8 >  | 706 |
| Simd512_impl< true, true, false, 8 >  | 713 |
| Simd512_impl< true, true, true, 8 >   | 724 |
| Simd512fp_base  | 733 |
| Simd512i_base   | 734 |
| SimdChooser< T, bool, bool >  | 735 |
| SimdChooser< T, false, b >  | 735 |
| SimdChooser< T, true, false >   | 735 |
| SimdChooser< T, true, true >  | 736 |
| simdToType< T >   | 736 |
| Single  | 736 |
| Sparse< Field, SparseMatrix_t, IdxT, PtrT >   | 736 |
| Sparse< _Field, SparseMatrix_t::COO >   | 737 |
| Sparse< _Field, SparseMatrix_t::COO_ZO >  | 738 |
| Sparse< _Field, SparseMatrix_t::CSR >   | 740 |
| Sparse< _Field, SparseMatrix_t::CSR_HYB >   | 742 |
| Sparse< _Field, SparseMatrix_t::CSR_ZO >  | 743 |
| Sparse< _Field, SparseMatrix_t::ELL >   | 745 |
| Sparse< _Field, SparseMatrix_t::ELL_simd >  | 747 |
| Sparse< _Field, SparseMatrix_t::ELL_simd_ZO >   | 748 |
| Sparse< _Field, SparseMatrix_t::ELL_ZO >  | 750 |
| Sparse< _Field, SparseMatrix_t::HYB_ZO >  | 752 |
| Sparse< _Field, SparseMatrix_t::SELL >  | 753 |
| Sparse< _Field, SparseMatrix_t::SELL_ZO >   | 755 |
| SpMat< Field, flag >  | 757 |

|   |     |
|---|-----|
| Static_error_check< bool >                              | 758 |
| Static_error_check< false >                             | 758 |
| StatsMatrix   | 758 |
| support_fast_mod< T >                                   | 762 |
| support_fast_mod< double >                              | 762 |
| support_fast_mod< float >                               | 762 |
| support_fast_mod< int64_t >                             | 762 |
| support_simd< T >                                       | 763 |
| support_simd_add< T >                                   | 763 |
| support_simd_mod< T >                                   | 763 |
| tfn_minus   | 764 |
| tfn_minus_eq  | 764 |
| tfn_mul   | 764 |
| tfn_mul_eq  | 765 |
| tfn_plus  | 765 |
| tfn_plus_eq   | 765 |
| Threads   | 766 |
| ThreeD  | 766 |
| ThreeDAdaptive  | 766 |
| ThreeDInPlace   | 766 |
| TRSMHelper< ReclterTrait, ParSeqTrait >                 |     |
| TRSM Helper   | 766 |
| TwoD  | 768 |
| TwoDAdaptive  | 768 |
| UnparametricTag   |     |
| If the field uses a representation with infix operators | 768 |
| Winograd  | 768 |
| WinogradPar   | 768 |





# Chapter 13

## File Index

### 13.1 File List

Here is a list of all files with brief descriptions:

|   |     |
|---|-----|
| <a href="#">arithprog.C</a>             | 769 |
| <a href="#">autotune/charpoly.C</a>     | 770 |
| <a href="#">examples/charpoly.C</a>     | 771 |
| <a href="#">fsyrk.C</a>                 | 771 |
| <a href="#">fsytrf.C</a>                | 772 |
| <a href="#">ftrtri.C</a>                | 773 |
| <a href="#">autotune/pluq.C</a>         | 774 |
| <a href="#">examples/pluq.C</a>         | 775 |
| <a href="#">winograd.C</a>              | 776 |
| <a href="#">benchmark-charpoly-mp.C</a> | 777 |
| <a href="#">benchmark-charpoly.C</a>    | 778 |
| <a href="#">benchmark-checkers.C</a>    | 779 |
| <a href="#">benchmark-dgemm.C</a>       | 780 |
| <a href="#">benchmark-dgetrf.C</a>      | 781 |
| <a href="#">benchmark-dgetri.C</a>      | 781 |
| <a href="#">benchmark-dsytrf.C</a>      | 782 |
| <a href="#">benchmark-dtrsm.C</a>       | 783 |
| <a href="#">benchmark-dtrtri.C</a>      | 784 |
| <a href="#">benchmark-fadd-lvl2.C</a>   | 785 |
| <a href="#">benchmark-fdot.C</a>        | 785 |
| <a href="#">benchmark-fgemm-mp.C</a>    | 786 |
| <a href="#">benchmark-fgemm-rns.C</a>   | 787 |
| <a href="#">benchmark-fgemm.C</a>       | 789 |
| <a href="#">benchmark-fgemv-mp.C</a>    | 790 |
| <a href="#">benchmark-fgemv.C</a>       | 791 |
| <a href="#">benchmark-fgesv.C</a>       | 794 |
| <a href="#">benchmark-fsyrk.C</a>       | 795 |
| <a href="#">benchmark-fsytrf.C</a>      | 796 |
| <a href="#">benchmark-ftrsm-mp.C</a>    | 797 |
| <a href="#">benchmark-ftrsm.C</a>       | 797 |
| <a href="#">benchmark-ftrsv.C</a>       | 798 |
| <a href="#">benchmark-ftrtri.C</a>      | 799 |
| <a href="#">benchmark-inverse.C</a>     | 799 |
| <a href="#">benchmark-lqup-mp.C</a>     | 800 |
| <a href="#">benchmark-lqup.C</a>        | 801 |

|  |     |
|--|-----|
| benchmark-pluq.C   | 801 |
| benchmark-wino.C   | 803 |
| config.h   | 804 |
| fflas-ffpack/config.h  | 808 |
| mainpage.doxy  | 812 |
| det.C  | 812 |
| matmul.C   | 812 |
| rank.C   | 813 |
| solve.C  | 813 |
| checker_charpoly.inl   | 813 |
| checker_det.inl  | 814 |
| checker_empty.h  | 814 |
| checker_fgemm.inl  | 815 |
| checker_ftrsm.inl  | 815 |
| checker_invert.inl   | 815 |
| checker_pluq.inl   | 816 |
| checkers.doxy  | 816 |
| checkers_fflas.h   | 816 |
| checkers_fflas.inl   | 817 |
| checkers_ffpack.h  | 817 |
| checkers_ffpack.inl  | 818 |
| config-blas.h  | 819 |
| fflas-ffpack-config.h  |     |
| Defaults for optimised values  | 825 |
| fflas-ffpack-default-thresholds.h  | 826 |
| fflas-ffpack-thresholds.h  | 827 |
| fflas-ffpack.doxy  | 827 |
| fflas-ffpack.h   |     |
| Includes <b>FFLAS</b> and <b>FFPACK</b>  | 827 |
| fflas.doxy   | 828 |
| fflas.h  |     |
| <b>Finite Field Linear Algebra Subroutines</b>   | 828 |
| fflas_bounds.inl   | 829 |
| fflas_enum.h   | 830 |
| fflas_fadd.h   | 831 |
| fflas_fadd.inl   | 832 |
| fflas_fassign.h  | 833 |
| fflas_fassign.inl  | 833 |
| fflas_faxpy.inl  | 834 |
| fflas_fdot.inl   | 835 |
| fflas_fgemm.inl  | 836 |
| fgemm_classical.inl  | 838 |
| fgemm_classical_mp.inl   |     |
| Matrix multiplication with multiprecision input (either over $\mathbb{Z}$ or over $\mathbb{Z}/p\mathbb{Z}$ ) | 838 |
| fgemm_winograd.inl   | 840 |
| matmul.doxy  | 841 |
| schedule_bini.inl  |     |
| Bini implementation  | 842 |
| schedule_winograd.inl  | 842 |
| schedule_winograd_acc.inl  | 843 |
| schedule_winograd_acc_ip.inl   | 844 |
| schedule_winograd_ip.inl   | 844 |
| fflas_fgenv.inl  | 845 |
| fflas_fgenv_mp.inl   | 847 |
| fflas_fger.inl   | 847 |
| fflas_fger_mp.inl  | 849 |
| fflas_freduce.h  | 849 |
| fflas_freduce.inl  | 850 |

|   |     |
|---|-----|
| fflas_freduce_mp.inl  | 852 |
| fflas_freivalds.inl   | 852 |
| fflas_fscal.h   | 853 |
| fflas_fscal.inl   | 853 |
| fflas_fscal_mp.inl  | 855 |
| fflas_fsyr2k.inl  | 855 |
| fflas_fsyrk.inl   | 856 |
| fflas_ftrmm.inl   | 857 |
| fflas_ftrsm.inl   | 857 |
| fflas_ftrsm_mp.inl  |     |
| Triangular system with matrix right hand side over multiprecision domain (either over $\mathbb{Z}$ or over $\mathbb{Z}/p\mathbb{Z}$ ) | 858 |
| fflas_ftrsv.inl   | 859 |
| fflas_helpers.inl   | 859 |
| igemm.doxy  | 860 |
| igemm.h   | 860 |
| igemm.inl   | 861 |
| igemm_kernels.h   | 862 |
| igemm_kernels.inl   | 862 |
| igemm_tools.h   | 863 |
| igemm_tools.inl   | 863 |
| fflas_level1.inl  | 864 |
| fflas_level2.inl  | 866 |
| fflas_level3.inl  | 869 |
| fflas_pfgemm.inl  | 871 |
| fflas_pftsm.inl   | 872 |
| fflas_simd.h  | 873 |
| simd.doxy   | 875 |
| simd128.inl   | 875 |
| simd128_double.inl  | 875 |
| simd128_float.inl   | 876 |
| simd128_int16.inl   | 876 |
| simd128_int32.inl   | 876 |
| simd128_int64.inl   | 877 |
| simd256.inl   | 877 |
| simd256_double.inl  | 878 |
| simd256_float.inl   | 878 |
| simd256_int16.inl   | 879 |
| simd256_int32.inl   | 879 |
| simd256_int64.inl   | 879 |
| simd512.inl   | 880 |
| simd512_double.inl  | 880 |
| simd512_float.inl   | 881 |
| simd512_int32.inl   | 881 |
| simd512_int64.inl   | 882 |
| simd_modular.inl  | 882 |
| fflas_sparse.h  | 882 |
| fflas_sparse.inl  | 887 |
| coo.h   | 889 |
| coo_spm.inl   | 889 |
| coo_spmv.inl  | 890 |
| coo_utils.inl   | 891 |
| csr.h   | 892 |
| csr_pspm.inl  | 892 |
| csr_spmv.inl  | 893 |
| csr_spm.inl   | 894 |
| csr_spmv.inl  | 895 |
| csr_utils.inl   | 896 |

|  |     |
|--|-----|
| csr_hyb.h  | 896 |
| csr_hyb_pspmm.inl  | 897 |
| csr_hyb_pspmv.inl  | 898 |
| csr_hyb_spmmm.inl  | 898 |
| csr_hyb_spmv.inl   | 899 |
| csr_hyb_utils.inl  | 899 |
| ell.h  | 900 |
| ell_pspmm.inl  | 900 |
| ell_pspmv.inl  | 901 |
| ell_spmmm.inl  | 902 |
| ell_spmv.inl   | 903 |
| ell_utils.inl  | 904 |
| ell_simd.h   | 904 |
| ell_simd_pspmv.inl   | 905 |
| ell_simd_spmv.inl  | 906 |
| ell_simd_utils.inl   | 907 |
| hyb_zo.h   | 907 |
| hyb_zo_pspmm.inl   | 907 |
| hyb_zo_pspmv.inl   | 908 |
| hyb_zo_spmmm.inl   | 909 |
| hyb_zo_spmv.inl  | 909 |
| hyb_zo_utils.inl   | 910 |
| read_sparse.h  | 910 |
| sell.h   | 911 |
| sell_pspmv.inl   | 912 |
| sell_spmv.inl  | 912 |
| sell_utils.inl   | 913 |
| sparse_matrix_traits.h                                     | 914 |
| utils.h  | 915 |
| ffpack.dox   | 916 |
| ffpack.h   | 916 |
| Set of elimination based routines for dense linear algebra | 916 |
| ffpack.inl   | 924 |
| ffpack_charpoly.inl  | 925 |
| ffpack_charpoly_danilevski.inl                             | 926 |
| ffpack_charpoly_kgfast.inl                                 | 927 |
| ffpack_charpoly_kgfastgeneralized.inl                      | 927 |
| ffpack_charpoly_kglu.inl                                   | 928 |
| ffpack_charpoly_mp.inl                                     | 928 |
| ffpack_det_mp.inl  | 929 |
| ffpack_echelonforms.inl                                    | 930 |
| ffpack_fgesv.inl   | 931 |
| ffpack_fgetrs.inl  | 932 |
| ffpack_frobenius.inl                                       | 932 |
| ffpack_fsytrf.inl  | 933 |
| ffpack_ftrssyr2k.inl                                       | 934 |
| ffpack_ftrstr.inl  | 935 |
| ffpack_fttr.inl  | 935 |
| ffpack_invert.inl  | 936 |
| ffpack_krylovelim.inl                                      | 936 |
| ffpack_ludivine.inl  | 937 |
| ffpack_ludivine_mp.inl                                     | 938 |
| ffpack_minpoly.inl   | 938 |
| ffpack_permutation.inl                                     | 939 |
| ffpack_pluq.inl  | 941 |
| ffpack_pluq_mp.inl   | 942 |
| ffpack_ppluq.inl   | 943 |
| ffpack_rankprofiles.inl                                    | 944 |

|   |      |
|---|------|
| field-traits.h  |      |
| Field Traits  | 945  |
| field.doxy  | 947  |
| rns-double-elt.h  |      |
| Rns elt structure with double support   | 947  |
| rns-double-recint.inl   | 948  |
| rns-double.h  |      |
| Rns structure with double support   | 948  |
| rns-double.inl  | 949  |
| rns-integer-mod.h   |      |
| Representation of $\mathbb{Z}/p\mathbb{Z}$ using RNS representation (note: fixed precision) | 949  |
| rns-integer.h   |      |
| Representation of $\mathbb{Z}$ using RNS representation (note: fixed precision)             | 950  |
| rns.h   | 951  |
| rns.inl   | 951  |
| interfaces.doxy   | 951  |
| fflas_c.h   | 951  |
| fflas_L1_inst.C   | 964  |
| fflas_L1_inst.h   | 966  |
| fflas_L1_inst_implem.inl  | 967  |
| fflas_L2_inst.C   | 968  |
| fflas_L2_inst.h   | 969  |
| fflas_L2_inst_implem.inl  | 971  |
| fflas_L3_inst.C   | 972  |
| fflas_L3_inst.h   | 973  |
| fflas_L3_inst_implem.inl  | 975  |
| fflas_lv1.C   |      |
| C functions calls for level 1 <b>FFLAS</b> in fflas-c.h                                     | 976  |
| fflas_lv2.C   |      |
| C functions calls for level 2 <b>FFLAS</b> in fflas-c.h                                     | 980  |
| fflas_lv3.C   |      |
| C functions calls for level 3 <b>FFLAS</b> in fflas-c.h                                     | 986  |
| fflas_sparse.C  |      |
| C functions calls for level 1.5 and 2.5 <b>FFLAS</b> in fflas-c.h                           | 988  |
| ffpack.C  |      |
| C functions calls for <b>FFPACK</b> in ffpack-c.h   | 988  |
| ffpack_c.h  | 1009 |
| ffpack_inst.C   | 1031 |
| ffpack_inst.h   | 1032 |
| ffpack_inst_implem.inl  | 1033 |
| blockcuts.inl   | 1037 |
| fflas_plevel1.h   | 1038 |
| kaapi_routines.inl  | 1039 |
| parallel.h  | 1039 |
| pfgemm_variants.inl   | 1046 |
| pfgemv.inl  | 1047 |
| align-allocator.h   | 1047 |
| args-parser.h   | 1047 |
| bit_manipulation.h  | 1049 |
| cast.h  | 1050 |
| debug.h   |      |
| Various utilities for debugging   | 1050 |
| fflas_intrinsic.h   | 1051 |
| fflas_io.h  | 1051 |
| fflas_memory.h  | 1052 |
| fflas_randommatrix.h  | 1053 |
| flimits.h   | 1055 |
| Matio.h   | 1056 |

|                       |      |
|-----------------------|------|
| test-utils.h          | 1056 |
| timer.h               | 1057 |
| cblas.C               | 1057 |
| clapack.C             | 1058 |
| cuda.C                | 1059 |
| fblas.C               | 1059 |
| gmp.C                 | 1060 |
| instrset.h            | 1060 |
| instrset_detect.cpp   | 1063 |
| lapack.C              | 1064 |
| regression-check.C    | 1064 |
| test-charpoly-check.C | 1066 |
| test-charpoly.C       | 1066 |
| test-compressQ.C      | 1067 |
| test-det-check.C      | 1068 |
| test-det.C            | 1069 |
| test-echelon.C        | 1070 |
| test-fadd.C           | 1072 |
| test-fdot.C           | 1074 |
| test-fgemm-check.C    | 1075 |
| test-fgemm.C          | 1076 |
| test-fgemv.C          | 1079 |
| test-fger.C           | 1080 |
| test-fgesv.C          | 1082 |
| test-finit.C          | 1084 |
| test-fscal.C          | 1085 |
| test-fsyr2k.C         | 1086 |
| test-fsyrk.C          | 1087 |
| test-fsytrf.C         | 1089 |
| test-ftrmm.C          | 1090 |
| test-ftrmv.C          | 1092 |
| test-ftrsm-check.C    | 1093 |
| test-ftrsm.C          | 1094 |
| test-ftrssyr2k.C      | 1095 |
| test-ftrstr.C         | 1096 |
| test-ftrsv.C          | 1097 |
| test-ftrtri.C         | 1098 |
| test-interfaces-c.c   | 1100 |
| test-invert-check.C   | 1100 |
| test-io.C             | 1101 |
| test-lu.C             | 1102 |
| test-maxdelayeddim.C  | 1106 |
| test-minpoly.C        | 1107 |
| test-multifile1.C     | 1108 |
| test-multifile2.C     | 1108 |
| test-nullspace.C      | 1108 |
| test-permutations.C   | 1109 |
| test-pluq-check.C     | 1111 |
| test-rankprofiles.C   | 1111 |
| test-rpm.C            | 1112 |
| test-simd.C           | 1113 |
| test-solve.C          | 1117 |
| 101-fgemm.C           | 1118 |
| 2x2-fgemm.C           | 1118 |
| 2x2-ftrsv.C           | 1119 |
| 2x2-pluq.C            | 1119 |
| fflas-101_1.C         | 1119 |
| fflas-101_3.C         | 1120 |

---

|                                  |      |
|----------------------------------|------|
| <a href="#">fflas_101.C</a>      | 1120 |
| <a href="#">fflas_101_lvl1.C</a> | 1121 |
| <a href="#">ffpack-fgesv.C</a>   | 1121 |
| <a href="#">ffpack-solve.C</a>   | 1121 |





## Chapter 14

# Module Documentation

### 14.1 CHECKER

Class CHECKER provides functions to verify computations in [FFLAS](#) and [FFPACK](#).

Class CHECKER provides functions to verify computations in [FFLAS](#) and [FFPACK](#).

### 14.2 FFLAS-FFPACK

the [FFLAS FFPACK](#) library

#### Modules

- [FFLAS](#)  
*The C-style wrapper of BLAS for finite field linear algebra.*
- [Interfaces](#)  
*Interfaces for FFLAS-FFPACK.*

#### 14.2.1 Detailed Description

the [FFLAS FFPACK](#) library

C++ header library for fast exact dense linear algebra

See also

[FFLAS](#)  
[FFPACK](#)

## 14.3 FFLAS

The C-style wrapper of BLAS for finite field linear algebra.

The C-style wrapper of BLAS for finite field linear algebra.

[FFLAS](#), Finite Field Linear Algebra Subroutines, provide basic linear algebra subroutines based on the BLAS interface. Therefore, the specifications are in C style; only the field given as a template parameter requires C++.

As much as possible, these routines use `ATLAS/BLAS` computations and achieve therefore high efficiency.

## 14.4 Matrix Multiplication Algorithms

Matrix Multiplication (level 3) algorithms.

### Files

- file [schedule\\_bini.inl](#)  
*Bini implementation.*

### 14.4.1 Detailed Description

Matrix Multiplication (level 3) algorithms.

**Todo** biblio

## 14.5 SIMD wrapper

wraps SIMD functions Supportst SSE4.1, AVX, AVX2.

wraps SIMD functions Supportst SSE4.1, AVX, AVX2.

**Todo** biblio

## 14.6 FFPACK

Class [FFPACK](#) provides functions using fflas much as Lapack uses BLAS.

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### 14.6.1 Detailed Description

Class [FFPACK](#) provides functions using fflas much as Lapack uses BLAS.

## 14.7 FFLAS-FFPACK fields

fields in the FFLAS-FFPACK library

### Files

- file [rns-double-elt.h](#)  
*rns elt structure with double support*
- file [rns-double.h](#)  
*rns structure with double support*
- file [rns-integer-mod.h](#)  
*representation of  $\mathbb{Z}/p\mathbb{Z}$  using RNS representation (note: fixed precision)*
- file [rns-integer.h](#)  
*representation of  $\mathbb{Z}$  using RNS representation (note: fixed precision)*
- file [rns.h](#)

### 14.7.1 Detailed Description

fields in the FFLAS-FFPACK library

Unparametric/Random elements

[Todo](#) biblio

## 14.8 RNS

just include them all

just include them all

## 14.9 Interfaces

Intefaces for FFLAS-FFPACK.

Intefaces for FFLAS-FFPACK.

C interface in folder

See also

libs



## Chapter 15

# Namespace Documentation

### 15.1 FFLAS Namespace Reference

#### Namespaces

- namespace [BLAS3](#)
- namespace [csr\\_hyb\\_details](#)
- namespace [CuttingStrategy](#)
- namespace [details](#)
- namespace [details\\_spmv](#)
- namespace [ElementCategories](#)
- namespace [FieldCategories](#)

*Traits and categories will need to be placed in a proper file later.*

- namespace [MMHelperAlgo](#)
- namespace [ModeCategories](#)

*Specifies the mode of action for an algorithm w.r.t.*

- namespace [ParSeqHelper](#)

*ParSeqHelper for both fgemm and ftrsm.*

- namespace [Protected](#)
- namespace [sell\\_details](#)
- namespace [sparse\\_details](#)
- namespace [sparse\\_details\\_impl](#)
- namespace [StrategyParameter](#)
- namespace [StructureHelper](#)

*StructureHelper for ftrsm.*

- namespace [vectorised](#)

#### Data Structures

- struct [AlgoChooser](#)
- struct [AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >](#)
- struct [associatedDelayedField](#)
- struct [associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >](#)
- struct [associatedDelayedField< const Givaro::Modular< T, X > >](#)
- struct [associatedDelayedField< const Givaro::ModularBalanced< T > >](#)
- struct [associatedDelayedField< const Givaro::ZRing< T > >](#)

- struct [Checker\\_Empty](#)
- class [CheckerImplem\\_fgemm](#)
- class [CheckerImplem\\_ftsm](#)
- struct [CooMat](#)
- struct [CsrMat](#)
- struct [ElementTraits](#)
  - ElementTraits.*
- struct [ElementTraits< double >](#)
- struct [ElementTraits< FFPACK::rns\\_double\\_elt >](#)
- struct [ElementTraits< float >](#)
- struct [ElementTraits< Givaro::Integer >](#)
- struct [ElementTraits< int16\\_t >](#)
- struct [ElementTraits< int32\\_t >](#)
- struct [ElementTraits< int64\\_t >](#)
- struct [ElementTraits< int8\\_t >](#)
- struct [ElementTraits< Reclnt::rint< K > >](#)
- struct [ElementTraits< Reclnt::rmint< K, MG > >](#)
- struct [ElementTraits< Reclnt::ruint< K > >](#)
- struct [ElementTraits< uint16\\_t >](#)
- struct [ElementTraits< uint32\\_t >](#)
- struct [ElementTraits< uint64\\_t >](#)
- struct [ElementTraits< uint8\\_t >](#)
- struct [ElIMat](#)
- struct [FieldTraits](#)
  - FieldTrait.*
- struct [FieldTraits< FFPACK::RNSInteger< T > >](#)
- struct [FieldTraits< FFPACK::RNSIntegerMod< T > >](#)
- struct [FieldTraits< Givaro::Modular< Element > >](#)
- struct [FieldTraits< Givaro::ModularBalanced< Element > >](#)
- struct [FieldTraits< Givaro::ZRing< double > >](#)
- struct [FieldTraits< Givaro::ZRing< float > >](#)
- struct [FieldTraits< Givaro::ZRing< Givaro::Integer > >](#)
- struct [FieldTraits< Givaro::ZRing< int16\\_t > >](#)
- struct [FieldTraits< Givaro::ZRing< int32\\_t > >](#)
- struct [FieldTraits< Givaro::ZRing< int64\\_t > >](#)
- struct [FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > >](#)
- struct [FieldTraits< Givaro::ZRing< uint16\\_t > >](#)
- struct [FieldTraits< Givaro::ZRing< uint32\\_t > >](#)
- struct [FieldTraits< Givaro::ZRing< uint64\\_t > >](#)
- struct [ForStrategy1D](#)
- struct [ForStrategy2D](#)
- struct [has\\_minus\\_eq\\_impl](#)
- struct [has\\_minus\\_impl](#)
- struct [has\\_mul\\_eq\\_impl](#)
- struct [has\\_mul\\_impl](#)
- struct [has\\_operation](#)
- struct [has\\_plus\\_eq\\_impl](#)
- struct [has\\_plus\\_impl](#)
- struct [HelperFlag](#)
- struct [isSparseMatrix](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::COO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::COO\\_ZO > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::CSR > >](#)
- struct [isSparseMatrix< Field, Sparse< Field, SparseMatrix\\_t::CSR\\_HYB > >](#)

- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::SELL > >
- struct [isSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >
- struct [isSparseMatrixMKLFormat](#)
- struct [isSparseMatrixSimdFormat](#)
- struct [isZOSparseMatrix](#)
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >
- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >
- struct [MMHelper](#)
- struct [MMHelper](#)< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >
- struct [MMHelper](#)< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >
- struct [MMHelper](#)< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >
- struct [MMHelper](#)< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >
- struct [MMHelper](#)< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >

*FGEMM Helper for Default and ConvertTo modes of operation.*

- struct [ModeTraits](#)
  - ModeTraits.*
  - struct [ModeTraits](#)< Givaro::Modular< Element, Compute > >
  - struct [ModeTraits](#)< Givaro::Modular< Givaro::Integer, Compute > >
  - struct [ModeTraits](#)< Givaro::Modular< int16\_t, Compute > >
  - struct [ModeTraits](#)< Givaro::Modular< int32\_t, Compute > >
  - struct [ModeTraits](#)< Givaro::Modular< int8\_t, Compute > >
  - struct [ModeTraits](#)< Givaro::Modular< ReclInt::ruint< K >, Compute > >
  - struct [ModeTraits](#)< Givaro::Modular< uint16\_t, Compute > >
  - struct [ModeTraits](#)< Givaro::Modular< uint32\_t, Compute > >
  - struct [ModeTraits](#)< Givaro::Modular< uint8\_t, Compute > >
  - struct [ModeTraits](#)< Givaro::ModularBalanced< Element > >
  - struct [ModeTraits](#)< Givaro::ModularBalanced< Givaro::Integer > >
  - struct [ModeTraits](#)< Givaro::ModularBalanced< int16\_t > >
  - struct [ModeTraits](#)< Givaro::ModularBalanced< int32\_t > >
  - struct [ModeTraits](#)< Givaro::ModularBalanced< int8\_t > >
  - struct [ModeTraits](#)< Givaro::Montgomery< T > >
  - struct [ModeTraits](#)< Givaro::ZRing< double > >
  - struct [ModeTraits](#)< Givaro::ZRing< float > >
  - struct [ModeTraits](#)< Givaro::ZRing< Givaro::Integer > >
  - struct [readMyMachineType](#)
  - struct [readMyMachineType](#)< Field, mpz\_t >
  - struct [Sparse](#)
  - struct [Sparse](#)< \_Field, SparseMatrix\_t::COO >
  - struct [Sparse](#)< \_Field, SparseMatrix\_t::COO\_ZO >
  - struct [Sparse](#)< \_Field, SparseMatrix\_t::CSR >
  - struct [Sparse](#)< \_Field, SparseMatrix\_t::CSR\_HYB >
  - struct [Sparse](#)< \_Field, SparseMatrix\_t::CSR\_ZO >
  - struct [Sparse](#)< \_Field, SparseMatrix\_t::ELL >
  - struct [Sparse](#)< \_Field, SparseMatrix\_t::ELL\_simd >
  - struct [Sparse](#)< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >

- struct [Sparse<\\_Field, SparseMatrix\\_t::ELL\\_ZO >](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::HYB\\_ZO >](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::SELL >](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::SELL\\_ZO >](#)
- struct [SpMat](#)
- struct [StatsMatrix](#)
- struct [support\\_fast\\_mod](#)
- struct [support\\_fast\\_mod< double >](#)
- struct [support\\_fast\\_mod< float >](#)
- struct [support\\_fast\\_mod< int64\\_t >](#)
- struct [support\\_simd](#)
- struct [support\\_simd\\_add](#)
- struct [support\\_simd\\_mod](#)
- struct [tfn\\_minus](#)
- struct [tfn\\_minus\\_eq](#)
- struct [tfn\\_mul](#)
- struct [tfn\\_mul\\_eq](#)
- struct [tfn\\_plus](#)
- struct [tfn\\_plus\\_eq](#)
- struct [TRSMHelper](#)

*TRSM Helper.*

## Typedefs

- template<class [Field](#) >  
using [Checker\\_fgemm](#) = [FFLAS::Checker\\_Empty< Field >](#)
- template<class [Field](#) >  
using [Checker\\_ftrsm](#) = [FFLAS::Checker\\_Empty< Field >](#)
- template<class [Field](#) >  
using [ForceCheck\\_fgemm](#) = [CheckerImplem\\_fgemm< Field >](#)
- template<class [Field](#) >  
using [ForceCheck\\_ftrsm](#) = [CheckerImplem\\_ftrsm< Field >](#)
- using [ZOSparseMatrix](#) = std::true\_type
- using [NotZOSparseMatrix](#) = std::false\_type
- using [SimdSparseMatrix](#) = std::true\_type
- using [NoSimdSparseMatrix](#) = std::false\_type
- using [MKLSparseMatrixFormat](#) = std::true\_type
- using [NotMKLSparseMatrixFormat](#) = std::false\_type
- template<class T >  
using [has\\_plus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_plus\\_impl< T >](#) >::type
- template<class T >  
using [has\\_minus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_minus\\_impl< T >](#) >::type
- template<class T >  
using [has\\_equal](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, std::is\_copyable & assignable< T > >::type
- template<class T >  
using [has\\_plus\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_plus\\_eq\\_impl< T >](#) >::type
- template<class T >  
using [has\\_minus\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, [has\\_minus\\_eq\\_impl< T >](#) >::type



- `template<class T >`  
using `has_mul` = `typename std::conditional< std::is_arithmetic< T >::value, std::true_type, has_mul_impl< T > >::type`
- `template<class T >`  
using `has_mul_eq` = `typename std::conditional< std::is_arithmetic< T >::value, std::true_type, has_mul_eq_impl< T > >::type`
- `typedef Givaro::Timer` `Timer`
- `typedef Givaro::BaseTimer` `BaseTimer`
- `typedef Givaro::UserTimer` `UserTimer`
- `typedef Givaro::SysTimer` `SysTimer`

## Enumerations

- enum `FFLAS_ORDER` { `FflasRowMajor` = 101 , `FflasColMajor` = 102 }  
*Storage by row or col ?*
- enum `FFLAS_TRANSPOSE` { `FflasNoTrans` = 111 , `FflasTrans` = 112 }  
*Is matrix transposed ?*
- enum `FFLAS_UPLO` { `FflasUpper` = 121 , `FflasLower` = 122 }  
*Is triangular matrix's shape upper ?*
- enum `FFLAS_DIAG` { `FflasNonUnit` = 131 , `FflasUnit` = 132 }  
*Is the triangular matrix implicitly unit diagonal ?*
- enum `FFLAS_SIDE` { `FflasLeft` = 141 , `FflasRight` = 142 }  
*On what side ?*
- enum `FFLAS_BASE` { `FflasDouble` = 151 , `FflasFloat` = 152 , `FflasGeneric` = 153 }  
*FFLAS\_BASE determines the type of the element representation for Matrix Mult kernel.*
- enum `number_kind` { `zero` = 0 , `one` = 1 , `mone` = -1 , `other` = 2 }
- enum class `SparseMatrix_t` {  
    `CSR` , `CSR_ZO` , `CSC` , `CSC_ZO` ,  
    `COO` , `COO_ZO` , `ELL` , `ELL_ZO` ,  
    `SELL` , `SELL_ZO` , `ELL_simd` , `ELL_simd_ZO` ,  
    `CSR_HYB` , `HYB_ZO` }
- enum `FFLAS_FORMAT` {  
    `FflasAuto` = 0 , `FflasDense` = 1 , `FflasSMS` = 2 , `FflasBinary` = 3 ,  
    `FflasMath` = 4 , `FflasMaple` = 5 , `FflasSageMath` = 6 }

## Functions

- `Givaro::Integer` `InfNorm` (const `size_t` M, const `size_t` N, const `Givaro::Integer` \*A, const `size_t` lda)
- `template<class T >`  
const `T` & `min3` (const `T` &m, const `T` &n, const `T` &k)
- `template<class T >`  
const `T` & `max3` (const `T` &m, const `T` &n, const `T` &k)
- `template<class T >`  
const `T` & `min4` (const `T` &m, const `T` &n, const `T` &k, const `T` &l)
- `template<class T >`  
const `T` & `max4` (const `T` &m, const `T` &n, const `T` &k, const `T` &l)
- `template<class Field >`  
void `fadd` (const `Field` &F, const `size_t` N, `typename Field::ConstElement_ptr` A, const `size_t` inca, `typename Field::ConstElement_ptr` B, const `size_t` incb, `typename Field::Element_ptr` C, const `size_t` incc)
- `template<class Field >`  
void `faddin` (const `Field` &F, const `size_t` N, `typename Field::ConstElement_ptr` B, const `size_t` incb, `typename Field::Element_ptr` C, const `size_t` incc)

- `template<class Field >`  
`void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition.*
- `template<class Field >`  
`void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsub : matrix subtraction.*
- `template<class Field >`  
`void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*faddin*
- `template<class Field >`  
`void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fsubin  $C = C - B$*
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition with scaling.*
- `template<class Field >`  
`void fassign (const Field &F, const size_t N, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  
*fassign :  $x \leftarrow y$ .*
- `template<> void fassign (const Givaro::Modular< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`

- `template<> void fassign (const Givaro::Modular< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<class Field >`  
`void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  

$$fassign : A \leftarrow B.$$
- `template<class Field >`  
`void faxpy (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<> void faxpy (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void faxpy (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<class Field >`  
`void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t ldx, typename Field::Element_ptr Y, const size_t ldy)`  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DelayedTag &MT)`
- `template<> Givaro::DoubleDomain::Element fdot (const Givaro::DoubleDomain &, const size_t N, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<> Givaro::FloatDomain::Element fdot (const Givaro::FloatDomain &, const size_t N, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultTag &MT)`
- `template<class Field, class T >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::ConvertTo< T > &MT)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, ModeCategories::DefaultBoundedTag &dbt)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, const ParSeqHelper::Sequential seq)`
- `template<class Field >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)`  

$$fdot: \text{dot product } x^T y.$$
- `template<class Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >, ParSeqHelper::Sequential > &H)`

- `template<typename Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq)`
- `template<typename Field , class Cut , class Param >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par)`
- `template<typename Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*fgemm: Field GEneral Matrix Multiply.*
- `template<typename Field , class ModeT , class ParSeq >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Auto, ModeT, ParSeq > &H)`
- `template<class Field >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)`
- `template<class Field >`  
`Field::Element_ptr fsquare (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*fsquare: Squares a matrix.*
- `template<> double * fsquare (const Givaro::ModularBalanced< double > &F, const FFLAS_TRANSPOSE ta, const size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const size_t ldc)`
- `template<> float * fsquare (const Givaro::ModularBalanced< float > &F, const FFLAS_TRANSPOSE ta, const size_t n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)`
- `template<> double * fsquare (const Givaro::Modular< double > &F, const FFLAS_TRANSPOSE ta, const size_t n, const double alpha, const double *A, const size_t lda, const double beta, double *C, const size_t ldc)`
- `template<> float * fsquare (const Givaro::Modular< float > &F, const FFLAS_TRANSPOSE ta, const size_t n, const float alpha, const float *A, const size_t lda, const float beta, float *C, const size_t ldc)`
- `template<typename RNS , typename ParSeqTrait >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > &H)`
- `template<typename RNS >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k,`

- const typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element alpha, typename [FFPACK::RNSInteger](#)< [RNS](#) >::ConstElement\_ptr Ad, const size\_t lda, typename [FFPACK::RNSInteger](#)< [RNS](#) >::ConstElement\_ptr Bd, const size\_t ldb, const typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element beta, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr Cd, const size\_t ldc, [MMHelper](#)< [FFPACK::RNSInteger](#)< [RNS](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#), [ParSeqHelper::Sequential](#) > &H)
- template<typename [RNS](#) , typename [ParSeqTrait](#) >  
[FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr fgemm (const [FFPACK::RNSInteger](#)< [RNS](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element alpha, typename [FFPACK::RNSInteger](#)< [RNS](#) >::ConstElement\_ptr Ad, const size\_t lda, typename [FFPACK::RNSInteger](#)< [RNS](#) >::ConstElement\_ptr Bd, const size\_t ldb, const typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element beta, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr Cd, const size\_t ldc, [MMHelper](#)< [FFPACK::RNSInteger](#)< [RNS](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#), [ParSeqHelper::Compose](#)< [ParSeqHelper::Parallel](#)< [CuttingStrategy::RNSModulus](#), [StrategyParameter::Threads](#) >, [ParSeqTrait](#) > > &H)
  - template<typename [RNS](#) , typename [Cut](#) , typename [Param](#) >  
[FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr fgemm (const [FFPACK::RNSInteger](#)< [RNS](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element alpha, typename [FFPACK::RNSInteger](#)< [RNS](#) >::ConstElement\_ptr Ad, const size\_t lda, typename [FFPACK::RNSInteger](#)< [RNS](#) >::ConstElement\_ptr Bd, const size\_t ldb, const typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element beta, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr Cd, const size\_t ldc, [MMHelper](#)< [FFPACK::RNSInteger](#)< [RNS](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#), [ParSeqHelper::Parallel](#)< [Cut](#), [Param](#) > > &H)
  - template<class [ParSeq](#) >  
[Givaro::Integer](#) \* fgemm (const [Givaro::ZRing](#)< [Givaro::Integer](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const [Givaro::Integer](#) alpha, const [Givaro::Integer](#) \*A, const size\_t lda, const [Givaro::Integer](#) \*B, const size\_t ldb, [Givaro::Integer](#) beta, [Givaro::Integer](#) \*C, const size\_t ldc, [MMHelper](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) >, [ParSeq](#) > &H)
  - template<typename [RNS](#) , class [ModeT](#) >  
[RNS::Element\\_ptr](#) fgemm (const [FFPACK::RNSInteger](#)< [RNS](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const typename [RNS::Element](#) alpha, typename [RNS::ConstElement\\_ptr](#) Ad, const size\_t lda, typename [RNS::ConstElement\\_ptr](#) Bd, const size\_t ldb, const typename [RNS::Element](#) beta, typename [RNS::Element\\_ptr](#) Cd, const size\_t ldc, [MMHelper](#)< [FFPACK::RNSInteger](#)< [RNS](#) >, [MMHelperAlgo::Winograd](#), [ModeT](#), [ParSeqHelper::Sequential](#) > &H)
  - template<typename [RNS](#) >  
[RNS::Element\\_ptr](#) fgemm (const [FFPACK::RNSIntegerMod](#)< [RNS](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const typename [RNS::Element](#) alpha, typename [RNS::ConstElement\\_ptr](#) Ad, const size\_t lda, typename [RNS::ConstElement\\_ptr](#) Bd, const size\_t ldb, const typename [RNS::Element](#) beta, typename [RNS::Element\\_ptr](#) Cd, const size\_t ldc, [MMHelper](#)< [FFPACK::RNSIntegerMod](#)< [RNS](#) >, [MMHelperAlgo::Winograd](#) > &H)
  - [Givaro::Integer](#) \* fgemm (const [Givaro::Modular](#)< [Givaro::Integer](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const [Givaro::Integer](#) alpha, const [Givaro::Integer](#) \*A, const size\_t lda, const [Givaro::Integer](#) \*B, const size\_t ldb, const [Givaro::Integer](#) beta, [Givaro::Integer](#) \*C, const size\_t ldc, [MMHelper](#)< [Givaro::Modular](#)< [Givaro::Integer](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) > > &H)
  - template<class [ParSeq](#) >  
[Givaro::Integer](#) \* fgemm (const [Givaro::Modular](#)< [Givaro::Integer](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const [Givaro::Integer](#) alpha, const [Givaro::Integer](#) \*A, const size\_t lda, const [Givaro::Integer](#) \*B, const size\_t ldb, const [Givaro::Integer](#) beta, [Givaro::Integer](#) \*C, const size\_t ldc, [MMHelper](#)< [Givaro::Modular](#)< [Givaro::Integer](#) >, [MMHelperAlgo::Auto](#), [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) >, [ParSeq](#) > &H)
  - template<size\_t K1, size\_t K2, class [ParSeq](#) >  
[Reclnt::ruint](#)< K1 > \* fgemm (const [Givaro::Modular](#)< [Reclnt::ruint](#)< K1 >, [Reclnt::ruint](#)< K2 > > &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const [Reclnt::ruint](#)< K1 > alpha, const [Reclnt::ruint](#)< K1 > \*A, const size\_t lda, const [Reclnt::ruint](#)< K1 > \*B, const size\_t ldb, [Reclnt::ruint](#)< K1 > beta, [Reclnt::ruint](#)< K1 > \*C, const size\_t

- ldc, MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)
- template<class Field, class ModeT >  
Field::Element\_ptr fgemm (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeT > &H)
  - template<class Field, class ModeT, class Cut, class Param >  
Field::Element\_ptr fgemm (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, MMHelper< Field, MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut, Param > > &H)
  - template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)
  - template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)
  - template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
  - template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)
  - template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE TransA, const size\_t M, const size\_t N, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, typename Field::Element\_ptr Y, const size\_t incY)
- finite prime Field GEneral Matrix Vector multiplication.*
- Givaro::ZRing< int64\_t >::Element\_ptr fgemv (const Givaro::ZRing< int64\_t > &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const int64\_t alpha, const int64\_t \*A, const size\_t lda, const int64\_t \*X, const size\_t incX, const int64\_t beta, const int64\_t \*Y, const size\_t incY, MMHelper< Givaro::ZRing< int64\_t >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
  - Givaro::DoubleDomain::Element\_ptr fgemv (const Givaro::DoubleDomain &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement\_ptr A, const size\_t lda, const Givaro::DoubleDomain::ConstElement\_ptr X, const size\_t incX, const Givaro::DoubleDomain::Element beta, const Givaro::DoubleDomain::Element\_ptr Y, const size\_t incY, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
  - template<class Field >  
Field::Element\_ptr fgemv (const Field &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t N, const typename Field::Element alpha, const typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::ConstElement\_ptr X, const size\_t incX, const typename Field::Element beta, const typename Field::Element\_ptr Y, const size\_t incY)



- beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::DefaultBoundedTag](#) > &H)
- [Givaro::FloatDomain::Element\\_ptr fgemv](#) (const [Givaro::FloatDomain](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const [Givaro::FloatDomain::Element](#) alpha, const [Givaro::FloatDomain::ConstElement\\_ptr](#) A, const size\_t lda, const [Givaro::FloatDomain::ConstElement\\_ptr](#) X, const size\_t incX, const [Givaro::FloatDomain::Element](#) beta, [Givaro::FloatDomain::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [Givaro::FloatDomain](#), [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
- [template<class Field, class Cut, class Param >](#)  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, [ParSeqHelper::Parallel](#)< Cut, Param > &parH)
- [template<class Field >](#)  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, [ParSeqHelper::Sequential](#) &seqH)
- [FFPACK::rns\\_double::Element\\_ptr fgemv](#) (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::ConstElement\\_ptr](#) X, const size\_t incX, const [FFPACK::rns\\_double::Element](#) beta, [FFPACK::rns\\_double::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
- [FFPACK::rns\\_double::Element\\_ptr fgemv](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::ConstElement\\_ptr](#) X, const size\_t incX, const [FFPACK::rns\\_double::Element](#) beta, [FFPACK::rns\\_double::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
- [Givaro::Integer \\* fgemv](#) (const [Givaro::ZRing](#)< [Givaro::Integer](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const [Givaro::Integer](#) alpha, [Givaro::Integer \\*A](#), const size\_t lda, [Givaro::Integer \\*X](#), const size\_t ldx, [Givaro::Integer](#) beta, [Givaro::Integer \\*Y](#), const size\_t ldy, [MMHelper](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) > &H)
- [Givaro::Integer \\* fgemv](#) (const [Givaro::Modular](#)< [Givaro::Integer](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const [Givaro::Integer](#) alpha, [Givaro::Integer \\*A](#), const size\_t lda, [Givaro::Integer \\*X](#), const size\_t ldx, [Givaro::Integer](#) beta, [Givaro::Integer \\*Y](#), const size\_t ldy, [MMHelper](#)< [Givaro::Modular](#)< [Givaro::Integer](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) > &H)
- [template<size\\_t K1, size\\_t K2, class ParSeq >](#)  
[RecInt::ruint](#)< K1 > \* [fgemv](#) (const [Givaro::Modular](#)< [RecInt::ruint](#)< K1 >, [RecInt::ruint](#)< K2 > > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const [RecInt::ruint](#)< K1 > alpha, const [RecInt::ruint](#)< K1 > \*A, const size\_t lda, const [RecInt::ruint](#)< K1 > \*X, const size\_t incx, [RecInt::ruint](#)< K1 > beta, [RecInt::ruint](#)< K1 > \*Y, const size\_t incy, [MMHelper](#)< [Givaro::Modular](#)< [RecInt::ruint](#)< K1 >, [RecInt::ruint](#)< K2 > >, [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) >, ParSeq > &H)
- [template<class Field >](#)  
void [fger](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*fger: rank one update of a general matrix*
- [template<class Field >](#)  
void [fger](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, typename [Field::Element\\_ptr](#) A, const size\_t lda, [MMHelper](#)< [Field](#), [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) > > &H)
- [template<class Field, class AnyTag >](#)  
void [fger](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename

- `Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, AnyTag > &H)`
- `void fger (const Givaro::DoubleDomain &F, const size_t M, const size_t N, const Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, const Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, Givaro::DoubleDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
  - `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, const typename Field::ConstElement_ptr x, const size_t incx, const typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)`
  - `void fger (const Givaro::FloatDomain &F, const size_t M, const size_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, const Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, Givaro::FloatDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
  - `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)`
  - `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)`
  - `void fger (const Givaro::Modular< Givaro::Integer > &F, const size_t M, const size_t N, const typename Givaro::Integer alpha, typename Givaro::Integer *x, const size_t incx, typename Givaro::Integer *y, const size_t incy, typename Givaro::Integer *A, const size_t lda, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)`
  - `template<typename RNS >`  
`void fger (const FFPACK::RNSInteger< RNS > &F, const size_t M, const size_t N, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::Element_ptr x, const size_t incx, typename FFPACK::RNSInteger< RNS >::Element_ptr y, const size_t incy, typename FFPACK::RNSInteger< RNS >::Element_ptr A, const size_t lda, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
  - `template<typename RNS >`  
`void fger (const FFPACK::RNSIntegerMod< RNS > &F, const size_t M, const size_t N, const typename FFPACK::RNSIntegerMod< RNS >::Element alpha, typename FFPACK::RNSIntegerMod< RNS >::Element_ptr x, const size_t incx, typename FFPACK::RNSIntegerMod< RNS >::Element_ptr y, const size_t incy, typename FFPACK::RNSIntegerMod< RNS >::Element_ptr A, const size_t lda, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Classic > &H)`
  - `template<class Field >`  
`void freduce (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  

$$\text{freduce } x \leftarrow y \bmod F.$$
  - `template<class Field >`  
`void freduce (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$\text{freduce } x \leftarrow x \bmod F.$$
  - `template<class Field >`  
`void freduce_constoverride (const Field &F, const size_t m, typename Field::ConstElement_ptr A, const size_t incX)`
  - `template<class Field, class ConstOtherElement_ptr >`  
`void finit (const Field &F, const size_t n, ConstOtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`
  - `template<class Field >`  
`void finit (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$\text{finit Initializes } X \text{ in } F\mathbb{Z}.$$



- template<class [Field](#) >  
void [freduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $freduce\ A \leftarrow A mod F.$
- template<class [Field](#) >  
void [pfreduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t numths)
- template<class [Field](#) >  
void [freduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $freduce\ A \leftarrow B mod F.$
- template<class [Field](#) >  
void [freduce\\_constoverride](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)
- template<class [Field](#), class OtherElement\_ptr >  
void [finit](#) (const [Field](#) &F, const size\_t m, const size\_t n, const OtherElement\_ptr B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $finit\ A \leftarrow B mod F.$
- template<class [Field](#) >  
void [finit](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)
- template<> void [freduce](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >::Element\_ptr A, size\_t inc)
- template<> void [freduce](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, [FFPACK::rns\\_double::Element\\_ptr](#) A, size\_t lda)
- template<class [Field](#) >  
bool [freivalds](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::ConstElement\\_ptr](#) C, const size\_t ldc)  
*freivalds: Freivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.*
- template<class [Field](#) >  
void [fscal](#) (const [Field](#) &F, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
 $fscal\ x \leftarrow \alpha \cdot x.$
- template<class [Field](#) >  
void [fscal](#) (const [Field](#) &F, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  
 $fscal\ y \leftarrow \alpha \cdot x.$
- template<> void [fscal](#) (const Givaro::DoubleDomain &, const size\_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::ConstElement\_ptr x, const size\_t incx, Givaro::DoubleDomain::Element\_ptr y, const size\_t incy)
- template<> void [fscal](#) (const Givaro::FloatDomain &, const size\_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement\_ptr x, const size\_t incx, Givaro::FloatDomain::Element\_ptr y, const size\_t incy)
- template<> void [fscal](#) (const Givaro::DoubleDomain &, const size\_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::Element\_ptr y, const size\_t incy)
- template<> void [fscal](#) (const Givaro::FloatDomain &, const size\_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::Element\_ptr y, const size\_t incy)
- template<class [Field](#) >  
void [fscal](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $fscal\ A \leftarrow a \cdot A.$
- template<class [Field](#) >  
void [fscal](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)  
 $fscal\ B \leftarrow a \cdot A.$

- `template<> void fscaln (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t inc)`
- `template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)`
- `template<> void fscaln (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t lda)`
- `template<> void fscal (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)`
- `template<> void fscaln (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha, typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, const size_t inc)`
- `template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t Ainc, FFPACK::rns_double::Element_ptr B, const size_t Binc)`
- `template<> void fscaln (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::Element_ptr A, const size_t lda)`
- `template<> void fscal (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const FFPACK::rns_double::Element alpha, FFPACK::rns_double::ConstElement_ptr A, const size_t lda, FFPACK::rns_double::Element_ptr B, const size_t ldb)`
- `template<class Field >`  
`Field::Element_ptr fsyr2k (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*fsyr2k: Symmetric Rank 2K update*
- `template<class Field >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*fsyrk: Symmetric Rank K update*
- `template<class Field >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)`  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold)`
- `template<class Field >`  
`Field::Element_ptr fsyrk (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::ConstElement_ptr D, const size_t incD, const std::vector< bool > &twoBlock, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const size_t threshold=__FFLASFFPACK_FSYRK_THRESHOLD)`  
*fsyrk: Symmetric Rank K update with diagonal scaling*

- `template<class Field >`  
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  
*ftrmm: TRIangular Matrix Multiply.*
- `template<class Field >`  
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`  
*ftrmm: TRIangular Matrix Multiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \beta C$  or  $C \leftarrow \alpha \text{Bop}(A) + \beta C$ .*
- `template<class Field >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`
- `template<class Field >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Sequential &PSH)`
- `template<class Field, class Cut, class Param >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Parallel< Cut, Param > &PSH)`
- `template<class Field, class ParSeqTrait = ParSeqHelper::Sequential>`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Recursive, ParSeqTrait > &H)`
- `void ftrsm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, Givaro::Integer *B, const size_t ldb)`
- `void cblas_imptrsm (const enum FFLAS_ORDER Order, const enum FFLAS_SIDE Side, const enum FFLAS_UPLO Uplo, const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_DIAG Diag, const int M, const int N, const FFPACK::rns_double_elt alpha, FFPACK::rns_double_elt_cstptr A, const int lda, FFPACK::rns_double_elt_ptr B, const int ldb)`
- `template<class Field >`  
`void ftrsv (const Field &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, int incX)`  
*ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*
- `void igemm_ (const enum FFLAS_ORDER Order, const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, const size_t M, const size_t N, const size_t K, const int64_t alpha, const int64_t *A, const size_t lda, const int64_t *B, const size_t ldb, const int64_t beta, int64_t *C, const size_t ldc)`
- `template<class Field, class OtherElement_ptr >`  
`void finit (const Field &F, const size_t n, const OtherElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  
*finit  $x \leftarrow y \bmod F$ .*
- `template<class Field, class OtherElement_ptr >`  
`void fconvert (const Field &F, const size_t n, OtherElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)`  
*fconvert  $x \leftarrow y \bmod F$ .*
- `template<class Field >`  
`void fnegin (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  
*fnegin  $x \leftarrow -x$ .*

- `template<class Field >`  
`void fneg (const Field &F, const size_t n, typename Field::ConstElement_ptr Y, const size_t incY, typename Field::Element_ptr X, const size_t incX)`  

$$fneg\ x \leftarrow -y.$$
- `template<class Field >`  
`void fzero (const Field &F, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$fzero : A \leftarrow 0.$$
- `template<class Field , class RandIter >`  
`void frand (const Field &F, RandIter &G, const size_t n, typename Field::Element_ptr X, const size_t incX)`  

$$frand : A \leftarrow random.$$
- `template<class Field >`  
`bool fiszero (const Field &F, const size_t n, typename Field::ConstElement_ptr X, const size_t incX)`  

$$fiszero : test\ X = 0.$$
- `template<class Field >`  
`bool fequal (const Field &F, const size_t n, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY)`  

$$fequal : test\ X = Y.$$
- `template<class Field >`  
`void faxpby (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)`  

$$faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$$
- `template<typename Field , class Cut , class Param >`  
`Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY, const ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class Field >`  
`void fswap (const Field &F, const size_t N, typename Field::Element_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`  

$$fswap : X \leftrightarrow Y.$$
- `template<class Field >`  
`void fzero (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  

$$fzero : A \leftarrow 0.$$
- `template<class Field , class RandIter >`  
`void frand (const Field &F, RandIter &G, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  

$$frand : A \leftarrow random.$$
- `template<class Field >`  
`bool fequal (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb)`  

$$fequal : test\ A = B.$$
- `template<class Field >`  
`bool fiszero (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda)`  

$$fiszero : test\ A = 0.$$
- `template<class Field >`  
`void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda, const typename Field::Element &d)`  

$$creates\ a\ diagonal\ matrix$$
- `template<class Field >`  
`void fidentity (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  

$$creates\ a\ diagonal\ matrix$$
- `template<class Field , class OtherElement_ptr >`  
`void finit (const Field &F, const size_t m, const size_t n, typename Field::Element_ptr A, const size_t lda)`  

$$finit\ Initializes\ A\ in\ F\$.$$

- template<class [Field](#) , class OtherElement\_ptr >  
void [fconvert](#) (const [Field](#) &F, const size\_t m, const size\_t n, OtherElement\_ptr A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)  
 $fconvert\ A \leftarrow BmodF.$
- template<class [Field](#) >  
void [fnegin](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $fnegin\ A \leftarrow -A.$
- template<class [Field](#) >  
void [fneg](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $fneg\ A \leftarrow -B.$
- template<class [Field](#) >  
void [faxpby](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t ldx, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t ldy)  
 $faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$
- template<class [Field](#) >  
void [fmove](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)  
 $fmove : A \leftarrow B\ and\ B \leftarrow 0.$
- template<class [Field](#) >  
size\_t [bitsize](#) (const [Field](#) &F, size\_t M, size\_t N, const typename [Field::ConstElement\\_ptr](#) A, size\_t lda)  
*bitsize: Computes the largest bitsize of the matrix' coefficients.*
- template<> size\_t [bitsize](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) > > (const [Givaro::ZRing](#)< [Givaro::Integer](#) > &F, size\_t M, size\_t N, const [Givaro::Integer](#) \*A, size\_t lda)
- template<class [Field](#) >  
void [ftmv](#) (const [Field](#) &F, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, int incX)  
*ftsm: TRIangular Matrix Vector prodcut Computes  $X \leftarrow op(A)X$*
- template<class [Field](#) >  
void [ftrsm](#) (const [Field](#) &F, const [FFLAS\\_SIDE](#) Side, const [FFLAS\\_UPLO](#) Uplo, const [FFLAS\\_TRANSPOSE](#) TransA, const [FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)  
*ftrsm: TRIangular System solve with Matrix.*
- template<typename [Field](#) >  
[Field::Element\\_ptr](#) [pfgemm](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, size\_t numthreads=0)
- template<class [Field](#) >  
[Field::Element](#) \* [pfgemm\\_1D\\_rec](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, const typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element](#) \*C, const size\_t ldc, size\_t seuil)
- template<class [Field](#) >  
[Field::Element](#) \* [pfgemm\\_2D\\_rec](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, const typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element](#) \*C, const size\_t ldc, size\_t seuil)
- template<class [Field](#) >  
[Field::Element](#) \* [pfgemm\\_3D\\_rec](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, const typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, size\_t seuil, size\_t \*x)

- `template<class Field >`  
`Field::Element_ptr pfgemm_3D_rec2` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `typename Field::Element` alpha, const `typename Field::Element_ptr` A, const `size_t` lda, const `typename Field::Element_ptr` B, const `size_t` ldb, const `typename Field::Element` beta, `typename Field::Element_ptr` C, const `size_t` ldc, `size_t` seuil, `size_t` \*x)
- `template<class Field , class ModeTrait , class Strat , class Param >`  
`std::enable_if<!std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >::value, typename Field::Element_ptr >::type fgemm` (const `Field` &F, const `FFLAS::FFLAS_TRANSPOSE` ta, const `FFLAS::FFLAS_TRANSPOSE` tb, const `size_t` m, const `size_t` n, const `size_t` k, const `typename Field::Element` alpha, `typename Field::ConstElement_ptr` A, const `size_t` lda, `typename Field::ConstElement_ptr` B, const `size_t` ldb, const `typename Field::Element` beta, `typename Field::Element_ptr` C, const `size_t` ldc, `MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat, Param > >` &H)
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr ftrsm` (const `Field` &F, const `FFLAS::FFLAS_SIDE` Side, const `FFLAS::FFLAS_UPLO` UpLo, const `FFLAS::FFLAS_TRANSPOSE` TA, const `FFLAS::FFLAS_DIAG` Diag, const `size_t` m, const `size_t` n, const `typename Field::Element` alpha, `typename Field::Element_ptr` A, const `size_t` lda, `typename Field::Element_ptr` B, const `size_t` ldb, `TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel< Cut, Param > >` &H)
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr ftrsm` (const `Field` &F, const `FFLAS::FFLAS_SIDE` Side, const `FFLAS::FFLAS_UPLO` UpLo, const `FFLAS::FFLAS_TRANSPOSE` TA, const `FFLAS::FFLAS_DIAG` Diag, const `size_t` m, const `size_t` n, const `typename Field::Element` alpha, `typename Field::Element_ptr` A, const `size_t` lda, `typename Field::Element_ptr` B, const `size_t` ldb, `TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > >` &H)
- `template<class Field , class SM >`  
`void fspmv` (const `Field` &F, const `SM` &A, `typename Field::ConstElement_ptr` x, const `typename Field::Element` &beta, `typename Field::Element_ptr` y)
- `template<class Field , class SM >`  
`void fspmm` (const `Field` &F, const `SM` &A, `size_t` blockSize, `typename Field::ConstElement_ptr` x, `int` ldx, const `typename Field::Element` &beta, `typename Field::Element_ptr` y, `int` ldy)
- `template<class Field , class IndexT >`  
`void sparse_init` (const `Field` &F, `Sparse< Field, SparseMatrix_t::COO >` &A, const `IndexT` \*row, const `IndexT` \*col, `typename Field::ConstElement_ptr` dat, `uint64_t` rowdim, `uint64_t` coldim, `uint64_t` nnz)
- `template<class Field , class IndexT >`  
`void sparse_init` (const `Field` &F, `Sparse< Field, SparseMatrix_t::COO_ZO >` &A, const `IndexT` \*row, const `IndexT` \*col, `typename Field::ConstElement_ptr` dat, `uint64_t` rowdim, `uint64_t` coldim, `uint64_t` nnz)
- `template<class Field >`  
`void sparse_delete` (const `Sparse< Field, SparseMatrix_t::COO >` &A)
- `template<class Field >`  
`void sparse_delete` (const `Sparse< Field, SparseMatrix_t::COO_ZO >` &A)
- `template<class Field , class IndexT >`  
`void sparse_init` (const `Field` &F, `Sparse< Field, SparseMatrix_t::CSR >` &A, const `IndexT` \*row, const `IndexT` \*col, `typename Field::ConstElement_ptr` dat, `uint64_t` rowdim, `uint64_t` coldim, `uint64_t` nnz)
- `template<class Field , class IndexT >`  
`void sparse_init` (const `Field` &F, `Sparse< Field, SparseMatrix_t::CSR_ZO >` &A, const `IndexT` \*row, const `IndexT` \*col, `typename Field::ConstElement_ptr` dat, `uint64_t` rowdim, `uint64_t` coldim, `uint64_t` nnz)
- `template<class Field >`  
`void sparse_delete` (const `Sparse< Field, SparseMatrix_t::CSR >` &A)
- `template<class Field >`  
`void sparse_delete` (const `Sparse< Field, SparseMatrix_t::CSR_ZO >` &A)
- `template<class Field >`  
`std::ostream & sparse_print` (std::ostream &os, const `Sparse< Field, SparseMatrix_t::CSR >` &A)
- `template<class IndexT >`  
`void sparse_init` (const `Givaro::Modular< Givaro::Integer >` &F, `Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR >` &A, const `IndexT` \*row, const `IndexT` \*col, `Givaro::Integer` \*dat, `uint64_t` rowdim, `uint64_t` coldim, `uint64_t` nnz)



- `template<class IndexT >`  
`void sparse_init (const Givaro::ZRing< Givaro::Integer > &F, Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT, size_t RECINT_SIZE>`  
`void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT, size_t RECINT_SIZE>`  
`void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_HYB > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_HYB > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_ZO > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > &A)`
- `template<class Field >`  
`void sparse_print (const Sparse< Field, SparseMatrix_t::ELL_simd > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::HYB_ZO > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::HYB_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<typename _Field >`  
`std::ostream & operator<< (std::ostream &os, const Sparse< _Field, SparseMatrix_t::HYB_ZO > &A)`
- `template<class Field, bool sorted = true, bool read_integer = false>`  
`void readSmsFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)`
- `template<class Field >`  
`void readSprFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnz)`
- `template<class T >`  
`std::enable_if< std::is_integral< T >::value, int > getData Type ()`

- `template<class T >`  
`std::enable_if< std::is_floating_point< T >::value, int > getDataType ()`
- `template<class T >`  
`std::enable_if< std::is_same< T, mpz_t >::value, int > getDataType ()`
- `template<class T >`  
`int getDataType ()`
- `template<class Field >`  
`void readMachineType (const Field &F, typename Field::Element &modulo, typename Field::Element_ptr val, std::ifstream &file, const uint64_t dims, const mask_t data_type, const mask_t field_desc)`
- `template<class Field >`  
`void readDnsFormat (const std::string &path, const Field &F, index_t &rowdim, index_t &coldim, typename Field::Element_ptr &val)`
- `template<class Field >`  
`void writeDnsFormat (const std::string &path, const Field &F, const index_t &rowdim, const index_t &coldim, typename Field::Element_ptr A, index_t lIdA)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field >`  
`void sparse\_delete (const Sparse< Field, SparseMatrix_t::SELL_ZO > &A)`
- `template<class Field >`  
`void sparse\_print (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field, class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz, uint64_t sigma=0)`
- `template<class Field, class IndexT >`  
`void sparse\_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class It >`  
`double computeDeviation (It begin, It end)`
- `template<class Field >`  
`StatsMatrix getStat (const Field &F, const index_t *row, const index_t *col, typename Field::ConstElement_ptr val, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<> void fflas\_delete (FFPACK::rns_double_elt_ptr A)`
- `template<> void fflas\_delete (FFPACK::rns_double_elt_cstptr A)`
- `template<> FFPACK::rns_double_elt_ptr fflas\_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const Alignment align)`
- `template<> FFPACK::rns_double_elt_ptr fflas\_new (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m, const size_t n, const Alignment align)`
- `template<typename RNS >`  
`void finit\_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)`
- `template<typename RNS >`  
`void finit\_trans\_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename RNS::Element_ptr A)`
- `template<typename RNS >`  
`void fconvert\_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)`
- `template<typename RNS >`  
`void fconvert\_trans\_rns (const FFPACK::RNSIntegerMod< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename RNS::ConstElement_ptr A)`
- `template<> FFPACK::rns_double_elt_ptr fflas\_new (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const Alignment align)`



- `template<> FFPACK::rns_double_elt_ptr fflas_new (const FFPACK::RNSInteger< FFPACK::rns_double > &F, const size_t m, const size_t n, const Alignment align)`
- `template<typename RNS > void finit_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, size_t k, const Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::Element_ptr A)`
- `template<typename RNS > void fconvert_rns (const FFPACK::RNSInteger< RNS > &F, const size_t m, const size_t n, Givaro::Integer alpha, Givaro::Integer *B, const size_t ldb, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr A)`
- `template INST_OR_DECL void freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)`  

$$\text{freduce } x \leftarrow x \bmod F.$$
- `template INST_OR_DECL void freduce (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)`  

$$\text{freduce } x \leftarrow y \bmod F.$$
- `template INST_OR_DECL void finit (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)`  

$$\text{finit } x \leftarrow y \bmod F.$$
- `template INST_OR_DECL void fconvert (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX, const FFLAS_ELT *Y, const size_t incY)`  

$$\text{fconvert } x \leftarrow y \bmod F.$$
- `template INST_OR_DECL void fnegin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)`  

$$\text{fnegin } x \leftarrow -x.$$
- `template INST_OR_DECL void fneg (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)`  

$$\text{fneg } x \leftarrow -y.$$
- `template INST_OR_DECL void fzero (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, FFLAS_ELT *X, const size_t incX)`  

$$\text{fzero} : A \leftarrow 0.$$
- `template INST_OR_DECL bool fiszero (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *X, const size_t incX)`  

$$\text{fiszero} : \text{test } X = 0.$$
- `template INST_OR_DECL bool fequal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT *X, const size_t incX, const FFLAS_ELT *Y, const size_t incY)`  

$$\text{fequal} : \text{test } X = Y.$$
- `template INST_OR_DECL void fassign (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *Y, const size_t incY, FFLAS_ELT *X, const size_t incX)`  

$$\text{fassign} : x \leftarrow y.$$
- `template INST_OR_DECL void fscaln (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT *X, const size_t incX)`  

$$\text{fscaln } x \leftarrow \alpha \cdot x.$$
- `template INST_OR_DECL void fscal (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)`  

$$\text{fscal } y \leftarrow \alpha \cdot x.$$
- `template INST_OR_DECL void faxpy (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT alpha, const FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)`  

$$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$$
- `template INST_OR_DECL FFLAS_ELT fdot (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *X, const size_t incX, const FFLAS_ELT *Y, const size_t incY)`  

$$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$$
- `template INST_OR_DECL void fswap (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, FFLAS_ELT *X, const size_t incX, FFLAS_ELT *Y, const size_t incY)`  

$$\text{fswap} : X \leftrightarrow Y.$$

- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
- template `INST_OR_DECL` void `fassign` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $fassign : A \leftarrow B.$
- template `INST_OR_DECL` void `fzero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
 $fzero : A \leftarrow 0.$
- template `INST_OR_DECL` bool `fequal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb)  
 $fequal : test A = B.$
- template `INST_OR_DECL` bool `fiszero` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*A, const size\_t lda)  
 $fiszero : test A = 0.$
- template `INST_OR_DECL` void `fidentity` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` &d)  
 $creates a diagonal matrix$
- template `INST_OR_DECL` void `fidentity` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
 $creates a diagonal matrix$
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
 $freduce A \leftarrow A mod F.$
- template `INST_OR_DECL` void `freduce` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $freduce A \leftarrow B mod F.$
- template `INST_OR_DECL` void `finit` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $finit A \leftarrow B mod F.$
- template `INST_OR_DECL` void `fnegin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda)  
 $fnegin A \leftarrow -A.$
- template `INST_OR_DECL` void `fneg` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*A, const size\_t lda)  
 $fneg A \leftarrow -B.$
- template `INST_OR_DECL` void `fscaln` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` \*A, const size\_t lda)  
 $fscaln A \leftarrow a \cdot A.$
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)  
 $fscal B \leftarrow a \cdot A.$
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t m, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t idx, `FFLAS_ELT` \*Y, const size\_t ldy)  
 $faxpy : y \leftarrow \alpha \cdot x + y.$

- template `INST_OR_DECL` void `fmove` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t m, const size\_t n, `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)
 
$$faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
 
$$fadd : \text{matrix addition.}$$
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
 
$$fsub : \text{matrix subtraction.}$$
- template `INST_OR_DECL` void `fsubin` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
 
$$fsubin : C = C - B$$
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
 
$$fadd : \text{matrix addition with scaling.}$$
- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)
 
$$faddin$$
- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemv` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` TransA, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` beta, `FFLAS_ELT` \*Y, const size\_t incY)
 
$$\text{finite prime } FFLAS\_FIELD<FFLAS\_ELT> \text{ G}eneral \text{ M}atrix \text{ V}ector \text{ m}ultiplication.$$
- template `INST_OR_DECL` void `fger` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*x, const size\_t incx, const `FFLAS_ELT` \*y, const size\_t incy, `FFLAS_ELT` \*A, const size\_t lda)
 
$$fger : \text{rank one update of a general matrix}$$
- template `INST_OR_DECL` void `ftsv` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*X, int incX)
 
$$ftsv : \text{TRIangular System solve with Vector Computes } X \leftarrow \text{op}(A^{-1})X$$
- template `INST_OR_DECL` void `ftsm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_SIDE` Side, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)
 
$$ftsm : \text{TRIangular System solve with Matrix.}$$
- template `INST_OR_DECL` void `ftmm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_SIDE` Side, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*B, const size\_t ldb)
 
$$ftmm : \text{TRIangular Matrix Multiply.}$$
- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const size\_t m, const size\_t n, const size\_t k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const size\_t ldc)
 
$$fgemm : \text{Field G}eneral \text{ M}atrix \text{ M}ultiply.$$
- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS_TRANSPOSE` ta, const `FFLAS_TRANSPOSE` tb, const size\_t m, const size\_t n, const size\_t k, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, const `FFLAS_ELT` beta, `FFLAS_ELT` \*C, const size\_t ldc, const `ParSeqHelper::Sequential` seq)

- template INST\_OR\_DECL FFLAS\_ELT \* fgemm (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const FFLAS\_ELT alpha, const FFLAS\_ELT \*A, const size\_t lda, const FFLAS\_ELT \*B, const size\_t ldb, const FFLAS\_ELT beta, FFLAS\_ELT \*C, const size\_t ldc, const ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive > par)
  - template INST\_OR\_DECL FFLAS\_ELT \* fgemm (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const FFLAS\_ELT alpha, const FFLAS\_ELT \*A, const size\_t lda, const FFLAS\_ELT \*B, const size\_t ldb, const FFLAS\_ELT beta, FFLAS\_ELT \*C, const size\_t ldc, const ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > par)
  - template INST\_OR\_DECL FFLAS\_ELT \* fsquare (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const FFLAS\_ELT alpha, const FFLAS\_ELT \*A, const size\_t lda, const FFLAS\_ELT beta, FFLAS\_ELT \*C, const size\_t ldc)
- fsquare: Squares a matrix.*
- template<class Cut = CuttingStrategy::Block, class Strat = StrategyParameter::Threads>  
void BlockCuts (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t numthreads)
  - template<> void BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t numthreads)
  - template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t numthreads)
  - template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t grainsize)
  - template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t grainsize)
  - template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t numthreads)
  - template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t grainsize)
  - template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t numthreads)
  - template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t numthreads)
  - template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t numthreads)
  - template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t numthreads)
  - template<class Cut = CuttingStrategy::Block, class Param = StrategyParameter::Threads>  
void BlockCuts (size\_t &rowBlockSize, size\_t &colBlockSize, size\_t &lastRBS, size\_t &lastCBS, size\_t &changeRBS, size\_t &changeCBS, size\_t &numRowBlock, size\_t &numColBlock, size\_t m, size\_t n, const size\_t numthreads)
  - template<class Field >  
void pfzero (const Field &F, size\_t m, size\_t n, typename Field::Element\_ptr C, size\_t BS=0)
  - template<class Field, class RandIter >  
void pfrand (const Field &F, RandIter &G, size\_t m, size\_t n, typename Field::Element\_ptr C, size\_t BS=0)
  - template<class Field, class Cut, class Param >  
Field::Element & fdot (const Field &F, const size\_t N, typename Field::ConstElement\_ptr x, const size\_t incx, typename Field::ConstElement\_ptr y, const size\_t incy, typename Field::Element &d, const ParSeqHelper::Parallel< Cut, Param > par)
  - template<class Field, class AlgoT, class FieldTrait >  
Field::Element \* pfgemm (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, const typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element \*C, const size\_t ldc, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > > &H)

- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDAdaptive`  
`> > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive >`  
`> &H)`
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr AA, const size_t lda, const typename Field::ConstElement_ptr BB, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoD > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element_ptr pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeD > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement_ptr A, const size_t lda, const typename Field::ConstElement_ptr B, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDInPlace >`  
`> &H)`
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t`  
`n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t`  
`lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta,`  
`typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel<`  
`CuttingStrategy::Recursive, StrategyParameter::Threads > > &H)`
- `template<class Field, class AlgoT, class FieldTrait, class Cut >`  
`Field::Element_ptr fgemv (const Field &F, const FFLAS_TRANSPOSE ta, const size_t m, const size_t`  
`n, const typename Field::Element alpha, const typename Field::ConstElement_ptr A, const size_t`  
`lda, const typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta,`  
`typename Field::Element_ptr Y, const size_t incY, MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel<`  
`CuttingStrategy::Row, Cut > > &H)`
- `void parseArguments (int argc, char **argv, Argument *args, bool printDefaults=true)`
- `std::ostream & writeCommandString (std::ostream &os, Argument *args, const char *programName=nullptr)`  
*writes the values of all arguments, preceded by the programName*
- `template<class Field >`  
`std::ostream & WriteMatrix (std::ostream &c, const Field &F, size_t m, size_t n, typename Field::ConstElement_ptr`  
`A, size_t lda, FFLAS_FORMAT format, bool column_major)`  
*WriteMatrix: write a matrix to an output stream.*
- `void preamble (std::ifstream &ifis, FFLAS_FORMAT &format)`
- `template<class Field >`  
`Field::Element_ptr ReadMatrix (std::ifstream &ifis, Field &F, size_t &m, size_t &n, typename Field::Element_ptr`  
`&A, FFLAS_FORMAT format=FflasAuto)`  
*ReadMatrix: read a matrix from an input stream.*

- `template<class Field >`  
`Field::Element_ptr ReadMatrix` (const std::string &matrix\_file, Field &F, size\_t &m, size\_t &n, typename Field::Element\_ptr &A, FFLAS\_FORMAT format=FflasAuto)  
*ReadMatrix: read a matrix from a file.*
- `template<class Field >`  
`void WriteMatrix` (std::string &matrix\_file, const Field &F, int m, int n, typename Field::ConstElement\_ptr A, size\_t lda, FFLAS\_FORMAT format=FflasDense, bool column\_major=false)  
*WriteMatrix: write a matrix to a file.*
- `std::ostream & WritePermutation` (std::ostream &c, const size\_t \*P, size\_t N)  
*WritePermutation: write a permutation matrix to an output stream.*
- `template<class Element >`  
`bool alignable` ()
- `template<> bool alignable< Givaro::Integer * > ()`
- `template<class Field >`  
`Field::Element_ptr fflas_new` (const Field &F, const size\_t m, const Alignment align=Alignment::DEFAULT)
- `template<class Field >`  
`Field::Element_ptr fflas_new` (const Field &F, const size\_t m, const size\_t n, const Alignment align=Alignment::DEFAULT)
- `template<class Element >`  
`Element * fflas_new` (const size\_t m, const Alignment align=Alignment::DEFAULT)
- `template<class Element_ptr >`  
`void fflas_delete` (Element\_ptr A)
- `template<class Ptr , class ... Args>`  
`void fflas_delete` (Ptr p, Args ... args)
- `void prefetch` (const int64\_t \*)
- `void getTLBSize` (int &tlb)
- `void queryCacheSizes` (int &l1, int &l2, int &l3)
- `int queryL1CacheSize` ()
- `int queryTopLevelCacheSize` ()
- `uint64_t getSeed` ()

## 15.1.1 Typedef Documentation

### 15.1.1.1 Checker\_fgemm

```
using Checker_fgemm = FFLAS::Checker_Empty<Field>
```

### 15.1.1.2 Checker\_ftrsm

```
using Checker_ftrsm = FFLAS::Checker_Empty<Field>
```

### 15.1.1.3 ForceCheck\_fgemm

```
using ForceCheck_fgemm = CheckerImplem_fgemm<Field>
```

#### 15.1.1.4 ForceCheck\_ftrsm

```
using ForceCheck_ftrsm = CheckerImplem_ftrsm<Field>
```

#### 15.1.1.5 ZOSparseMatrix

```
using ZOSparseMatrix = std::true_type
```

#### 15.1.1.6 NotZOSparseMatrix

```
using NotZOSparseMatrix = std::false_type
```

#### 15.1.1.7 SimdSparseMatrix

```
using SimdSparseMatrix = std::true_type
```

#### 15.1.1.8 NoSimdSparseMatrix

```
using NoSimdSparseMatrix = std::false_type
```

#### 15.1.1.9 MKLSparseMatrixFormat

```
using MKLSparseMatrixFormat = std::true_type
```

#### 15.1.1.10 NotMKLSparseMatrixFormat

```
using NotMKLSparseMatrixFormat = std::false_type
```

#### 15.1.1.11 has\_plus

```
using has_plus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_plus_impl<T> >::type
```

#### 15.1.1.12 has\_minus

```
using has_minus = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_minus_impl<T> >::type
```

#### 15.1.1.13 has\_equal

```
using has_equal = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
std::is_copy_assignable<T> >::type
```

#### 15.1.1.14 has\_plus\_eq

```
using has_plus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_plus_eq_impl<T> >::type
```

#### 15.1.1.15 has\_minus\_eq

```
using has_minus_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_minus_eq_impl<T> >::type
```

#### 15.1.1.16 has\_mul

```
using has_mul = typename std::conditional<std::is_arithmetic<T>::value, std::true_type, has_mul_impl<T>  
>::type
```

#### 15.1.1.17 has\_mul\_eq

```
using has_mul_eq = typename std::conditional<std::is_arithmetic<T>::value, std::true_type,  
has_mul_eq_impl<T> >::type
```

#### 15.1.1.18 Timer

```
typedef Givaro::Timer Timer
```



#### 15.1.1.19 BaseTimer

```
typedef Givaro::BaseTimer BaseTimer
```

#### 15.1.1.20 UserTimer

```
typedef Givaro::UserTimer UserTimer
```

#### 15.1.1.21 SysTimer

```
typedef Givaro::SysTimer SysTimer
```

### 15.1.2 Enumeration Type Documentation

#### 15.1.2.1 FFLAS\_ORDER

```
enum FFLAS_ORDER
```

Storage by row or col ?

Enumerator

|               |           |
|---------------|-----------|
| FflasRowMajor | row major |
| FflasColMajor | col major |

#### 15.1.2.2 FFLAS\_TRANSPOSE

```
enum FFLAS_TRANSPOSE
```

Is matrix transposed ?

Enumerator

|              |                           |
|--------------|---------------------------|
| FflasNoTrans | Matrix is not transposed. |
| FflasTrans   | Matrix is transposed.     |

### 15.1.2.3 FFLAS\_UPLO

enum [FFLAS\\_UPLO](#)

Is triangular matrix's shape upper ?

Enumerator

|            |  |
|------------|--|
| FflasUpper | Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$ ) |
| FflasLower | Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$ ) |

### 15.1.2.4 FFLAS\_DIAG

enum [FFLAS\\_DIAG](#)

Is the triangular matrix implicitly unit diagonal ?

Enumerator

|              |   |
|--------------|---|
| FflasNonUnit | Triangular matrix has an explicit arbitrary diagonal.             |
| FflasUnit    | Triangular matrix has an implicit unit diagonal ( $T_{i,i} = 1$ ) |

### 15.1.2.5 FFLAS\_SIDE

enum [FFLAS\\_SIDE](#)

On what side ?

Enumerator

|            |                                 |
|------------|---------------------------------|
| FflasLeft  | Operator applied on the left.   |
| FflasRight | Operator applied on the righth. |

### 15.1.2.6 FFLAS\_BASE

enum [FFLAS\\_BASE](#)

FFLAS\_BASE determines the type of the element representation for Matrix Mult kernel.

(deprecated, should not be used)

## Enumerator

|              |  |
|--------------|--|
| FflasDouble  | to use the double precision BLAS   |
| FflasFloat   | to use the single precison BLAS  |
| FflasGeneric | for any other domain, that can not be converted to floating point integers |

## 15.1.2.7 number\_kind

```
enum number_kind
```

## Enumerator

|       |  |
|-------|--|
| zero  |  |
| one   |  |
| mone  |  |
| other |  |

## 15.1.2.8 SparseMatrix\_t

```
enum class SparseMatrix_t [strong]
```

## Enumerator

|             |  |
|-------------|--|
| CSR         |  |
| CSR_ZO      |  |
| CSC         |  |
| CSC_ZO      |  |
| COO         |  |
| COO_ZO      |  |
| ELL         |  |
| ELL_ZO      |  |
| SELL        |  |
| SELL_ZO     |  |
| ELL_simd    |  |
| ELL_simd_ZO |  |
| CSR_HYB     |  |
| HYB_ZO      |  |

## 15.1.2.9 FFLAS\_FORMAT

```
enum FFLAS_FORMAT
```

## Enumerator

|               |  |
|---------------|--|
| FflasAuto     |  |
| FflasDense    |  |
| FflasSMS      |  |
| FflasBinary   |  |
| FflasMath     |  |
| FflasMaple    |  |
| FflasSageMath |  |

### 15.1.3 Function Documentation

#### 15.1.3.1 InfNorm()

```
Givaro::Integer InfNorm (
    const size_t M,
    const size_t N,
    const Givaro::Integer * A,
    const size_t lda ) [inline]
```

#### 15.1.3.2 min3()

```
const T & min3 (
    const T & m,
    const T & n,
    const T & k )
```

#### 15.1.3.3 max3()

```
const T & max3 (
    const T & m,
    const T & n,
    const T & k )
```

#### 15.1.3.4 min4()

```
const T & min4 (
    const T & m,
    const T & n,
    const T & k,
    const T & l )
```

### 15.1.3.5 max4()

```
const T & max4 (
    const T & m,
    const T & n,
    const T & k,
    const T & l )
```

### 15.1.3.6 fadd() [1/8]

```
void fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

### 15.1.3.7 faddin() [1/4]

```
void faddin (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

### 15.1.3.8 fsub() [1/4]

```
void fsub (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

### 15.1.3.9 fsubin() [1/3]

```
void fsubin (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

### 15.1.3.10 fadd() [2/8]

```
void fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc )
```

**Todo** optimise here

### 15.1.3.11 pfadd()

```
void pfadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t numths )
```

### 15.1.3.12 pfsub()

```
void pfsub (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t numths )
```

### 15.1.3.13 pfaddin()

```
void pfaddin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t numths )
```

### 15.1.3.14 pfsubin()

```
void pfsubin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t numths )
```

### 15.1.3.15 fadd() [3/8]

```
void fadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
```

```
const size_t ldb,  
typename Field::Element_ptr C,  
const size_t ldc )
```

fadd : matrix addition.

Computes  $C = A + B$ .



## Parameters

|            |                                   |
|------------|-----------------------------------|
| <i>F</i>   | field                             |
| <i>M</i>   | rows                              |
| <i>N</i>   | cols                              |
| <i>A</i>   | dense matrix of size $M \times N$ |
| <i>lda</i> | leading dimension of A            |
| <i>B</i>   | dense matrix of size $M \times N$ |
| <i>ldb</i> | leading dimension of B            |
| <i>C</i>   | dense matrix of size $M \times N$ |
| <i>ldc</i> | leading dimension of C            |

**15.1.3.16 fsub()** [2/4]

```
void fsub (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )
```

fsub : matrix subtraction.

Computes  $C = A - B$ .

## Parameters

|            |                                   |
|------------|-----------------------------------|
| <i>F</i>   | field                             |
| <i>M</i>   | rows                              |
| <i>N</i>   | cols                              |
| <i>A</i>   | dense matrix of size $M \times N$ |
| <i>lda</i> | leading dimension of A            |
| <i>B</i>   | dense matrix of size $M \times N$ |
| <i>ldb</i> | leading dimension of B            |
| <i>C</i>   | dense matrix of size $M \times N$ |
| <i>ldc</i> | leading dimension of C            |

**15.1.3.17 faddin()** [2/4]

```
void faddin (
    const Field & F,
```

```

    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

faddin

#### 15.1.3.18 fsubin() [2/3]

```

void fsubin (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

fsubin C = C - B

#### 15.1.3.19 fadd() [4/8]

```

void fadd (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr C,
    const size_t ldc )

```

fadd : matrix addition with scaling.

Computes  $C = A + \alpha B$ .

##### Parameters

|              |                          |
|--------------|--------------------------|
| <i>F</i>     | field                    |
| <i>M</i>     | rows                     |
| <i>N</i>     | cols                     |
| <i>A</i>     | dense matrix of size MxN |
| <i>lda</i>   | leading dimension of A   |
| <i>alpha</i> | some scalar              |
| <i>B</i>     | dense matrix of size MxN |
| <i>ldb</i>   | leading dimension of B   |
| <i>C</i>     | dense matrix of size MxN |
| <i>ldc</i>   | leading dimension of C   |

**15.1.3.20 fassign()** [1/10]

```

void fassign (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX ) [inline]

```

$\text{fassign} : x \leftarrow y.$

X is preallocated

**Todo** variant for triangular matrix

**Parameters**

|     |        |                     |
|-----|--------|---------------------|
|     | $F$    | field               |
|     | $N$    | size of the vectors |
| out | $X$    | vector in F         |
|     | $incX$ | stride of X         |
| in  | $Y$    | vector in F         |
|     | $incY$ | stride of Y         |

**15.1.3.21 fassign()** [2/10]

```

void fassign (
    const Givaro::Modular< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]

```

**15.1.3.22 fassign()** [3/10]

```

void fassign (
    const Givaro::ModularBalanced< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]

```

**15.1.3.23 fassign()** [4/10]

```
void fassign (
    const Givaro::ZRing< float > & F,
    const size_t N,
    const float * Y,
    const size_t incY,
    float * X,
    const size_t incX ) [inline]
```

**15.1.3.24 fassign()** [5/10]

```
void fassign (
    const Givaro::Modular< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

**15.1.3.25 fassign()** [6/10]

```
void fassign (
    const Givaro::ModularBalanced< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

**15.1.3.26 fassign()** [7/10]

```
void fassign (
    const Givaro::ZRing< double > & F,
    const size_t N,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX ) [inline]
```

**15.1.3.27 fassign()** [8/10]

```

void fassign (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )

```

$\text{fassign} : A \leftarrow B.$

**Parameters**

|       |                        |
|-------|------------------------|
| $F$   | field                  |
| $m$   | number of rows to copy |
| $n$   | number of cols to copy |
| $A$   | matrix in $F$          |
| $lda$ | stride of $A$          |
| $B$   | vector in $F$          |
| $ldb$ | stride of $B$          |

**15.1.3.28 faxpy()** [1/6]

```

void faxpy (
    const Field & F,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]

```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

**Parameters**

|         |         |                     |
|---------|---------|---------------------|
|         | $F$     | field               |
|         | $N$     | size of the vectors |
|         | $alpha$ | scalar              |
| in      | $X$     | vector in $F$       |
|         | $incX$  | stride of $X$       |
| in, out | $Y$     | vector in $F$       |
|         | $incY$  | stride of $Y$       |

**15.1.3.29 faxpy()** [2/6]

```
void faxpy (
    const Givaro::DoubleDomain & ,
    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]
```

**15.1.3.30 faxpy()** [3/6]

```
void faxpy (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]
```

**15.1.3.31 faxpy()** [4/6]

```
void faxpy (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t ldx,
    typename Field::Element_ptr Y,
    const size_t ldy ) [inline]
```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

**Parameters**

|         |          |                          |
|---------|----------|--------------------------|
|         | $F$      | field                    |
|         | $m$      | row dimension            |
|         | $n$      | column dimension         |
|         | $\alpha$ | scalar                   |
| in      | $X$      | vector in $F$            |
|         | $ldx$    | leading dimension of $X$ |
| in, out | $Y$      | vector in $F$            |
|         | $ldy$    | leading dimension of $Y$ |

**15.1.3.32 fdot()** [1/11]

```
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]
```

**15.1.3.33 fdot()** [2/11]

```
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DelayedTag & MT ) [inline]
```

**15.1.3.34 fdot()** [3/11]

```
Givaro::DoubleDomain::Element fdot (
    const Givaro::DoubleDomain & ,
    const size_t N,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]
```

**15.1.3.35 fdot()** [4/11]

```
Givaro::FloatDomain::Element fdot (
    const Givaro::FloatDomain & ,
    const size_t N,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultTag & MT ) [inline]
```

**15.1.3.36 fdot()** [5/11]

```
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::ConvertTo< T > & MT ) [inline]
```

**15.1.3.37 fdot()** [6/11]

```
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    ModeCategories::DefaultBoundedTag & dbt ) [inline]
```

**15.1.3.38 fdot()** [7/11]

```
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    const ParSeqHelper::Sequential seq ) [inline]
```

**15.1.3.39 fdot()** [8/11]

```
Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY ) [inline]
```

fdot: dot product  $x^T y$ .



## Parameters

|             |                     |
|-------------|---------------------|
| <i>F</i>    | field               |
| <i>N</i>    | size of the vectors |
| <i>X</i>    | vector in <i>F</i>  |
| <i>incX</i> | stride of <i>X</i>  |
| <i>Y</i>    | vector in <i>F</i>  |
| <i>incY</i> | stride of <i>Y</i>  |

## 15.1.3.40 fgemm() [1/23]

```
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFl
>, ParSeqHelper::Sequential > & H ) [inline]
```

## 15.1.3.41 fgemm() [2/23]

```
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Sequential seq ) [inline]
```

## 15.1.3.42 fgemm() [3/23]

```
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Parallel< Cut, Param > par ) [inline]
```

## 15.1.3.43 fgemm() [4/23]

```
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

fgemm: **Field** **GE**neral **M**atrix **M**ultiply.

Computes  $C = \alpha \text{op}(A) \times \text{op}(B) + \beta C$  Automatically set Winograd recursion level

## Parameters

|              |  |
|--------------|--|
| <i>F</i>     | field.   |
| <i>ta</i>    | if $ta == \text{FflasTrans}$ then $\text{op}(A) = A^t$ , else $\text{op}(A) = A$ , |
| <i>tb</i>    | same for matrix B  |
| <i>m</i>     | see A  |
| <i>n</i>     | see B  |
| <i>k</i>     | see A  |
| <i>alpha</i> | scalar   |
| <i>beta</i>  | scalar   |
| <i>A</i>     | $\text{op}(A)$ is $m \times k$   |
| <i>B</i>     | $\text{op}(B)$ is $k \times n$   |

## Parameters

|       |   |
|-------|---|
| $C$   | $C$ is $m \times n$   |
| $lda$ | leading dimension of A  |
| $ldb$ | leading dimension of B  |
| $ldc$ | leading dimension of C  |
| $w$   | recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of $w$ . |

## Warning

$\alpha$  must be invertible

## 15.1.3.44 fgemm() [5/23]

```
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Auto, ModeT, ParSeq > & H ) [inline]
```

## 15.1.3.45 fgemm() [6/23]

```
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential
> & H ) [inline]
```

**15.1.3.46 fsquare()** [1/6]

```
Field::Element_ptr fsquare (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

fsquare: Squares a matrix.

compute  $C \leftarrow \alpha \text{op}(A) \text{op}(A) + \beta C$  over a Field  $F$  Avoid the conversion of  $B$

**Parameters**

|              |   |
|--------------|---|
| <i>ta</i>    | if $ta == \text{FflasTrans}$ , $\text{op}(A) = A^T$ . |
| <i>F</i>     | field   |
| <i>n</i>     | size of $A$   |
| <i>alpha</i> | scalar  |
| <i>beta</i>  | scalar  |
| <i>A</i>     | dense matrix of size $n \times n$                     |
| <i>lda</i>   | leading dimension of $A$                              |
| <i>C</i>     | dense matrix of size $n \times n$                     |
| <i>ldc</i>   | leading dimension of $C$                              |

**Bug** why double ?

**15.1.3.47 fsquare()** [2/6]

```
double * fsquare (
    const Givaro::ModularBalanced< double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    const double beta,
    double * C,
    const size_t ldc ) [inline]
```

**15.1.3.48 fsquare()** [3/6]

```
float * fsquare (
    const Givaro::ModularBalanced< float > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const float alpha,
    const float * A,
    const size_t lda,
    const float beta,
    float * C,
    const size_t ldc ) [inline]
```

**15.1.3.49 fsquare()** [4/6]

```
double * fsquare (
    const Givaro::Modular< double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    const double beta,
    double * C,
    const size_t ldc ) [inline]
```

**15.1.3.50 fsquare()** [5/6]

```
float * fsquare (
    const Givaro::Modular< float > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const float alpha,
    const float * A,
    const size_t lda,
    const float beta,
    float * C,
    const size_t ldc ) [inline]
```

**15.1.3.51 fgemm()** [7/23]

```
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
```

```

    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > & H ) [inline]

```

### 15.1.3.52 fgemm() [8/23]

```

FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Sequential > & H ) [inline]

```

### 15.1.3.53 fgemm() [9/23]

```

FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads
    >, ParSeqTrait > > & H ) [inline]

```

**15.1.3.54 fgemm()** [10/23]

```
FFPACK::RNSInteger< RNS >::Element_ptr fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad,
    const size_t lda,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd,
    const size_t ldb,
    const typename FFPACK::RNSInteger< RNS >::Element beta,
    typename FFPACK::RNSInteger< RNS >::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag,
    ParSeqHelper::Parallel< Cut, Param > > & H ) [inline]
```

**15.1.3.55 fgemm()** [11/23]

```
Givaro::Integer * fgemm (
    const Givaro::ZRing< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,
    const size_t ldb,
    Givaro::Integer beta,
    Givaro::Integer * C,
    const size_t ldc,
    MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
    ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]
```

**15.1.3.56 fgemm()** [12/23]

```
RNS::Element_ptr fgemm (
    const FFPACK::RNSInteger< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename RNS::Element alpha,
```

```

    typename RNS::ConstElement_ptr Ad,
    const size_t lda,
    typename RNS::ConstElement_ptr Bd,
    const size_t ldb,
    const typename RNS::Element beta,
    typename RNS::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential
> & H ) [inline]

```

### 15.1.3.57 fgemm() [13/23]

```

RNS::Element_ptr fgemm (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename RNS::Element alpha,
    typename RNS::ConstElement_ptr Ad,
    const size_t lda,
    typename RNS::ConstElement_ptr Bd,
    const size_t ldb,
    const typename RNS::Element beta,
    typename RNS::Element_ptr Cd,
    const size_t ldc,
    MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > & H ) [inline]

```

### 15.1.3.58 fgemm() [14/23]

```

Givaro::Integer * fgemm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,
    const size_t ldb,
    const Givaro::Integer beta,
    Givaro::Integer * C,
    const size_t ldc,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]

```



**15.1.3.59 fgemm()** [15/23]

```
Givaro::Integer * fgemm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    const Givaro::Integer * B,
    const size_t ldb,
    const Givaro::Integer beta,
    Givaro::Integer * C,
    const size_t ldc,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Auto, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]
```

**15.1.3.60 fgemm()** [16/23]

```
RecInt::ruint< K1 > * fgemm (
    const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const RecInt::ruint< K1 > alpha,
    const RecInt::ruint< K1 > * A,
    const size_t lda,
    const RecInt::ruint< K1 > * B,
    const size_t ldb,
    RecInt::ruint< K1 > beta,
    RecInt::ruint< K1 > * C,
    const size_t ldc,
    MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic,
ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]
```

**15.1.3.61 fgemm()** [17/23]

```
Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
```

```

typename Field::ConstElement_ptr A,
const size_t lda,
typename Field::ConstElement_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
MMHelper< Field, MMHelperAlgo::Winograd, ModeT > & H ) [inline]

```

### 15.1.3.62 fgemm() [18/23]

```

Field::Element_ptr fgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut,
Param > > & H ) [inline]

```

### 15.1.3.63 fgemv() [1/19]

```

Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
> > & H ) [inline]

```

**15.1.3.64 fgemv()** [2/19]

```
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H ) [inline]
```

**15.1.3.65 fgemv()** [3/19]

```
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H ) [inline]
```

**15.1.3.66 fgemv()** [4/19]

```
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H ) [inline]
```

### 15.1.3.67 fgemv() [5/19]

```
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE TransA,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]
```

finite prime Field GEneral Matrix Vector multiplication.

Computes  $Y \leftarrow \alpha \text{op}(A)X + \beta Y$ .

#### Parameters

|     |          |   |
|-----|----------|---|
|     | $F$      | field   |
|     | $TransA$ | if $TransA == FflasTrans$ then $\text{op}(A) = A^t$ . |
|     | $M$      | rows  |
|     | $N$      | cols  |
|     | $alpha$  | scalar  |
|     | $A$      | dense matrix of size $M \times N$                     |
|     | $lda$    | leading dimension of $A$                              |
|     | $X$      | dense vector of size $N$                              |
|     | $incX$   | stride of $X$   |
|     | $beta$   | scalar  |
| out | $Y$      | dense vector of size $M$                              |
|     | $incY$   | stride of $Y$   |

### 15.1.3.68 fgemv() [6/19]

```
Givaro::ZRing< int64_t >::Element_ptr fgemv (
    const Givaro::ZRing< int64_t > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const int64_t alpha,
    const int64_t * A,
    const size_t lda,
    const int64_t * X,
    const size_t incX,
    const int64_t beta,
    int64_t * Y,
    const size_t incY,
```

```

    MMHelper< Givaro::ZRing< int64_t >, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

### 15.1.3.69 fgemv() [7/19]

```

Givaro::DoubleDomain::Element_ptr fgemv (
    const Givaro::DoubleDomain & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const Givaro::DoubleDomain::Element alpha,
    const Givaro::DoubleDomain::ConstElement_ptr A,
    const size_t lda,
    const Givaro::DoubleDomain::ConstElement_ptr X,
    const size_t incX,
    const Givaro::DoubleDomain::Element beta,
    Givaro::DoubleDomain::Element_ptr Y,
    const size_t incY,
    MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

### 15.1.3.70 fgemv() [8/19]

```

Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H )
[inline]

```

### 15.1.3.71 fgemv() [9/19]

```

Givaro::FloatDomain::Element_ptr fgemv (
    const Givaro::FloatDomain & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const Givaro::FloatDomain::Element alpha,

```

```

    const Givaro::FloatDomain::ConstElement_ptr A,
    const size_t lda,
    const Givaro::FloatDomain::ConstElement_ptr X,
    const size_t incX,
    const Givaro::FloatDomain::Element beta,
    Givaro::FloatDomain::Element_ptr Y,
    const size_t incY,
    MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

### 15.1.3.72 fgemv() [10/19]

```

Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    ParSeqHelper::Parallel< Cut, Param > & parH )

```

### 15.1.3.73 fgemv() [11/19]

```

Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    ParSeqHelper::Sequential & seqH )

```

**15.1.3.74 fgemv()** [12/19]

```
FFPACK::rns_double::Element_ptr fgemv (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::ConstElement_ptr X,
    const size_t incX,
    const FFPACK::rns_double::Element beta,
    FFPACK::rns_double::Element_ptr Y,
    const size_t incY,
    MMHelper< FFPACK::RNSInteger< FFPACK::rns_double >, MMHelperAlgo::Classic, ModeCategories::Default
> & H ) [inline]
```

**15.1.3.75 fgemv()** [13/19]

```
FFPACK::rns_double::Element_ptr fgemv (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::ConstElement_ptr X,
    const size_t incX,
    const FFPACK::rns_double::Element beta,
    FFPACK::rns_double::Element_ptr Y,
    const size_t incY,
    MMHelper< FFPACK::RNSIntegerMod< FFPACK::rns_double >, MMHelperAlgo::Classic,
ModeCategories::DefaultTag > & H ) [inline]
```

**15.1.3.76 fgemv()** [14/19]

```
Givaro::Integer * fgemv (
    const Givaro::ZRing< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const Givaro::Integer alpha,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * X,
    const size_t ldX,
    Givaro::Integer beta,
    Givaro::Integer * Y,
    const size_t ldY,
    MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]
```

**15.1.3.77 fgemv()** [15/19]

```
Givaro::Integer * fgemv (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const Givaro::Integer alpha,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * X,
    const size_t ldx,
    Givaro::Integer beta,
    Givaro::Integer * Y,
    const size_t ldy,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]
```

**15.1.3.78 fgemv()** [16/19]

```
RecInt::ruint< K1 > * fgemv (
    const Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const RecInt::ruint< K1 > alpha,
    const RecInt::ruint< K1 > * A,
    const size_t lda,
    const RecInt::ruint< K1 > * X,
    const size_t incx,
    RecInt::ruint< K1 > beta,
    RecInt::ruint< K1 > * Y,
    const size_t incy,
    MMHelper< Givaro::Modular< RecInt::ruint< K1 >, RecInt::ruint< K2 > >, MMHelperAlgo::Classic,
ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > & H ) [inline]
```

**15.1.3.79 fger()** [1/12]

```
void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]
```

fger: rank one update of a general matrix

Computes  $A \leftarrow \alpha x y^T + A$



## Parameters

|         |          |   |
|---------|----------|---|
|         | $F$      | field   |
|         | $M$      | rows  |
|         | $N$      | cols  |
|         | $\alpha$ | scalar  |
| in, out | $A$      | dense matrix of size $M \times N$ and leading dimension $lda$ |
|         | $lda$    | leading dimension of $A$                                      |
|         | $x$      | dense vector of size $M$                                      |
|         | $incx$   | stride of $x$   |
|         | $y$      | dense vector of size $N$                                      |
|         | $incy$   | stride of $y$   |

## 15.1.3.80 fger() [2/12]

```

void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
> > & H ) [inline]

```

## 15.1.3.81 fger() [3/12]

```

void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, AnyTag > & H ) [inline]

```

**15.1.3.82 fger()** [4/12]

```

void fger (
    const Givaro::DoubleDomain & F,
    const size_t M,
    const size_t N,
    const Givaro::DoubleDomain::Element alpha,
    const Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    const Givaro::DoubleDomain::ConstElement_ptr y,
    const size_t incy,
    Givaro::DoubleDomain::Element_ptr A,
    const size_t lda,
    MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

**15.1.3.83 fger()** [5/12]

```

void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr x,
    const size_t incx,
    const typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H )
[inline]

```

**15.1.3.84 fger()** [6/12]

```

void fger (
    const Givaro::FloatDomain & F,
    const size_t M,
    const size_t N,
    const Givaro::FloatDomain::Element alpha,
    const Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    const Givaro::FloatDomain::ConstElement_ptr y,
    const size_t incy,
    Givaro::FloatDomain::Element_ptr A,
    const size_t lda,
    MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

**15.1.3.85 fger()** [7/12]

```

void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H ) [inline]

```

**15.1.3.86 fger()** [8/12]

```

void fger (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda,
    MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H ) [inline]

```

**15.1.3.87 fger()** [9/12]

```

void fger (
    const Givaro::Modular< Givaro::Integer > & F,
    const size_t M,
    const size_t N,
    const typename Givaro::Integer alpha,
    typename Givaro::Integer * x,
    const size_t incx,
    typename Givaro::Integer * y,
    const size_t incy,
    typename Givaro::Integer * A,
    const size_t lda,
    MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo<
ElementCategories::RNSElementTag > > & H ) [inline]

```

**15.1.3.88 fger()** [10/12]

```

void fger (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t M,
    const size_t N,
    const typename FFPACK::RNSInteger< RNS >::Element alpha,
    typename FFPACK::RNSInteger< RNS >::Element_ptr x,
    const size_t incx,
    typename FFPACK::RNSInteger< RNS >::Element_ptr y,
    const size_t incy,
    typename FFPACK::RNSInteger< RNS >::Element_ptr A,
    const size_t lda,
    MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H ) [inline]

```

**15.1.3.89 fger()** [11/12]

```

void fger (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t M,
    const size_t N,
    const typename FFPACK::RNSIntegerMod< RNS >::Element alpha,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr x,
    const size_t incx,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr y,
    const size_t incy,
    typename FFPACK::RNSIntegerMod< RNS >::Element_ptr A,
    const size_t lda,
    MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Classic > & H ) [inline]

```

**15.1.3.90 freduce()** [1/10]

```

void freduce (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )

```

freduce  $x \leftarrow y \bmod F$ .

**Parameters**

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $n$    | size of the vectors |
| $Y$    | vector of Element   |
| $incY$ | stride of Y         |
| $X$    | vector in F         |
| $incX$ | stride of X         |

**Bug** use `cblas_(d)scal` when possible

### 15.1.3.91 `freduce()` [2/10]

```
void freduce (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

$\text{freduce } x \leftarrow x \bmod F.$

Parameters

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $n$    | size of the vectors |
| $X$    | vector in $F$       |
| $incX$ | stride of $X$       |

**Bug** use `cblas_(d)scal` when possible

### 15.1.3.92 `freduce_constoverride()` [1/2]

```
void freduce_constoverride (
    const Field & F,
    const size_t m,
    typename Field::ConstElement_ptr A,
    const size_t incX )
```

### 15.1.3.93 `finit()` [1/8]

```
void finit (
    const Field & F,
    const size_t n,
    ConstOtherElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )
```

**15.1.3.94 finit()** [2/8]

```
void finit (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

finit Initializes  $X$  in  $F$ .

**Parameters**

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $n$    | size of the vectors |
| $X$    | vector in $F$       |
| $incX$ | stride of $X$       |

**15.1.3.95 freduce()** [3/10]

```
void freduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

freduce  $A \leftarrow A \bmod F$ .

**Parameters**

|       |                |
|-------|----------------|
| $F$   | field          |
| $m$   | number of rows |
| $n$   | number of cols |
| $A$   | matrix in $F$  |
| $lda$ | stride of $A$  |

**15.1.3.96 pfreduce()**

```
void pfreduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t numths )
```

**15.1.3.97 freduce()** [4/10]

```

void freduce (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )

```

$\text{freduce } A \leftarrow B \bmod F.$

**Parameters**

|       |                            |
|-------|----------------------------|
| $F$   | field                      |
| $m$   | number of rows             |
| $n$   | number of cols             |
| $A$   | matrix in $F$              |
| $lda$ | stride of $A$              |
| $B$   | matrix in $\text{Element}$ |
| $ldb$ | stride of $B$              |

**15.1.3.98 freduce\_constoverride()** [2/2]

```

void freduce_constoverride (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,
    const size_t lda )

```

**15.1.3.99 finit()** [3/8]

```

void finit (
    const Field & F,
    const size_t m,
    const size_t n,
    const OtherElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )

```

$\text{finit } A \leftarrow B \bmod F.$

**Parameters**

|     |                |
|-----|----------------|
| $F$ | field          |
| $m$ | number of rows |

## Parameters

|            |                                     |
|------------|-------------------------------------|
| <i>n</i>   | number of cols                      |
| <i>A</i>   | matrix in $F$                       |
| <i>lda</i> | stride of $A$                       |
| <i>B</i>   | matrix in <code>OtherElement</code> |
| <i>ldb</i> | stride of $B$                       |

**15.1.3.100 finit()** [4/8]

```
void finit (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

**15.1.3.101 freduce()** [5/10]

```
void freduce (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A,
    size_t inc ) [inline]
```

**15.1.3.102 freduce()** [6/10]

```
void freduce (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    FFPACK::rns_double::Element_ptr A,
    size_t lda ) [inline]
```



**15.1.3.103 freivalds()**

```

bool freivalds (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::ConstElement_ptr C,
    const size_t ldc ) [inline]

```

freivalds: Freivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.

Randomly Checks  $C = \alpha \text{op}(A) \times \text{op}(B)$

**Parameters**

|          |  |
|----------|--|
| $F$      | field.   |
| $ta$     | if $ta == \text{FflasTrans}$ then $\text{op}(A) = A^t$ , else $\text{op}(A) = A$ , |
| $tb$     | same for matrix B  |
| $m$      | see A  |
| $n$      | see B  |
| $k$      | see A  |
| $\alpha$ | scalar   |
| $A$      | $\text{op}(A)$ is $m \times k$   |
| $B$      | $\text{op}(B)$ is $k \times n$   |
| $C$      | $C$ is $m \times n$  |
| $lda$    | leading dimension of A   |
| $ldb$    | leading dimension of B   |
| $ldc$    | leading dimension of C   |

**15.1.3.104 fscaln()** [1/10]

```

void fscaln (
    const Field & F,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr X,
    const size_t incX ) [inline]

```

fscaln  $x \leftarrow \alpha \cdot x$ .

**Parameters**

|     |       |
|-----|-------|
| $F$ | field |
|-----|-------|

## Parameters

|          |                        |
|----------|------------------------|
| $n$      | size of the vectors    |
| $\alpha$ | scalar                 |
| $X$      | vector in $\mathbb{F}$ |
| $incX$   | stride of $X$          |

**Bug** use `cblas_(d)scal` when possible

**Todo** check if comparison with  $\pm 1, 0$  is necessary.

15.1.3.105 `fscal()` [1/10]

```
void fscal (
    const Field & F,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]
```

`fscal`  $y \leftarrow \alpha \cdot x$ .

## Parameters

|     |          |                        |
|-----|----------|------------------------|
|     | $F$      | field                  |
|     | $n$      | size of the vectors    |
|     | $\alpha$ | scalar                 |
| in  | $X$      | vector in $\mathbb{F}$ |
|     | $incX$   | stride of $X$          |
| out | $Y$      | vector in $\mathbb{F}$ |
|     | $incY$   | stride of $Y$          |

**Bug** use `cblas_(d)scal` when possible

**Todo** check if comparison with  $\pm 1, 0$  is necessary.

15.1.3.106 `fscal()` [2/10]

```
void fscal (
    const Givaro::DoubleDomain & ,
```

```

    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]

```

#### 15.1.3.107 fscal() [3/10]

```

void fscal (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::ConstElement_ptr x,
    const size_t incx,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]

```

#### 15.1.3.108 fscaln() [2/10]

```

void fscaln (
    const Givaro::DoubleDomain & ,
    const size_t N,
    const Givaro::DoubleDomain::Element a,
    Givaro::DoubleDomain::Element_ptr y,
    const size_t incy ) [inline]

```

#### 15.1.3.109 fscaln() [3/10]

```

void fscaln (
    const Givaro::FloatDomain & ,
    const size_t N,
    const Givaro::FloatDomain::Element a,
    Givaro::FloatDomain::Element_ptr y,
    const size_t incy ) [inline]

```

#### 15.1.3.110 fscaln() [4/10]

```

void fscaln (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]

```

$\text{fscaln } A \leftarrow a \cdot A.$

## Parameters

|          |                  |
|----------|------------------|
| $F$      | field            |
| $m$      | number of rows   |
| $n$      | number of cols   |
| $\alpha$ | homotecie scalar |
| $A$      | matrix in $F$    |
| $lda$    | stride of $A$    |

15.1.3.111 **fscal()** [4/10]

```
void fscal (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb ) [inline]
```

$\text{fscal } B \leftarrow a \cdot A.$

## Parameters

|     |          |                  |
|-----|----------|------------------|
|     | $F$      | field            |
|     | $m$      | number of rows   |
|     | $n$      | number of cols   |
|     | $\alpha$ | homotecie scalar |
| in  | $A$      | matrix in $F$    |
|     | $lda$    | stride of $A$    |
| out | $B$      | matrix in $F$    |
|     | $ldb$    | stride of $B$    |

15.1.3.112 **fscaln()** [5/10]

```
void fscaln (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t inc ) [inline]
```

**15.1.3.113 fscal()** [5/10]

```

void fscal (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t Ainc,
    FFPACK::rns_double::Element_ptr B,
    const size_t Binc ) [inline]

```

**15.1.3.114 fscaln()** [6/10]

```

void fscaln (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t lda ) [inline]

```

**15.1.3.115 fscal()** [6/10]

```

void fscal (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::Element_ptr B,
    const size_t ldb ) [inline]

```

**15.1.3.116 fscaln()** [7/10]

```

void fscaln (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    const typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element alpha,
    typename FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A,
    const size_t inc ) [inline]

```

**15.1.3.117 fscal()** [7/10]

```
void fscal (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t Ainc,
    FFPACK::rns_double::Element_ptr B,
    const size_t Binc ) [inline]
```

**15.1.3.118 fscaln()** [8/10]

```
void fscaln (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::Element_ptr A,
    const size_t lda ) [inline]
```

**15.1.3.119 fscal()** [8/10]

```
void fscal (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const FFPACK::rns_double::Element alpha,
    FFPACK::rns_double::ConstElement_ptr A,
    const size_t lda,
    FFPACK::rns_double::Element_ptr B,
    const size_t ldb ) [inline]
```

**15.1.3.120 fsyr2k()**

```
Field::Element_ptr fsyr2k (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

fsyr2k: Symmetric Rank 2K update

Computes the Lower or Upper triangular part of  $C = \alpha(A \times B^T + B \times A^T) + \beta C$  or  $C = \alpha(A^T \times B + B^T \times A) + \beta C$

## Parameters

|              |  |
|--------------|--|
| <i>F</i>     | field.   |
| <i>UpLo</i>  | whether to compute the upper or the lower triangular part of the symmetric matrix C  |
| <i>trans</i> | if <code>ta==FflasNoTrans</code> then compute $C = \alpha(A \times B^T + B \times A^T) + \beta C$ , else $C = \alpha(A^T \times B + B^T \times A) + \beta C$ |
| <i>n</i>     | order of matrix C  |
| <i>k</i>     | see A  |
| <i>alpha</i> | scalar   |
| <i>A</i>     | <i>A</i> is $n \times k$ (FflasNoTrans) or <i>A</i> is $k \times n$ (FflasTrans)   |
| <i>lda</i>   | leading dimension of A   |
| <i>beta</i>  | scalar   |
| <i>C</i>     | <i>C</i> is $n \times n$   |
| <i>ldc</i>   | leading dimension of C   |

## Warning

$\alpha$  must be invertible

## 15.1.3.121 fsyrk() [1/5]

```
Field::Element_ptr fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]
```

fsyrk: Symmetric Rank K update

Computes the Lower or Upper triangular part of  $C = \alpha A \times A^T + \beta C$  or  $C = \alpha A^T \times A + \beta C$

## Parameters

|              |  |
|--------------|--|
| <i>F</i>     | field.   |
| <i>UpLo</i>  | whether to compute the upper or the lower triangular part of the symmetric matrix C  |
| <i>trans</i> | if <code>ta==FflasNoTrans</code> then compute $C = \alpha A \times A^T + \beta C$ , else $C = \alpha A^T \times A + \beta C$ |
| <i>n</i>     | order of matrix C  |
| <i>k</i>     | see A  |
| <i>alpha</i> | scalar   |
| <i>A</i>     | <i>A</i> is $n \times k$ or <i>A</i> is $k \times n$   |
| <i>lda</i>   | leading dimension of A   |
| <i>beta</i>  | scalar   |
| <i>C</i>     | <i>C</i> is $n \times n$   |
| <i>ldc</i>   | leading dimension of C   |

## Warning

$\alpha$  must be invertible

## 15.1.3.122 fsyrk() [2/5]

```
Field::Element_ptr fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
    const size_t incD,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t threshold = __FFLASFFPACK_FSYRK_THRESHOLD ) [inline]
```

fsyrk: Symmetric Rank K update with diagonal scaling

Computes the Lower or Upper triangular part of  $C = \alpha A \times D \times A^T + \beta C$  or  $C = \alpha A^T \times D \times A + \beta C$  where  $D$  is a diagonal matrix. Matrix  $A$  is updated into  $D \times A$  (if trans = FflasTrans) or  $A \times D$  (if trans = FflasNoTrans).

## Parameters

|         |  |
|---------|--|
| $F$     | field.   |
| $UpLo$  | whether to compute the upper or the lower triangular part of the symmetric matrix $C$                                  |
| $trans$ | if $trans == FflasNoTrans$ then compute $C = \alpha A \times A^T + \beta C$ , else $C = \alpha A^T \times A + \beta C$ |
| $n$     | order of matrix $C$  |
| $k$     | see $A$  |
| $alpha$ | scalar   |
| $A$     | $A$ is $n \times k$ or $A$ is $k \times n$   |
| $lda$   | leading dimension of $A$   |
| $D$     | $D$ is $k \times k$ diagonal matrix, stored as a vector of $k$ coefficients  |
| $lda$   | leading dimension of $A$   |
| $beta$  | scalar   |
| $C$     | $C$ is $n \times n$  |
| $ldc$   | leading dimension of $C$   |

## Warning

$\alpha$  must be invertible



**15.1.3.123 fsyrk()** [3/5]

```
Field::Element_ptr fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
    const size_t incD,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Sequential seq,
    const size_t threshold ) [inline]
```

**15.1.3.124 fsyrk()** [4/5]

```
Field::Element_ptr fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t N,
    const size_t K,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
    const size_t incD,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const ParSeqHelper::Parallel< Cut, Param > par,
    const size_t threshold ) [inline]
```

**15.1.3.125 fsyrk()** [5/5]

```
Field::Element_ptr fsyrk (
    const Field & F,
    const FFLAS_UPLO UpLo,
    const FFLAS_TRANSPOSE trans,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr D,
```

```

const size_t incD,
const std::vector< bool > & twoBlock,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const size_t threshold = __FFLASFFPACK_FSYRK_THRESHOLD ) [inline]

```

fsyrk: Symmetric Rank K update with diagonal scaling

Computes the Lower or Upper triangular part of  $C = \alpha A \times \text{Delta} D \times A^T + \beta C$  or  $C = \alpha A^T \times \text{Delta} D \times A + \beta C$  where D is a diagonal matrix and Delta is a block diagonal with either 1 on the diagonal or 2x2 swap blocks Matrix A is updated into  $D \times A$  (if trans = FflasTrans) or  $A \times D$  (if trans = FflasNoTrans).

#### Parameters

|                  |   |
|------------------|---|
| <i>F</i>         | field.  |
| <i>UpLo</i>      | whether to compute the upper or the lower triangular part of the symmetric matrix C   |
| <i>trans</i>     | if ta==FflasNoTrans then compute $C = \alpha A \text{Delta} D \times A^T + \beta C$ , else $C = \alpha A^T \text{Delta} D \times A + \beta C$ |
| <i>n</i>         | see B   |
| <i>k</i>         | see A   |
| <i>alpha</i>     | scalar  |
| <i>A</i>         | <i>A</i> is $n \times k$ or <i>A</i> is $k \times n$  |
| <i>lda</i>       | leading dimension of A  |
| <i>D</i>         | <i>D</i> is $k \times k$ diagonal matrix, stored as a vector of k coefficients  |
| <i>twoBlocks</i> | a vector boolean indicating the beginning of each 2x2 blocs in Delta  |
| <i>lda</i>       | leading dimension of A  |
| <i>beta</i>      | scalar  |
| <i>C</i>         | <i>C</i> is $n \times n$  |
| <i>ldc</i>       | leading dimension of C  |

#### Warning

$\alpha$  must be invertible

#### 15.1.3.126 ftrmm() [1/3]

```

void ftrmm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb ) [inline]

```

ftrmm: **TR**iangular **M**atrix **M**ultiply.

Computes  $B \leftarrow \alpha \text{op}(A)B$  or  $B \leftarrow \alpha B \text{op}(A)$ .

## Parameters

|               |   |
|---------------|---|
| <i>F</i>      | field   |
| <i>Side</i>   | if <code>Side==FflasLeft</code> then $B \leftarrow \alpha \text{op}(A)B$ is computed.                   |
| <i>Uplo</i>   | if <code>Uplo==FflasUpper</code> then A is upper triangular   |
| <i>TransA</i> | if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$ .  |
| <i>Diag</i>   | if <code>Diag==FflasUnit</code> then A is implicitly unit.  |
| <i>M</i>      | rows of B   |
| <i>N</i>      | cols of B   |
| <i>alpha</i>  | scalar  |
| <i>A</i>      | triangular matrix. If <code>Side==FflasLeft</code> then A is $N \times N$ , otherwise A is $M \times M$ |
| <i>lda</i>    | leading dim of A  |
| <i>B</i>      | matrix of size MxN  |
| <i>ldb</i>    | leading dim of B  |

## 15.1.3.127 ftrmm() [2/3]

```

void ftrmm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]

```

ftrmm: **TR**iangular **M**atrix **M**ultiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \beta C$  or  $C \leftarrow \alpha B \text{op}(A) + \beta C$ .

## Parameters

|               |   |
|---------------|---|
| <i>F</i>      | field   |
| <i>Side</i>   | if <code>Side==FflasLeft</code> then $B \leftarrow \alpha \text{op}(A)B$ is computed.                   |
| <i>Uplo</i>   | if <code>Uplo==FflasUpper</code> then A is upper triangular   |
| <i>TransA</i> | if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$ .  |
| <i>Diag</i>   | if <code>Diag==FflasUnit</code> then A is implicitly unit.  |
| <i>M</i>      | rows of B   |
| <i>N</i>      | cols of B   |
| <i>alpha</i>  | scalar  |
| <i>A</i>      | triangular matrix. If <code>Side==FflasLeft</code> then A is $N \times N$ , otherwise A is $M \times M$ |
| <i>lda</i>    | leading dim of A  |
| <i>B</i>      | matrix of size MxN  |

## Parameters

|             |                    |
|-------------|--------------------|
| <i>ldb</i>  | leading dim of B   |
| <i>beta</i> | scalar             |
| <i>C</i>    | matrix of size MxN |
| <i>ldc</i>  | leading dim of C   |

**15.1.3.128 ftrsm()** [1/9]

```

void ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb ) [inline]

```

**15.1.3.129 ftrsm()** [2/9]

```

void ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const ParSeqHelper::Sequential & PSH ) [inline]

```

**15.1.3.130 ftrsm()** [3/9]

```

void ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const ParSeqHelper::Parallel< Cut, Param > & PSH ) [inline]

```

**15.1.3.131 ftrsm()** [4/9]

```

void ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    TRSMHelper< StructureHelper::Recursive, ParSeqTrait > & H ) [inline]

```

**15.1.3.132 ftrsm()** [5/9]

```

void ftrsm (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const Givaro::Integer alpha,
    const Givaro::Integer * A,
    const size_t lda,
    Givaro::Integer * B,
    const size_t ldb ) [inline]

```

**15.1.3.133 cblas\_impstrsm()**

```

void cblas_impstrsm (
    const enum FFLAS_ORDER Order,
    const enum FFLAS_SIDE Side,
    const enum FFLAS_UPLO Uplo,
    const enum FFLAS_TRANSPOSE TransA,
    const enum FFLAS_DIAG Diag,
    const int M,
    const int N,
    const FFPACK::rns_double_elt alpha,
    FFPACK::rns_double_elt_cstptr A,
    const int lda,
    FFPACK::rns_double_elt_ptr B,
    const int ldb ) [inline]

```

**15.1.3.134 ftrsv() [1/2]**

```

void ftrsv (
    const Field & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    int incX ) [inline]

```

ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$

**Parameters**

|               |  |
|---------------|--|
| <i>F</i>      | field  |
| <i>X</i>      | vector of size N on a field F                      |
| <i>incX</i>   | stride of X  |
| <i>A</i>      | a matrix of leading dimension lda and size N       |
| <i>lda</i>    | leading dimension of A                             |
| <i>N</i>      | number of rows or columns of A according to TransA |
| <i>TransA</i> | if TransA==FflasTrans then $\text{op}(A) = A^t$ .  |
| <i>Diag</i>   | if Diag==FflasUnit then A is unit.                 |
| <i>Uplo</i>   | if Uplo==FflasUpper then A is upper triangular     |

**15.1.3.135 igemm\_()**

```

void igemm_ (
    const enum FFLAS_ORDER Order,

```

```

const enum FFLAS_TRANSPOSE TransA,
const enum FFLAS_TRANSPOSE TransB,
const size_t M,
const size_t N,
const size_t K,
const int64_t alpha,
const int64_t * A,
const size_t lda,
const int64_t * B,
const size_t ldb,
const int64_t beta,
int64_t * C,
const size_t ldc ) [inline]

```

### 15.1.3.136 finit() [5/8]

```

void finit (
    const Field & F,
    const size_t n,
    const OtherElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )

```

$\text{finit } x \leftarrow y \bmod F.$

#### Parameters

|        |                        |
|--------|------------------------|
| $F$    | field                  |
| $n$    | size of the vectors    |
| $Y$    | vector of OtherElement |
| $incY$ | stride of Y            |
| $X$    | vector in F            |
| $incX$ | stride of X            |

**Bug** use `cblas_(d)scal` when possible

### 15.1.3.137 fconvert() [1/3]

```

void fconvert (
    const Field & F,
    const size_t n,
    OtherElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY )

```

$\text{fconvert } x \leftarrow y \bmod F.$

## Parameters

|        |                                     |
|--------|-------------------------------------|
| $F$    | field                               |
| $n$    | size of the vectors                 |
| $Y$    | vector of $F$                       |
| $incY$ | stride of $Y$                       |
| $X$    | vector in <code>OtherElement</code> |
| $incX$ | stride of $X$                       |

**Bug** use `cblas_(d)scal` when possible

### 15.1.3.138 `fnegin()` [1/4]

```
void fnegin (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

`fnegin`  $x \leftarrow -x$ .

## Parameters

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $n$    | size of the vectors |
| $X$    | vector in $F$       |
| $incX$ | stride of $X$       |

**Bug** use `cblas_(d)scal` when possible

### 15.1.3.139 `fneg()` [1/4]

```
void fneg (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    typename Field::Element_ptr X,
    const size_t incX )
```

`fneg`  $x \leftarrow -y$ .



## Parameters

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $n$    | size of the vectors |
| $X$    | vector in $F$       |
| $incX$ | stride of $X$       |
| $Y$    | vector in $F$       |
| $incY$ | stride of $Y$       |

**Bug** use `cblas_(d)scal` when possible

**15.1.3.140 fzero()** [1/4]

```
void fzero (
    const Field & F,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

$fzero : A \leftarrow 0.$

## Parameters

|        |                            |
|--------|----------------------------|
| $F$    | field                      |
| $n$    | number of elements to zero |
| $X$    | vector in $F$              |
| $incX$ | stride of $X$              |

**15.1.3.141 frand()** [1/2]

```
void frand (
    const Field & F,
    RandIter & G,
    const size_t n,
    typename Field::Element_ptr X,
    const size_t incX )
```

$frand : A \leftarrow random.$

## Parameters

|        |                                 |
|--------|---------------------------------|
| $F$    | field                           |
| $G$    | randomiterator                  |
| $n$    | number of elements to randomize |
| $X$    | vector in $F$                   |
| $incX$ | stride of $X$                   |

**15.1.3.142 fiszero()** [1/4]

```
bool fiszero (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr X,
    const size_t incX )
```

**fiszero** : test  $X = 0$ .

**Parameters**

|        |                  |
|--------|------------------|
| $F$    | field            |
| $n$    | vector dimension |
| $X$    | vector in $F$    |
| $incX$ | increment of $X$ |

**15.1.3.143 fequal()** [1/4]

```
bool fequal (
    const Field & F,
    const size_t n,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY )
```

**fequal** : test  $X = Y$ .

**Parameters**

|        |                  |
|--------|------------------|
| $F$    | field            |
| $n$    | vector dimension |
| $X$    | vector in $F$    |
| $incX$ | increment of $X$ |
| $Y$    | vector in $F$    |
| $incY$ | increment of $Y$ |

**15.1.3.144 faxpby()** [1/2]

```
void faxpby (
    const Field & F,
    const size_t N,
```

```

const typename Field::Element alpha,
typename Field::ConstElement_ptr X,
const size_t incX,
const typename Field::Element beta,
typename Field::Element_ptr Y,
const size_t incY )

```

$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$

#### Parameters

|         |               |                     |
|---------|---------------|---------------------|
|         | $F$           | field               |
|         | $N$           | size of the vectors |
|         | $\alpha$      | scalar              |
| in      | $X$           | vector in $F$       |
|         | $\text{incX}$ | stride of $X$       |
|         | $\beta$       | scalar              |
| in, out | $Y$           | vector in $F$       |
|         | $\text{incY}$ | stride of $Y$       |

#### Note

this is a catlas function

#### 15.1.3.145 fdot() [9/11]

```

Field::Element fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::ConstElement_ptr Y,
    const size_t incY,
    const ParSeqHelper::Parallel< Cut, Param > par ) [inline]

```

#### 15.1.3.146 fswap() [1/2]

```

void fswap (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY )

```

$\text{fswap} : X \leftrightarrow Y.$

**Bug** use `cblas_dswap` when double

## Parameters

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $N$    | size of the vectors |
| $X$    | vector in $F$       |
| $incX$ | stride of $X$       |
| $Y$    | vector in $F$       |
| $incY$ | stride of $Y$       |

**15.1.3.147 fzero()** [2/4]

```
void fzero (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

$fzero : A \leftarrow 0.$

## Parameters

|       |                        |
|-------|------------------------|
| $F$   | field                  |
| $m$   | number of rows to zero |
| $n$   | number of cols to zero |
| $A$   | matrix in $F$          |
| $lda$ | stride of $A$          |

## Warning

may be buggy if Element is larger than int

**15.1.3.148 frand()** [2/2]

```
void frand (
    const Field & F,
    RandIter & G,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

$frand : A \leftarrow random.$

## Parameters

|            |                             |
|------------|-----------------------------|
| <i>F</i>   | field                       |
| <i>G</i>   | randomiterator              |
| <i>m</i>   | number of rows to randomize |
| <i>n</i>   | number of cols to randomize |
| <i>A</i>   | matrix in F                 |
| <i>lda</i> | stride of A                 |

**15.1.3.149 fequal()** [2/4]

```
bool fequal (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb )
```

fequal : test  $A = B$ .

## Parameters

|            |                        |
|------------|------------------------|
| <i>F</i>   | field                  |
| <i>m</i>   | row dimension          |
| <i>n</i>   | column dimension       |
| <i>A</i>   | m x n matrix in F      |
| <i>lda</i> | leading dimension of A |
| <i>B</i>   | m x n matrix in F      |
| <i>ldb</i> | leading dimension of B |

**15.1.3.150 fiszero()** [2/4]

```
bool fiszero (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr A,
    const size_t lda )
```

fiszero : test  $A = 0$ .

## Parameters

|          |               |
|----------|---------------|
| <i>F</i> | field         |
| <i>m</i> | row dimension |

## Parameters

|       |                          |
|-------|--------------------------|
| $n$   | column dimension         |
| $A$   | m x n matrix in $F$      |
| $lda$ | leading dimension of $A$ |

**15.1.3.151 fidentity()** [1/4]

```
void fidentity (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element & d )
```

creates a diagonal matrix

**15.1.3.152 fidentity()** [2/4]

```
void fidentity (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

creates a diagonal matrix

**15.1.3.153 finit()** [6/8]

```
void finit (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )
```

finit Initializes  $A$  in  $F^{\$}$ .

## Parameters

|       |                |
|-------|----------------|
| $F$   | field          |
| $m$   | number of rows |
| $n$   | number of cols |
| $A$   | matrix in $F$  |
| $lda$ | stride of $A$  |

**15.1.3.154 fconvert()** [2/3]

```

void fconvert (
    const Field & F,
    const size_t m,
    const size_t n,
    OtherElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb )

```

$\text{fconvert } A \leftarrow B \bmod F.$

**Parameters**

|       |                        |
|-------|------------------------|
| $F$   | field                  |
| $m$   | number of rows         |
| $n$   | number of cols         |
| $A$   | matrix in OtherElement |
| $lda$ | stride of A            |
| $B$   | matrix in F            |
| $ldb$ | stride of B            |

**Todo** check if  $n == lda$

**15.1.3.155 fnegin()** [2/4]

```

void fnegin (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda )

```

$\text{fnegin } A \leftarrow -A.$

**Parameters**

|       |                |
|-------|----------------|
| $F$   | field          |
| $m$   | number of rows |
| $n$   | number of cols |
| $A$   | matrix in F    |
| $lda$ | stride of A    |

**Todo** check if  $n == lda$

**15.1.3.156 fneg()** [2/4]

```

void fneg (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::Element_ptr A,
    const size_t lda )

```

$\text{fneg } A \leftarrow -B.$

**Parameters**

|       |                |
|-------|----------------|
| $F$   | field          |
| $m$   | number of rows |
| $n$   | number of cols |
| $A$   | matrix in F    |
| $lda$ | stride of A    |

**Todo** check if  $n == lda$

**15.1.3.157 faxpby()** [2/2]

```

void faxpby (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr X,
    const size_t ldx,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t ldy )

```

$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$

**Parameters**

|         |          |                        |
|---------|----------|------------------------|
|         | $F$      | field                  |
|         | $m$      | row dimension          |
|         | $n$      | column dimension       |
|         | $\alpha$ | scalar                 |
| in      | $X$      | vector in F            |
|         | $ldx$    | leading dimension of X |
|         | $\beta$  | scalar                 |
| in, out | $Y$      | vector in F            |
|         | $ldy$    | leading dimension of Y |



## Note

this is a catlas function

**15.1.3.158 fmove()** [1/2]

```
void fmove (
    const Field & F,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb )
```

fmove :  $A \leftarrow B$  and  $B \leftarrow 0$ .

## Parameters

|       |                        |
|-------|------------------------|
| $F$   | field                  |
| $m$   | number of rows to copy |
| $n$   | number of cols to copy |
| $A$   | matrix in $F$          |
| $lda$ | stride of $A$          |
| $B$   | matrix in $F$          |
| $ldb$ | stride of $B$          |

**15.1.3.159 bitsize()**

```
size_t bitsize (
    const Field & F,
    size_t M,
    size_t N,
    const typename Field::ConstElement_ptr A,
    size_t lda ) [inline]
```

bitsize: Computes the largest bitsize of the matrix' coefficients.

If the matrix is over a modular prime field, it returns the bitsize of the largest element (in a bsolute value)

## Parameters

|        |   |
|--------|---|
| $F$    | field   |
| $M$    | rows  |
| $N$    | cols  |
| $incX$ | stride of $X$   |
| $A$    | a matrix of leading dimension $lda$ and size $M \times N$ |
| $lda$  | leading dimension of $A$                                  |

**15.1.3.160 bitsize< Givaro::ZRing< Givaro::Integer > >()**

```
size_t bitsize< Givaro::ZRing< Givaro::Integer > > (
    const Givaro::ZRing< Givaro::Integer > & F,
    size_t M,
    size_t N,
    const Givaro::Integer * A,
    size_t lda ) [inline]
```

**15.1.3.161 ftrmv()**

```
void ftrmv (
    const Field & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    int incX )
```

ftrsm: TRIangular Matrix Vector prodcut Computes  $X \leftarrow \text{op}(A)X$

**Parameters**

|               |   |
|---------------|---|
| <i>F</i>      | field   |
| <i>X</i>      | vector of size N on a field F                     |
| <i>incX</i>   | stride of X                                       |
| <i>A</i>      | a matrix of leading dimension lda and size N      |
| <i>lda</i>    | leading dimension of A                            |
| <i>N</i>      | number of rows and columns of A                   |
| <i>TransA</i> | if TransA==FflasTrans then $\text{op}(A) = A^T$ . |
| <i>Diag</i>   | if Diag==FflasUnit then A is unit diagonal.       |
| <i>Uplo</i>   | if Uplo==FflasUpper then A is upper triangular    |

**15.1.3.162 ftrsm() [6/9]**

```
void ftrsm (
    const Field & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
```

```

const FFLAS_DIAG Diag,
const size_t M,
const size_t N,
const typename Field::Element alpha,
typename Field::ConstElement_ptr A,
const size_t lda,
typename Field::Element_ptr B,
const size_t ldb )

```

ftsm: **T**riangular **S**ystem solve with **M**atrix.

Computes  $B \leftarrow \alpha \text{op}(A^{-1})B$  or  $B \leftarrow \alpha B \text{op}(A^{-1})$ .

#### Parameters

|               |   |
|---------------|---|
| <i>F</i>      | field   |
| <i>Side</i>   | if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A^{-1})B$ is computed.                         |
| <i>Uplo</i>   | if Uplo==FflasUpper then A is upper triangular  |
| <i>TransA</i> | if TransA==FflasTrans then $\text{op}(A) = A^t$ .   |
| <i>Diag</i>   | if Diag==FflasUnit then A is unit.  |
| <i>M</i>      | rows of B   |
| <i>N</i>      | cols of B   |
| <i>alpha</i>  | scalar  |
| <i>A</i>      | triangular invertible matrix. If Side==FflasLeft then A is $N \times N$ , otherwise A is $M \times M$ |
| <i>lda</i>    | leading dim of A  |
| <i>B</i>      | matrix of size MxN  |
| <i>ldb</i>    | leading dim of B  |

**Bug**  $\alpha$  must be non zero.

#### 15.1.3.163 pfgemm() [1/7]

```

Field::Element_ptr pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t numthreads = 0 )

```

**15.1.3.164 pfgemm\_1D\_rec()**

```
Field::Element * pfgemm_1D_rec (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    size_t seuil )
```

**15.1.3.165 pfgemm\_2D\_rec()**

```
Field::Element * pfgemm_2D_rec (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    size_t seuil )
```

**15.1.3.166 pfgemm\_3D\_rec()**

```
Field::Element * pfgemm_3D_rec (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
```

```

const typename Field::Element_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
size_t seuil,
size_t * x )

```

### 15.1.3.167 pfgemm\_3D\_rec2()

```

Field::Element_ptr pfgemm_3D_rec2 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    size_t seuil,
    size_t * x )

```

### 15.1.3.168 fgemm() [19/23]

```

std::enable_if<!std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag
> >::value, typename Field::Element_ptr >::type fgemm (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE ta,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat,
Param > > & H ) [inline]

```

**15.1.3.169 ftrsm()** [7/9]

```
Field::Element_ptr ftrsm (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_TRANSPOSE TA,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel< Cut, Param > > &
    H ) [inline]
```

**15.1.3.170 ftrsm()** [8/9]

```
Field::Element_ptr ftrsm (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_TRANSPOSE TA,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > > & H
) [inline]
```

**15.1.3.171 fspmv()** [1/2]

```
void fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    const typename Field::Element & beta,
    typename Field::Element_ptr y ) [inline]
```

**15.1.3.172 fspmm()**

```
void fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    const typename Field::Element & beta,
    typename Field::Element_ptr y,
    int ldy ) [inline]
```

**15.1.3.173 sparse\_init() [1/16]**

```
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::COO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.174 sparse\_init() [2/16]**

```
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.175 sparse\_delete() [1/12]**

```
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::COO > & A ) [inline]
```

**15.1.3.176 sparse\_delete()** [2/12]

```
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A ) [inline]
```

**15.1.3.177 sparse\_init()** [3/16]

```
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.178 sparse\_init()** [4/16]

```
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.179 sparse\_delete()** [3/12]

```
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR > & A ) [inline]
```

**15.1.3.180 sparse\_delete()** [4/12]

```
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A ) [inline]
```



**15.1.3.181 sparse\_print()** [1/3]

```
std::ostream & sparse_print (
    std::ostream & os,
    const Sparse< Field, SparseMatrix_t::CSR > & A ) [inline]
```

**15.1.3.182 sparse\_init()** [5/16]

```
void sparse_init (
    const Givaro::Modular< Givaro::Integer > & F,
    Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR > & A,
    const IndexT * row,
    const IndexT * col,
    Givaro::Integer * dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.183 sparse\_init()** [6/16]

```
void sparse_init (
    const Givaro::ZRing< Givaro::Integer > & F,
    Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    Givaro::Integer * dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.184 sparse\_init()** [7/16]

```
void sparse_init (
    const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > & F,
    Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR_ZO >
& A,
    const IndexT * row,
    const IndexT * col,
    typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.185 sparse\_init()** [8/16]

```

void sparse_init (
    const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > & F,
    Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR > &
A,

    const IndexT * row,
    const IndexT * col,
    typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.186 sparse\_delete()** [5/12]

```

void sparse_delete (
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A ) [inline]

```

**15.1.3.187 sparse\_init()** [9/16]

```

void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.188 sparse\_init()** [10/16]

```

void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]

```

**15.1.3.189 sparse\_init()** [11/16]

```
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.190 sparse\_delete()** [6/12]

```
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL > & A ) [inline]
```

**15.1.3.191 sparse\_delete()** [7/12]

```
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A ) [inline]
```

**15.1.3.192 sparse\_init()** [12/16]

```
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.193 sparse\_init()** [13/16]

```
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.194 sparse\_delete()** [8/12]

```
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A ) [inline]
```

**15.1.3.195 sparse\_delete()** [9/12]

```
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A ) [inline]
```

**15.1.3.196 sparse\_print()** [2/3]

```
void sparse_print (
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A ) [inline]
```

**15.1.3.197 sparse\_delete()** [10/12]

```
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A ) [inline]
```

**15.1.3.198 sparse\_init()** [14/16]

```
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.199 operator<<()**

```
std::ostream & operator<< (
    std::ostream & os,
    const Sparse< _Field, SparseMatrix_t::HYB_ZO > & A )
```

**15.1.3.200 readSmsFormat()**

```
void readSmsFormat (
    const std::string & path,
    const Field & f,
    index_t *& row,
    index_t *& col,
    typename Field::Element_ptr & val,
    index_t & rowdim,
    index_t & coldim,
    uint64_t & nnz )
```

**15.1.3.201 readSprFormat()**

```
void readSprFormat (
    const std::string & path,
    const Field & f,
    index_t *& row,
    index_t *& col,
    typename Field::Element_ptr & val,
    index_t & rowdim,
    index_t & coldim,
    uint64_t & nnz )
```

**15.1.3.202 getDataType() [1/4]**

```
std::enable_if< std::is_integral< T >::value, int > getDataType ( )
```

**15.1.3.203 getDataType() [2/4]**

```
std::enable_if< std::is_floating_point< T >::value, int > getDataType ( )
```

**15.1.3.204 getDataType() [3/4]**

```
std::enable_if< std::is_same< T, mpz_t >::value, int > getDataType ( )
```

**15.1.3.205 getDataType() [4/4]**

```
int getDataType ( )
```

**15.1.3.206 readMachineType()**

```
void readMachineType (
    const Field & F,
    typename Field::Element & modulo,
    typename Field::Element_ptr val,
    std::ifstream & file,
    const uint64_t dims,
    const mask_t data_type,
    const mask_t field_desc )
```

**15.1.3.207 readDnsFormat()**

```
void readDnsFormat (
    const std::string & path,
    const Field & F,
    index_t & rowdim,
    index_t & coldim,
    typename Field::Element_ptr & val )
```

**15.1.3.208 writeDnsFormat()**

```
void writeDnsFormat (
    const std::string & path,
    const Field & F,
    const index_t & rowdim,
    const index_t & coldim,
    typename Field::Element_ptr A,
    index_t ldA )
```

**15.1.3.209 fspmv()** [2/2]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ) [inline]
```

**15.1.3.210 sparse\_delete()** [11/12]

```
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::SELL > & A ) [inline]
```

**15.1.3.211 sparse\_delete()** [12/12]

```
void sparse_delete (
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A ) [inline]
```

**15.1.3.212 sparse\_print()** [3/3]

```
void sparse_print (
    const Sparse< Field, SparseMatrix_t::SELL > & A ) [inline]
```

**15.1.3.213 sparse\_init()** [15/16]

```
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::SELL > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz,
    uint64_t sigma = 0 ) [inline]
```

**15.1.3.214 sparse\_init()** [16/16]

```
void sparse_init (
    const Field & F,
    Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    const IndexT * row,
    const IndexT * col,
    typename Field::ConstElement_ptr dat,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz ) [inline]
```

**15.1.3.215 computeDeviation()**

```
double computeDeviation (
    It begin,
    It end )
```

**15.1.3.216 getStat()**

```
StatsMatrix getStat (
    const Field & F,
    const index_t * row,
    const index_t * col,
    typename Field::ConstElement_ptr val,
    uint64_t rowdim,
    uint64_t coldim,
    uint64_t nnz )
```

**15.1.3.217 fflas\_delete() [1/4]**

```
void fflas_delete (
    FFPACK::rns_double_elt_ptr A ) [inline]
```

**15.1.3.218 fflas\_delete() [2/4]**

```
void fflas_delete (
    FFPACK::rns_double_elt_cstptr A ) [inline]
```

**15.1.3.219 fflas\_new() [1/7]**

```
FFPACK::rns_double_elt_ptr fflas_new (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const Alignment align ) [inline]
```

**15.1.3.220 fflas\_new() [2/7]**

```
FFPACK::rns_double_elt_ptr fflas_new (
    const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const Alignment align ) [inline]
```



**15.1.3.221 finit\_rns()** [1/2]

```
void finit_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename RNS::Element_ptr A )
```

**15.1.3.222 finit\_trans\_rns()**

```
void finit_trans_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename RNS::Element_ptr A )
```

**15.1.3.223 fconvert\_rns()** [1/2]

```
void fconvert_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename RNS::ConstElement_ptr A )
```

**15.1.3.224 fconvert\_trans\_rns()**

```
void fconvert_trans_rns (
    const FFPACK::RNSIntegerMod< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename RNS::ConstElement_ptr A )
```

**15.1.3.225 fflas\_new()** [3/7]

```
FFPACK::rns_double_elt_ptr fflas_new (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const Alignment align ) [inline]
```

**15.1.3.226 fflas\_new()** [4/7]

```
FFPACK::rns_double_elt_ptr fflas_new (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    const size_t m,
    const size_t n,
    const Alignment align ) [inline]
```

**15.1.3.227 finit\_rns()** [2/2]

```
void finit_rns (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t m,
    const size_t n,
    size_t k,
    const Givaro::Integer * B,
    const size_t ldb,
    typename FFPACK::RNSInteger< RNS >::Element_ptr A )
```

**15.1.3.228 fconvert\_rns()** [2/2]

```
void fconvert_rns (
    const FFPACK::RNSInteger< RNS > & F,
    const size_t m,
    const size_t n,
    Givaro::Integer alpha,
    Givaro::Integer * B,
    const size_t ldb,
    typename FFPACK::RNSInteger< RNS >::ConstElement_ptr A )
```

**15.1.3.229 freduce()** [7/10]

```
template INST_OR_DECL void freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )
```

freduce  $x \leftarrow x \bmod F$ .

## Parameters

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $n$    | size of the vectors |
| $X$    | vector in $F$       |
| $incX$ | stride of $X$       |

**Bug** use `cblas_(d)scal` when possible

15.1.3.230 `freduce()` [8/10]

```
template INST_OR_DECL void freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

`freduce`  $x \leftarrow y \bmod F$ .

## Parameters

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $n$    | size of the vectors |
| $Y$    | vector of Element   |
| $incY$ | stride of $Y$       |
| $X$    | vector in $F$       |
| $incX$ | stride of $X$       |

**Bug** use `cblas_(d)scal` when possible

15.1.3.231 `finit()` [7/8]

```
template INST_OR_DECL void finit (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

`finit`  $x \leftarrow y \bmod F$ .

## Parameters

|        |                        |
|--------|------------------------|
| $F$    | field                  |
| $n$    | size of the vectors    |
| $Y$    | vector of OtherElement |
| $incY$ | stride of $Y$          |
| $X$    | vector in $F$          |
| $incX$ | stride of $X$          |

**Bug** use `cblas_(d)scal` when possible

15.1.3.232 `fconvert()` [3/3]

```
template INST_OR_DECL void fconvert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT * Y,
    const size_t incY )
```

`fconvert`  $x \leftarrow y \bmod F$ .

## Parameters

|        |                        |
|--------|------------------------|
| $F$    | field                  |
| $n$    | size of the vectors    |
| $Y$    | vector of $F$          |
| $incY$ | stride of $Y$          |
| $X$    | vector in OtherElement |
| $incX$ | stride of $X$          |

**Bug** use `cblas_(d)scal` when possible

15.1.3.233 `fnegin()` [3/4]

```
template INST_OR_DECL void fnegin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )
```

`fnegin`  $x \leftarrow -x$ .

## Parameters

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $n$    | size of the vectors |
| $X$    | vector in $F$       |
| $incX$ | stride of $X$       |

**Bug** use `cblas_(d)scal` when possible

15.1.3.234 `fneg()` [3/4]

```
template INST_OR_DECL void fneg (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

`fneg`  $x \leftarrow -y$ .

## Parameters

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $n$    | size of the vectors |
| $X$    | vector in $F$       |
| $incX$ | stride of $X$       |
| $Y$    | vector in $F$       |
| $incY$ | stride of $Y$       |

**Bug** use `cblas_(d)scal` when possible

15.1.3.235 `fzero()` [3/4]

```
template INST_OR_DECL void fzero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    FFLAS_ELT * X,
    const size_t incX )
```

`fzero` :  $A \leftarrow 0$ .

## Parameters

|        |                            |
|--------|----------------------------|
| $F$    | field                      |
| $n$    | number of elements to zero |
| $X$    | vector in $F$              |
| $incX$ | stride of $X$              |

**15.1.3.236** `fiszero()` [3/4]

```
template INST_OR_DECL bool fiszero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * X,
    const size_t incX )
```

`fiszero` : test  $X = 0$ .

## Parameters

|        |                  |
|--------|------------------|
| $F$    | field            |
| $n$    | vector dimension |
| $X$    | vector in $F$    |
| $incX$ | increment of $X$ |

**15.1.3.237** `fequal()` [3/4]

```
template INST_OR_DECL bool fequal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT * Y,
    const size_t incY )
```

`fequal` : test  $X = Y$ .

## Parameters

|        |                  |
|--------|------------------|
| $F$    | field            |
| $n$    | vector dimension |
| $X$    | vector in $F$    |
| $incX$ | increment of $X$ |
| $Y$    | vector in $F$    |
| $incY$ | increment of $Y$ |

**15.1.3.238 fassign()** [9/10]

```
template INST_OR_DECL void fassign (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * Y,
    const size_t incY,
    FFLAS_ELT * X,
    const size_t incX )
```

fassign :  $x \leftarrow y$ .

X is preallocated

**Todo** variant for triagular matrix

**Parameters**

|     |        |                     |
|-----|--------|---------------------|
|     | $F$    | field               |
|     | $N$    | size of the vectors |
| out | $X$    | vector in F         |
|     | $incX$ | stride of X         |
| in  | $Y$    | vector in F         |
|     | $incY$ | stride of Y         |

**15.1.3.239 fscaln()** [9/10]

```
template INST_OR_DECL void fscaln (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT alpha,
    FFLAS_ELT * X,
    const size_t incX )
```

fscaln  $x \leftarrow \alpha \cdot x$ .

**Parameters**

|         |                     |
|---------|---------------------|
| $F$     | field               |
| $n$     | size of the vectors |
| $alpha$ | scalar              |
| $X$     | vector in F         |
| $incX$  | stride of X         |

**Bug** use cblas\_(d)scal when possible

**Todo** check if comparison with +/-1,0 is necessary.

**15.1.3.240 fscal()** [9/10]

```
template INST_OR_DECL void fscal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * X,
    const size_t incX,
    FFLAS_ELT * Y,
    const size_t incY )
```

$\text{fscal } y \leftarrow \alpha \cdot x.$

**Parameters**

|     |               |                     |
|-----|---------------|---------------------|
|     | $F$           | field               |
|     | $n$           | size of the vectors |
|     | $\alpha$      | scalar              |
| in  | $X$           | vector in F         |
|     | $\text{incX}$ | stride of X         |
| out | $Y$           | vector in F         |
|     | $\text{incY}$ | stride of Y         |

**Bug** use cblas\_(d)scal when possible

**Todo** check if comparison with +/-1,0 is necessary.

**15.1.3.241 faxpy()** [5/6]

```
template INST_OR_DECL void faxpy (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * X,
    const size_t incX,
    FFLAS_ELT * Y,
    const size_t incY )
```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

**Parameters**

|         |               |                     |
|---------|---------------|---------------------|
|         | $F$           | field               |
|         | $N$           | size of the vectors |
|         | $\alpha$      | scalar              |
| in      | $X$           | vector in F         |
|         | $\text{incX}$ | stride of X         |
| in, out | $Y$           | vector in F         |
|         | $\text{incY}$ | stride of Y         |



**15.1.3.242 fdot()** [10/11]

```
template INST_OR_DECL FFLAS_ELT fdot (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT * Y,
    const size_t incY )
```

faxpby :  $y \leftarrow \alpha \cdot x + \beta \cdot y$ .

**Parameters**

|         |          |                     |
|---------|----------|---------------------|
|         | $F$      | field               |
|         | $N$      | size of the vectors |
|         | $\alpha$ | scalar              |
| in      | $X$      | vector in F         |
|         | $incX$   | stride of X         |
|         | $\beta$  | scalar              |
| in, out | $Y$      | vector in F         |
|         | $incY$   | stride of Y         |

**Note**

this is a catlas function

fdot: dot product  $x^T y$ .

**Parameters**

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $N$    | size of the vectors |
| $X$    | vector in F         |
| $incX$ | stride of X         |
| $Y$    | vector in F         |
| $incY$ | stride of Y         |

**15.1.3.243 fswap()** [2/2]

```
template INST_OR_DECL void fswap (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    FFLAS_ELT * X,
    const size_t incX,
```

```

FFLAS_ELT * Y,
const size_t incY )

```

fswap:  $X \leftrightarrow Y$ .

**Bug** use cblas\_dswap when double

#### Parameters

|        |                     |
|--------|---------------------|
| $F$    | field               |
| $N$    | size of the vectors |
| $X$    | vector in $F$       |
| $incX$ | stride of $X$       |
| $Y$    | vector in $F$       |
| $incY$ | stride of $Y$       |

#### 15.1.3.244 fadd() [5/8]

```

template INST_OR_DECL void fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t inca,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )

```

#### 15.1.3.245 fsub() [3/4]

```

template INST_OR_DECL void fsub (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t inca,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )

```

**15.1.3.246 faddin()** [3/4]

```
template INST_OR_DECL void faddin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )
```

**15.1.3.247 fadd()** [6/8]

```
template INST_OR_DECL void fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t inca,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * B,
    const size_t incb,
    FFLAS_ELT * C,
    const size_t incc )
```

**15.1.3.248 fassign()** [10/10]

```
template INST_OR_DECL void fassign (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

fassign :  $A \leftarrow B$ .

**Parameters**

|       |                        |
|-------|------------------------|
| $F$   | field                  |
| $m$   | number of rows to copy |
| $n$   | number of cols to copy |
| $A$   | matrix in F            |
| $lda$ | stride of A            |
| $B$   | vector in F            |
| $ldb$ | stride of B            |

**15.1.3.249 fzero()** [4/4]

```
template INST_OR_DECL void fzero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

fzero :  $A \leftarrow 0$ .

**Parameters**

|       |                        |
|-------|------------------------|
| $F$   | field                  |
| $m$   | number of rows to zero |
| $n$   | number of cols to zero |
| $A$   | matrix in F            |
| $lda$ | stride of A            |

**Warning**

may be buggy if Element is larger than int

**15.1.3.250 fequal()** [4/4]

```
template INST_OR_DECL bool fequal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb )
```

fequal : test  $A = B$ .

**Parameters**

|       |                        |
|-------|------------------------|
| $F$   | field                  |
| $m$   | row dimension          |
| $n$   | column dimension       |
| $A$   | m x n matrix in F      |
| $lda$ | leading dimension of A |
| $B$   | m x n matrix in F      |
| $ldb$ | leading dimension of B |

**15.1.3.251 fiszero()** [4/4]

```
template INST_OR_DECL bool fiszero (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * A,
    const size_t lda )
```

**fiszero** : test  $A = 0$ .

**Parameters**

|       |                        |
|-------|------------------------|
| $F$   | field                  |
| $m$   | row dimension          |
| $n$   | column dimension       |
| $A$   | m x n matrix in F      |
| $lda$ | leading dimension of A |

**15.1.3.252 fidentity()** [3/4]

```
template INST_OR_DECL void fidentity (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT & d )
```

creates a diagonal matrix

**15.1.3.253 fidentity()** [4/4]

```
template INST_OR_DECL void fidentity (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

creates a diagonal matrix

**15.1.3.254 freduce()** [9/10]

```
template INST_OR_DECL void freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

**freduce**  $A \leftarrow A \bmod F$ .

## Parameters

|       |                |
|-------|----------------|
| $F$   | field          |
| $m$   | number of rows |
| $n$   | number of cols |
| $A$   | matrix in $F$  |
| $lda$ | stride of $A$  |

**15.1.3.255** **freduce()** [10/10]

```
template INST_OR_DECL void freduce (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{freduce } A \leftarrow B \bmod F.$

## Parameters

|       |                   |
|-------|-------------------|
| $F$   | field             |
| $m$   | number of rows    |
| $n$   | number of cols    |
| $A$   | matrix in $F$     |
| $lda$ | stride of $A$     |
| $B$   | matrix in Element |
| $ldb$ | stride of $B$     |

**15.1.3.256** **finit()** [8/8]

```
template INST_OR_DECL void finit (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{finit } A \leftarrow B \bmod F.$

## Parameters

|     |       |
|-----|-------|
| $F$ | field |
|-----|-------|

## Parameters

|       |                |
|-------|----------------|
| $m$   | number of rows |
| $n$   | number of cols |
| $A$   | matrix in $F$  |
| $lda$ | stride of $A$  |
| $B$   | matrix in $F$  |
| $ldb$ | stride of $B$  |

## 15.1.3.257 fnegin() [4/4]

```
template INST_OR_DECL void fnegin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{fnegin } A \leftarrow -A.$

## Parameters

|       |                |
|-------|----------------|
| $F$   | field          |
| $m$   | number of rows |
| $n$   | number of cols |
| $A$   | matrix in $F$  |
| $lda$ | stride of $A$  |

## 15.1.3.258 fneg() [4/4]

```
template INST_OR_DECL void fneg (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{fneg } A \leftarrow -B.$

## Parameters

|       |                |
|-------|----------------|
| $F$   | field          |
| $m$   | number of rows |
| $n$   | number of cols |
| $A$   | matrix in $F$  |
| $lda$ | stride of $A$  |

**15.1.3.259 fscaln()** [10/10]

```
template INST_OR_DECL void fscaln (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT alpha,
    FFLAS_ELT * A,
    const size_t lda )
```

$\text{fscaln } A \leftarrow a \cdot A.$

**Parameters**

|          |                  |
|----------|------------------|
| $F$      | field            |
| $m$      | number of rows   |
| $n$      | number of cols   |
| $\alpha$ | homotecie scalar |
| $A$      | matrix in F      |
| $lda$    | stride of A      |

**15.1.3.260 fscal()** [10/10]

```
template INST_OR_DECL void fscal (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

$\text{fscal } B \leftarrow a \cdot A.$

**Parameters**

|     |          |                  |
|-----|----------|------------------|
|     | $F$      | field            |
|     | $m$      | number of rows   |
|     | $n$      | number of cols   |
|     | $\alpha$ | homotecie scalar |
| in  | $A$      | matrix in F      |
|     | $lda$    | stride of A      |
| out | $B$      | matrix in F      |
|     | $ldb$    | stride of B      |



**15.1.3.261 faxpy()** [6/6]

```
template INST_OR_DECL void faxpy (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * X,
    const size_t ldx,
    FFLAS_ELT * Y,
    const size_t ldy )
```

$\text{faxpy} : y \leftarrow \alpha \cdot x + y.$

**Parameters**

|         |          |                          |
|---------|----------|--------------------------|
|         | $F$      | field                    |
|         | $m$      | row dimension            |
|         | $n$      | column dimension         |
|         | $\alpha$ | scalar                   |
| in      | $X$      | vector in $F$            |
|         | $ldx$    | leading dimension of $X$ |
| in, out | $Y$      | vector in $F$            |
|         | $ldy$    | leading dimension of $Y$ |

**15.1.3.262 fmove()** [2/2]

```
template INST_OR_DECL void fmove (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t m,
    const size_t n,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

$\text{faxpby} : y \leftarrow \alpha \cdot x + \beta \cdot y.$

**Parameters**

|         |          |                          |
|---------|----------|--------------------------|
|         | $F$      | field                    |
|         | $m$      | row dimension            |
|         | $n$      | column dimension         |
|         | $\alpha$ | scalar                   |
| in      | $X$      | vector in $F$            |
|         | $ldx$    | leading dimension of $X$ |
|         | $\beta$  | scalar                   |
| in, out | $Y$      | vector in $F$            |
|         | $ldy$    | leading dimension of $Y$ |

**Note**

this is a catlas function

fmove :  $A \leftarrow B$  and  $B \leftarrow 0$ .

**Parameters**

|       |                        |
|-------|------------------------|
| $F$   | field                  |
| $m$   | number of rows to copy |
| $n$   | number of cols to copy |
| $A$   | matrix in $F$          |
| $lda$ | stride of $A$          |
| $B$   | vector in $F$          |
| $ldb$ | stride of $B$          |

**15.1.3.263 fadd() [7/8]**

```
template INST_OR_DECL void fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )
```

fadd : matrix addition.

Computes  $C = A + B$ .

**Parameters**

|       |                                   |
|-------|-----------------------------------|
| $F$   | field                             |
| $M$   | rows                              |
| $N$   | cols                              |
| $A$   | dense matrix of size $M \times N$ |
| $lda$ | leading dimension of $A$          |
| $B$   | dense matrix of size $M \times N$ |
| $ldb$ | leading dimension of $B$          |
| $C$   | dense matrix of size $M \times N$ |
| $ldc$ | leading dimension of $C$          |

**15.1.3.264 fsub() [4/4]**

```
template INST_OR_DECL void fsub (
```

```

const FFLAS_FIELD< FFLAS_ELT > & F,
const size_t M,
const size_t N,
const FFLAS_ELT * A,
const size_t lda,
const FFLAS_ELT * B,
const size_t ldb,
FFLAS_ELT * C,
const size_t ldc )

```

**fsub** : matrix subtraction.

Computes  $C = A - B$ .

#### Parameters

|            |                          |
|------------|--------------------------|
| <i>F</i>   | field                    |
| <i>M</i>   | rows                     |
| <i>N</i>   | cols                     |
| <i>A</i>   | dense matrix of size MxN |
| <i>lda</i> | leading dimension of A   |
| <i>B</i>   | dense matrix of size MxN |
| <i>ldb</i> | leading dimension of B   |
| <i>C</i>   | dense matrix of size MxN |
| <i>ldc</i> | leading dimension of C   |

#### 15.1.3.265 fsubin() [3/3]

```

template INST_OR_DECL void fsubin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )

```

**fsubin**  $C = C - B$

#### 15.1.3.266 fadd() [8/8]

```

template INST_OR_DECL void fadd (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT alpha,

```

```

    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )

```

fadd : matrix addition with scaling.

Computes  $C = A + \alpha B$ .

#### Parameters

|              |                          |
|--------------|--------------------------|
| <i>F</i>     | field                    |
| <i>M</i>     | rows                     |
| <i>N</i>     | cols                     |
| <i>A</i>     | dense matrix of size MxN |
| <i>lda</i>   | leading dimension of A   |
| <i>alpha</i> | some scalar              |
| <i>B</i>     | dense matrix of size MxN |
| <i>ldb</i>   | leading dimension of B   |
| <i>C</i>     | dense matrix of size MxN |
| <i>ldc</i>   | leading dimension of C   |

#### 15.1.3.267 faddin() [4/4]

```

template INST_OR_DECL void faddin (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT * B,
    const size_t ldb,
    FFLAS_ELT * C,
    const size_t ldc )

```

faddin

#### 15.1.3.268 fgemv() [17/19]

```

template INST_OR_DECL FFLAS_ELT * fgemv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE TransA,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * X,
    const size_t incX,
    const FFLAS_ELT beta,

```

```
FFLAS_ELT * Y,  
const size_t incY )
```

finite prime FFLAS\_FIELD<FFLAS\_ELT> GEneral Matrix Vector multiplication.

Computes  $Y \leftarrow \alpha \text{op}(A)X + \beta Y$ .

## Parameters

|     |               |  |
|-----|---------------|--|
|     | <i>F</i>      | field  |
|     | <i>TransA</i> | if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$ . |
|     | <i>M</i>      | rows   |
|     | <i>N</i>      | cols   |
|     | <i>alpha</i>  | scalar   |
|     | <i>A</i>      | dense matrix of size MxN                                       |
|     | <i>lda</i>    | leading dimension of A   |
|     | <i>X</i>      | dense vector of size N   |
|     | <i>incX</i>   | stride of X  |
|     | <i>beta</i>   | scalar   |
| out | <i>Y</i>      | dense vector of size M   |
|     | <i>incY</i>   | stride of Y  |

## 15.1.3.269 fger() [12/12]

```
template INST_OR_DECL void fger (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * x,
    const size_t incx,
    const FFLAS_ELT * y,
    const size_t incy,
    FFLAS_ELT * A,
    const size_t lda )
```

fger: rank one update of a general matrix

Computes  $A \leftarrow \alpha x y^T + A$

## Parameters

|         |              |  |
|---------|--------------|--|
|         | <i>F</i>     | field  |
|         | <i>M</i>     | rows   |
|         | <i>N</i>     | cols   |
|         | <i>alpha</i> | scalar   |
| in, out | <i>A</i>     | dense matrix of size MxN and leading dimension lda |
|         | <i>lda</i>   | leading dimension of A                             |
|         | <i>x</i>     | dense vector of size M                             |
|         | <i>incx</i>  | stride of X  |
|         | <i>y</i>     | dense vector of size N                             |
|         | <i>incy</i>  | stride of Y  |

**15.1.3.270 ftrsv()** [2/2]

```
template INST_OR_DECL void ftrsv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    int incX )
```

ftrsv: **TR**angular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$

**Parameters**

|               |  |
|---------------|--|
| <i>F</i>      | field  |
| <i>X</i>      | vector of size N on a field F                      |
| <i>incX</i>   | stride of X  |
| <i>A</i>      | a matrix of leading dimension lda and size N       |
| <i>lda</i>    | leading dimension of A                             |
| <i>N</i>      | number of rows or columns of A according to TransA |
| <i>TransA</i> | if TransA==FflasTrans then $\text{op}(A) = A^t$ .  |
| <i>Diag</i>   | if Diag==FflasUnit then A is unit.                 |
| <i>Uplo</i>   | if Uplo==FflasUpper then A is upper triangular     |

**15.1.3.271 ftrsm()** [9/9]

```
template INST_OR_DECL void ftrsm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

ftrsm: **TR**angular System solve with **M**atrix.

Computes  $B \leftarrow \alpha \text{op}(A^{-1})B$  or  $B \leftarrow \alpha B \text{op}(A^{-1})$ .

**Parameters**

|             |   |
|-------------|---|
| <i>F</i>    | field   |
| <i>Side</i> | if Side==FflasLeft then $B \leftarrow \alpha \text{op}(A^{-1})B$ is computed. |

## Parameters

|               |  |
|---------------|--|
| <i>Uplo</i>   | if <code>Uplo==FflasUpper</code> then A is upper triangular  |
| <i>TransA</i> | if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$ .   |
| <i>Diag</i>   | if <code>Diag==FflasUnit</code> then A is unit.  |
| <i>M</i>      | rows of B  |
| <i>N</i>      | cols of B  |
| <i>alpha</i>  | scalar   |
| <i>A</i>      | triangular invertible matrix. If <code>Side==FflasLeft</code> then A is $N \times N$ , otherwise A is $M \times M$ |
| <i>lda</i>    | leading dim of A   |
| <i>B</i>      | matrix of size MxN   |
| <i>ldb</i>    | leading dim of B   |

**Bug**  $\alpha$  must be non zero.

15.1.3.272 `ftmm()` [3/3]

```
template INST_OR_DECL void ftmm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_SIDE Side,
    const FFLAS_UPLO Uplo,
    const FFLAS_TRANSPOSE TransA,
    const FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb )
```

`ftmm`: **TR**iangular **M**atrix **M**ultiply.

Computes  $B \leftarrow \alpha \text{op}(A)B$  or  $B \leftarrow \alpha B \text{op}(A)$ .

## Parameters

|               |   |
|---------------|---|
| <i>F</i>      | field   |
| <i>Side</i>   | if <code>Side==FflasLeft</code> then $B \leftarrow \alpha \text{op}(A)B$ is computed.                   |
| <i>Uplo</i>   | if <code>Uplo==FflasUpper</code> then A is upper triangular   |
| <i>TransA</i> | if <code>TransA==FflasTrans</code> then $\text{op}(A) = A^t$ .  |
| <i>Diag</i>   | if <code>Diag==FflasUnit</code> then A is implicitly unit.  |
| <i>M</i>      | rows of B   |
| <i>N</i>      | cols of B   |
| <i>alpha</i>  | scalar  |
| <i>A</i>      | triangular matrix. If <code>Side==FflasLeft</code> then A is $N \times N$ , otherwise A is $M \times M$ |
| <i>lda</i>    | leading dim of A  |
| <i>B</i>      | matrix of size MxN  |
| <i>ldb</i>    | leading dim of B  |



**15.1.3.273 fgemm()** [20/23]

```
template INST_OR_DECL FFLAS_ELT * fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc )
```

fgemm: **F**ield **G**eneral **M**atrix **M**ultiply.

Computes  $C = \alpha \text{op}(A) \times \text{op}(B) + \beta C$  Automatically set Winograd recursion level

**Parameters**

|              |   |
|--------------|---|
| <i>F</i>     | field.  |
| <i>ta</i>    | if <code>ta==FflasTrans</code> then $\text{op}(A) = A^t$ , else $\text{op}(A) = A$ ,                  |
| <i>tb</i>    | same for matrix B   |
| <i>m</i>     | see A   |
| <i>n</i>     | see B   |
| <i>k</i>     | see A   |
| <i>alpha</i> | scalar  |
| <i>beta</i>  | scalar  |
| <i>A</i>     | $\text{op}(A)$ is $m \times k$  |
| <i>B</i>     | $\text{op}(B)$ is $k \times n$  |
| <i>C</i>     | $C$ is $m \times n$   |
| <i>lda</i>   | leading dimension of A  |
| <i>ldb</i>   | leading dimension of B  |
| <i>ldc</i>   | leading dimension of C  |
| <i>w</i>     | recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of $w$ . |

**Warning**

$\alpha$  must be invertible

**15.1.3.274 fgemm()** [21/23]

```
template INST_OR_DECL FFLAS_ELT * fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
```

```

const FFLAS_TRANSPOSE ta,
const FFLAS_TRANSPOSE tb,
const size_t m,
const size_t n,
const size_t k,
const FFLAS_ELT alpha,
const FFLAS_ELT * A,
const size_t lda,
const FFLAS_ELT * B,
const size_t ldb,
const FFLAS_ELT beta,
FFLAS_ELT * C,
const size_t ldc,
const ParSeqHelper::Sequential seq )

```

#### 15.1.3.275 fgemm() [22/23]

```

template INST_OR_DECL FFLAS_ELT * fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc,
    const ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive
> par )

```

#### 15.1.3.276 fgemm() [23/23]

```

template INST_OR_DECL FFLAS_ELT * fgemm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * B,
    const size_t ldb,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,

```

```

        const size_t ldc,
        const ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads
> par )

```

### 15.1.3.277 fsquare() [6/6]

```

template INST_OR_DECL FFLAS_ELT * fsquare (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const FFLAS_ELT alpha,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT beta,
    FFLAS_ELT * C,
    const size_t ldc )

```

fsquare: Squares a matrix.

compute  $C \leftarrow \alpha \text{op}(A) \text{op}(A) + \beta C$  over a FFLAS\_FIELD <FFLAS\_ELT> F Avoid the conversion of B

#### Parameters

|              |   |
|--------------|---|
| <i>ta</i>    | if ta==FflasTrans, $\text{op}(A) = A^T$ . |
| <i>F</i>     | field                                     |
| <i>n</i>     | size of A                                 |
| <i>alpha</i> | scalar                                    |
| <i>beta</i>  | scalar                                    |
| <i>A</i>     | dense matrix of size $n \times n$         |
| <i>lda</i>   | leading dimension of A                    |
| <i>C</i>     | dense matrix of size $n \times n$         |
| <i>ldc</i>   | leading dimension of C                    |

### 15.1.3.278 BlockCuts() [1/2]

```

void BlockCuts (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]

```

**15.1.3.279 BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >()**

```
void BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.280 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >()**

```
void BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.281 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >()**

```
void BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

**15.1.3.282 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >()**

```
void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

**15.1.3.283 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >()**

```
void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.284 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >()**

```
void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t grainsize ) [inline]
```

**15.1.3.285 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >()**

```
void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.286 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >()**

```
void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.287 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >()**

```
void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.288 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >()**

```
void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (
    size_t & RBLOCKSIZE,
    size_t & CBLOCKSIZE,
    const size_t m,
    const size_t n,
    const size_t numthreads ) [inline]
```

**15.1.3.289 BlockCuts()** [2/2]

```

void BlockCuts (
    size_t & rowBlockSize,
    size_t & colBlockSize,
    size_t & lastRBS,
    size_t & lastCBS,
    size_t & changeRBS,
    size_t & changeCBS,
    size_t & numRowsBlock,
    size_t & numColBlock,
    size_t m,
    size_t n,
    const size_t numthreads ) [inline]

```

**15.1.3.290 pfzero()**

```

void pfzero (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr C,
    size_t BS = 0 )

```

**15.1.3.291 pfrand()**

```

void pfrand (
    const Field & F,
    RandIter & G,
    size_t m,
    size_t n,
    typename Field::Element_ptr C,
    size_t BS = 0 )

```

**15.1.3.292 fdot()** [11/11]

```

Field::Element & fdot (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element & d,
    const ParSeqHelper::Parallel< Cut, Param > par ) [inline]

```

**15.1.3.293 pfgemm() [2/7]**

```
Field::Element * pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block,
StrategyParameter::Threads > > & H )
```

**15.1.3.294 pfgemm() [3/7]**

```
Field::Element * pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr AA,
    const size_t lda,
    const typename Field::ConstElement_ptr BB,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeDAdaptive > > & H )
```

**15.1.3.295 pfgemm() [4/7]**

```
Field::Element * pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
```

```

    const typename Field::ConstElement_ptr AA,
    const size_t lda,
    const typename Field::ConstElement_ptr BB,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::TwoDAdaptive > > & H )

```

### 15.1.3.296 pfgemm() [5/7]

```

Field::Element * pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr AA,
    const size_t lda,
    const typename Field::ConstElement_ptr BB,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element * C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::TwoD > > & H )

```

### 15.1.3.297 pfgemm() [6/7]

```

Field::Element_ptr pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeD > > & H )

```



**15.1.3.298 pfgemm()** [7/7]

```
Field::Element * pfgemm (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::ThreeDInPlace > > & H )
```

**15.1.3.299 fgemv()** [18/19]

```
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive,
StrategyParameter::Threads > > & H )
```

**15.1.3.300 fgemv()** [19/19]

```
Field::Element_ptr fgemv (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::ConstElement_ptr X,
    const size_t incX,
```

```

    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY,
    MMHelper< Field, AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Row,
Cut > > & H )

```

#### 15.1.3.301 parseArguments()

```

void parseArguments (
    int argc,
    char ** argv,
    Argument * args,
    bool printDefaults = true )

```

#### 15.1.3.302 writeCommandString()

```

std::ostream & writeCommandString (
    std::ostream & os,
    Argument * args,
    const char * programName = nullptr )

```

writes the values of all arguments, preceded by the programName

#### 15.1.3.303 WriteMatrix() [1/2]

```

std::ostream & WriteMatrix (
    std::ostream & c,
    const Field & F,
    size_t m,
    size_t n,
    typename Field::ConstElement_ptr A,
    size_t lda,
    FFLAS_FORMAT format,
    bool column_major ) [inline]

```

WriteMatrix: write a matrix to an output stream.

##### Parameters

|                     |  |
|---------------------|--|
| <i>c</i>            | output stream  |
| <i>F</i>            | base field   |
| <i>m</i>            | row dimension  |
| <i>n</i>            | column dimension   |
| <i>A</i>            | matrix   |
| <i>format</i>       | input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)          |
| <i>column_major</i> | whether the matrix is stored in column or row major (row by default) |

**15.1.3.304 preamble()**

```
void preamble (
    std::ifstream & ifs,
    FFLAS_FORMAT & format ) [inline]
```

**15.1.3.305 ReadMatrix() [1/2]**

```
Field::Element_ptr ReadMatrix (
    std::ifstream & ifs,
    Field & F,
    size_t & m,
    size_t & n,
    typename Field::Element_ptr & A,
    FFLAS_FORMAT format = FflasAuto )
```

ReadMatrix: read a matrix from an input stream.

**Parameters**

|     |               |   |
|-----|---------------|---|
|     | <i>ifs</i>    | input stream  |
|     | <i>F</i>      | base field  |
| out | <i>m</i>      | row dimension   |
| out | <i>n</i>      | column dimension  |
| out | <i>A</i>      | output matrix   |
|     | <i>format</i> | input format (FflasAuto, FflasDense, FflasSMS, FflasBinary) |

**15.1.3.306 ReadMatrix() [2/2]**

```
Field::Element_ptr ReadMatrix (
    const std::string & matrix_file,
    Field & F,
    size_t & m,
    size_t & n,
    typename Field::Element_ptr & A,
    FFLAS_FORMAT format = FflasAuto ) [inline]
```

ReadMatrix: read a matrix from a file.

**Parameters**

|                      |                    |   |
|----------------------|--------------------|---|
|                      | <i>matrix_file</i> | filename  |
|                      | <i>F</i>           | base field  |
| out                  | <i>m</i>           | row dimension   |
| out                  | <i>n</i>           | column dimension  |
| out                  | <i>A</i>           | output matrix   |
| Generated by Doxygen | <i>format</i>      | input format (FflasAuto, FflasDense, FflasSMS, FflasBinary) |

**15.1.3.307 WriteMatrix()** [2/2]

```

void WriteMatrix (
    std::string & matrix_file,
    const Field & F,
    int m,
    int n,
    typename Field::ConstElement_ptr A,
    size_t lda,
    FFLAS_FORMAT format = FflasDense,
    bool column_major = false )

```

WriteMatrix: write a matrix to a file.

**Parameters**

|                     |  |
|---------------------|--|
| <i>matrix_file</i>  | file name  |
| <i>F</i>            | base field   |
| <i>m</i>            | row dimension  |
| <i>n</i>            | column dimension   |
| <i>A</i>            | matrix   |
| <i>format</i>       | input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)          |
| <i>column_major</i> | whether the matrix is stored in column or row major (row by default) |

**15.1.3.308 WritePermutation()**

```

std::ostream & WritePermutation (
    std::ostream & c,
    const size_t * P,
    size_t N ) [inline]

```

WritePermutation: write a permutation matrix to an output stream.

**Parameters**

|          |                         |
|----------|-------------------------|
| <i>c</i> | output stream           |
| <i>P</i> | permutation             |
| <i>N</i> | size of the permutation |

**15.1.3.309 alignable()**

```

bool alignable ( ) [inline]

```

**15.1.3.310 alignable< Givaro::Integer \* >()**

```
bool alignable< Givaro::Integer * > ( ) [inline]
```

**15.1.3.311 fflas\_new() [5/7]**

```
Field::Element_ptr fflas_new (
    const Field & F,
    const size_t m,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

**15.1.3.312 fflas\_new() [6/7]**

```
Field::Element_ptr fflas_new (
    const Field & F,
    const size_t m,
    const size_t n,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

**15.1.3.313 fflas\_new() [7/7]**

```
Element * fflas_new (
    const size_t m,
    const Alignment align = Alignment::DEFAULT ) [inline]
```

**15.1.3.314 fflas\_delete() [3/4]**

```
void fflas_delete (
    Element_ptr A ) [inline]
```

**15.1.3.315 fflas\_delete() [4/4]**

```
void fflas_delete (
    Ptr p,
    Args ... args ) [inline]
```

**15.1.3.316 prefetch()**

```
void prefetch (
    const int64_t * ) [inline]
```

**15.1.3.317 getTLBSize()**

```
void getTLBSize (
    int & tlb ) [inline]
```

**15.1.3.318 queryCacheSizes()**

```
void queryCacheSizes (
    int & l1,
    int & l2,
    int & l3 ) [inline]
```

Queries and returns the cache sizes in Bytes of the L1, L2, and L3 data caches respectively

**15.1.3.319 queryL1CacheSize()**

```
int queryL1CacheSize ( ) [inline]
```

**Returns**

the size in Bytes of the L1 data cache

**15.1.3.320 queryTopLevelCacheSize()**

```
int queryTopLevelCacheSize ( ) [inline]
```

**Returns**

the size in Bytes of the L2 or L3 cache if this later is present

**15.1.3.321 getSeed()**

```
uint64_t getSeed ( )
```

## 15.2 FFLAS::BLAS3 Namespace Reference

### Functions

- `template<class Field >`  
`void Bini (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr,`  
`const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A,`  
`const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta,`  
`typename Field::Element_ptr C, const size_t ldc, const size_t kmax, const size_t w, const FFLAS_BASE`  
`base, const size_t rec_level)`
- `template<class Field , class FieldTrait , class Strat , class Param >`  
`Field::Element_ptr WinoPar (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel< Strat, Param > > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t`  
`mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr`  
`A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element`  
`beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, Field↵`  
`Trait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_3_23 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_3_21 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_2_24 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const type-`  
`name Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_2_27 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const type-`  
`name Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field,`  
`MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_LR (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const`  
`size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr`  
`A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta,`  
`typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, Field↵`  
`Trait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const type-`  
`name Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const type-`  
`name Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field,`  
`MMHelperAlgo::Winograd, FieldTrait > &WH)`

- `template<class Field , class FieldTrait >`  
`void WinogradAcc_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd_LR_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd_L_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void Winograd_R_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

## 15.2.1 Function Documentation

### 15.2.1.1 Bini()

```
void Bini (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const size_t kmax,
    const size_t w,
    const FFLAS_BASE base,
    const size_t rec_level ) [inline]
```



### 15.2.1.2 WinoPar()

```
Field::Element_ptr WinoPar (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::WinogradPar, FieldTrait, ParSeqHelper::Parallel<
Strat, Param > > & WH ) [inline]
```

### 15.2.1.3 Winograd()

```
void Winograd (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

### 15.2.1.4 WinogradAcc\_3\_23()

```
void WinogradAcc_3_23 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
```

```

const size_t lda,
typename Field::ConstElement_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.2.1.5 WinogradAcc\_3\_21()

```

void WinogradAcc_3_21 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.2.1.6 WinogradAcc\_2\_24()

```

void WinogradAcc_2_24 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

**15.2.1.7 WinogradAcc\_2\_27()**

```

void WinogradAcc_2_27 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

**15.2.1.8 WinogradAcc\_LR()**

```

void WinogradAcc_LR (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

**15.2.1.9 WinogradAcc\_R\_S()**

```

void WinogradAcc_R_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,

```

```

typename Field::Element_ptr B,
const size_t ldb,
const typename Field::Element beta,
typename Field::Element_ptr C,
const size_t ldc,
const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.2.1.10 WinogradAcc\_L\_S()

```

void WinogradAcc_L_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

#### 15.2.1.11 Winograd\_LR\_S()

```

void Winograd_LR_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]

```

### 15.2.1.12 Winograd\_L\_S()

```
void Winograd_L_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

### 15.2.1.13 Winograd\_R\_S()

```
void Winograd_R_S (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    const typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH ) [inline]
```

## 15.3 FFLAS::csr\_hyb\_details Namespace Reference

### Data Structures

- struct [Coo](#)
- struct [Info](#)

## 15.4 FFLAS::CuttingStrategy Namespace Reference

### Data Structures

- struct [Block](#)
- struct [Column](#)
- struct [Recursive](#)
- struct [Row](#)
- struct [Single](#)

## Typedefs

- typedef [Row](#) [RNSModulus](#)

### 15.4.1 Typedef Documentation

#### 15.4.1.1 RNSModulus

```
typedef Row RNSModulus
```

## 15.5 FFLAS::details Namespace Reference

### Functions

- template<class [Field](#), bool ADD>  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< typename [Field::Element](#) >::value, void >::type [fadd](#)  
(const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [FieldCategories::ModularTag](#))
- template<class [Field](#), bool ADD>  
std::enable\_if<![FFLAS::support\\_simd\\_add](#)< typename [Field::Element](#) >::value, void >::type [fadd](#)  
(const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [FieldCategories::ModularTag](#))
- template<class [Field](#), bool ADD>  
void [fadd](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, type-  
name [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [FieldCategories::GenericTag](#))
- template<class [Field](#), bool ADD>  
std::enable\_if<![FFLAS::support\\_simd\\_add](#)< typename [Field::Element](#) >::value, void >::type [fadd](#)  
(const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [FieldCategories::UnparametricTag](#))
- template<class [Field](#), bool ADD>  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< typename [Field::Element](#) >::value, void >::type [fadd](#)  
(const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [FieldCategories::UnparametricTag](#))
- template<class [Field](#) >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [freduce](#) (const [Field](#) &F, const size\_t m, typename [Field::Element\\_ptr](#) A, const size\_t incX, [FieldCategories::ModularTag](#))
- template<class [Field](#) >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [freduce](#)  
(const [Field](#) &F, const size\_t m, typename [Field::ConstElement\\_ptr](#) B, const size\_t incY, typename [Field::Element\\_ptr](#) A, const size\_t incX, [FieldCategories::ModularTag](#))
- template<class [Field](#), class FC >  
void [freduce](#) (const [Field](#) &F, const size\_t m, typename [Field::Element\\_ptr](#) A, const size\_t incX, FC)
- template<class [Field](#), class FC >  
void [freduce](#) (const [Field](#) &F, const size\_t m, typename [Field::ConstElement\\_ptr](#) B, const size\_t incY, type-  
name [Field::Element\\_ptr](#) A, const size\_t incX, FC)

- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscalin (const`  
`Field &F, const size_t N, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX,`  
`FieldCategories::ModularTag)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscal (const`  
`Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t`  
`incX, typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)`
- `template<class Field , class FC >`  
`void fscalin (const Field &F, const size_t n, const typename Field::Element a, typename Field::Element_ptr X,`  
`const size_t incX, FC)`
- `template<class Field , class FC >`  
`void fscal (const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr`  
`X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)`
- `template<enum number_kind K>`  
`void igebb44 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t`  
`*blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb24 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t`  
`*blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb14 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t`  
`*blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb41 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t`  
`*blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb21 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t`  
`*blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb11 (size_t i, size_t j, size_t depth, size_t pdeth, const int64_t alpha, const int64_t *blA, const int64_t`  
`*blB, int64_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebp (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *blockA, size_t lda, const`  
`int64_t *blockB, size_t ldb, int64_t *C, size_t ldc)`
- `template<size_t k, bool transpose>`  
`void pack_lhs (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)`
- `template<size_t k, bool transpose>`  
`void pack_rhs (int64_t *XX, const int64_t *X, size_t ldx, size_t rows, size_t cols)`
- `void gebp (size_t rows, size_t cols, size_t depth, int64_t *C, size_t ldc, const int64_t *blockA, size_t lda, const`  
`int64_t *BlockB, size_t ldb, int64_t *BlockW)`
- `void BlockingFactor (size_t &m, size_t &n, size_t &k)`

## 15.5.1 Function Documentation

### 15.5.1.1 fadd() [1/5]

```
std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
```

```

typename Field::ConstElement_ptr B,
const size_t incb,
typename Field::Element_ptr C,
const size_t incc,
FieldCategories::ModularTag )

```

#### 15.5.1.2 fadd() [2/5]

```

std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::ModularTag )

```

#### 15.5.1.3 fadd() [3/5]

```

void fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::GenericTag )

```

#### 15.5.1.4 fadd() [4/5]

```

std::enable_if<!FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::UnparametricTag ) [inline]

```



**15.5.1.5 fadd()** [5/5]

```
std::enable_if< FFLAS::support_simd_add< typenameField::Element >::value, void >::type fadd (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t inca,
    typename Field::ConstElement_ptr B,
    const size_t incb,
    typename Field::Element_ptr C,
    const size_t incc,
    FieldCategories::UnparametricTag ) [inline]
```

**15.5.1.6 freduce()** [1/4]

```
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce
(
    const Field & F,
    const size_t m,
    typename Field::Element_ptr A,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]
```

**15.5.1.7 freduce()** [2/4]

```
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce
(
    const Field & F,
    const size_t m,
    typename Field::ConstElement_ptr B,
    const size_t incY,
    typename Field::Element_ptr A,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]
```

**15.5.1.8 freduce()** [3/4]

```
void freduce (
    const Field & F,
    const size_t m,
    typename Field::Element_ptr A,
    const size_t incX,
    FC ) [inline]
```

**15.5.1.9 freduce()** [4/4]

```

void freduce (
    const Field & F,
    const size_t m,
    typename Field::ConstElement_ptr B,
    const size_t incY,
    typename Field::Element_ptr A,
    const size_t incX,
    FC ) [inline]

```

**15.5.1.10 fscaln()** [1/2]

```

std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscaln
(
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::Element_ptr X,
    const size_t incX,
    FieldCategories::ModularTag ) [inline]

```

**15.5.1.11 fscale()** [1/2]

```

std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscale
(
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    FieldCategories::ModularTag ) [inline]

```

**15.5.1.12 fscln()** [2/2]

```

void fscln (
    const Field & F,
    const size_t n,
    const typename Field::Element a,
    typename Field::Element_ptr X,
    const size_t incX,
    FC ) [inline]

```

#### 15.5.1.13 fscal() [2/2]

```
void fscal (
    const Field & F,
    const size_t N,
    const typename Field::Element a,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    typename Field::Element_ptr Y,
    const size_t incY,
    FC ) [inline]
```

#### 15.5.1.14 igebb44()

```
void igebb44 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

#### 15.5.1.15 igebb24()

```
void igebb24 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

#### 15.5.1.16 igebb14()

```
void igebb14 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

#### 15.5.1.17 igebb41()

```
void igebb41 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

bug ,B\_0 dans VEC\_MADD\_32 ?

bug ,B\_0 dans VEC\_MADD\_32 ?

#### 15.5.1.18 igebb21()

```
void igebb21 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

#### 15.5.1.19 igebb11()

```
void igebb11 (
    size_t i,
    size_t j,
    size_t depth,
    size_t pdeth,
    const int64_t alpha,
    const int64_t * blA,
    const int64_t * blB,
    int64_t * C,
    size_t ldc ) [inline]
```

### 15.5.1.20 igebp()

```
void igebp (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * blockA,
    size_t lda,
    const int64_t * blockB,
    size_t ldb,
    int64_t * C,
    size_t ldc )
```

### 15.5.1.21 pack\_lhs()

```
void pack_lhs (
    int64_t * XX,
    const int64_t * X,
    size_t ldx,
    size_t rows,
    size_t cols )
```

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign

### 15.5.1.22 pack\_rhs()

```
void pack_rhs (
    int64_t * XX,
    const int64_t * X,
    size_t ldx,
    size_t rows,
    size_t cols )
```

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign

**Bug** this is fassign

### 15.5.1.23 `gebp()`

```
void gebp (
    size_t rows,
    size_t cols,
    size_t depth,
    int64_t * C,
    size_t ldc,
    const int64_t * blockA,
    size_t lda,
    const int64_t * BlockB,
    size_t ldb,
    int64_t * BlockW )
```

### 15.5.1.24 `BlockingFactor()`

```
void BlockingFactor (
    size_t & m,
    size_t & n,
    size_t & k ) [inline]
```

## 15.6 `FFLAS::details_spmv` Namespace Reference

### Data Structures

- struct [Coo](#)

## 15.7 `FFLAS::ElementCategories` Namespace Reference

### Data Structures

- struct [ArbitraryPrecIntTag](#)  
*Arbitrary precision integers: GMP.*
- struct [FixedPrecIntTag](#)  
*Fixed precision integers above machine precision: Givaro::reclnt.*
- struct [GenericTag](#)  
*default is generic*
- struct [MachineFloatTag](#)  
*float or double*
- struct [MachineIntTag](#)  
*short, int, long, long long, and unsigned variants*
- struct [RNSElementTag](#)  
*Representation in a Residue Number System.*

## 15.8 FFLAS::FieldCategories Namespace Reference

Traits and categories will need to be placed in a proper file later.

### Data Structures

- struct [GenericTag](#)  
*generic ring.*
- struct [ModularTag](#)  
*This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`*
- struct [UnparametricTag](#)  
*If the field uses a representation with infix operators.*

#### 15.8.1 Detailed Description

Traits and categories will need to be placed in a proper file later.

## 15.9 FFLAS::MMHelperAlgo Namespace Reference

### Data Structures

- struct [Auto](#)
- struct [Bini](#)
- struct [Classic](#)
- struct [Winograd](#)
- struct [WinogradPar](#)

## 15.10 FFLAS::ModeCategories Namespace Reference

Specifies the mode of action for an algorithm w.r.t.

### Data Structures

- struct [ConvertTo](#)  
*Force conversion to appropriate element type of `ElementCategory T`.*
- struct [DefaultBoundedTag](#)  
*Use standard field operations, but keeps track of bounds on input and output.*
- struct [DefaultTag](#)  
*No specific mode of action: use standard field operations.*
- struct [DelayedTag](#)  
*Performs field operations with delayed mod reductions. Ensures result is reduced.*
- struct [LazyTag](#)  
*Performs field operations with delayed mod only when necessary. Result may not be reduced.*

### 15.10.1 Detailed Description

Specifies the mode of action for an algorithm w.r.t.

its field

## 15.11 FFLAS::ParSeqHelper Namespace Reference

[ParSeqHelper](#) for both fgemm and ftrsm.

### Data Structures

- struct [Compose](#)
- struct [Parallel](#)
- struct [Sequential](#)

### 15.11.1 Detailed Description

[ParSeqHelper](#) for both fgemm and ftrsm.

[ParSeqHelper](#) for both fgemm and ftrsm

## 15.12 FFLAS::Protected Namespace Reference

### Data Structures

- class [AreEqual](#)
- class [AreEqual< X, X >](#)
- class [ftrmmLeftLowerNoTransNonUnit](#)
- class [ftrmmLeftLowerNoTransUnit](#)
- class [ftrmmLeftLowerTransNonUnit](#)
- class [ftrmmLeftLowerTransUnit](#)
- class [ftrmmLeftUpperNoTransNonUnit](#)
- class [ftrmmLeftUpperNoTransUnit](#)
- class [ftrmmLeftUpperTransNonUnit](#)
- class [ftrmmLeftUpperTransUnit](#)
- class [ftrmmRightLowerNoTransNonUnit](#)
- class [ftrmmRightLowerNoTransUnit](#)
- class [ftrmmRightLowerTransNonUnit](#)
- class [ftrmmRightLowerTransUnit](#)
- class [ftrmmRightUpperNoTransNonUnit](#)
- class [ftrmmRightUpperNoTransUnit](#)
- class [ftrmmRightUpperTransNonUnit](#)
- class [ftrmmRightUpperTransUnit](#)
- class [ftrsmLeftLowerNoTransNonUnit](#)
- class [ftrsmLeftLowerNoTransUnit](#)
- class [ftrsmLeftLowerTransNonUnit](#)



- class [ftrsmLeftLowerTransUnit](#)
- class [ftrsmLeftUpperNoTransNonUnit](#)

*Computes the maximal size for delaying the modular reduction in a triangular system resolution.*

- class [ftrsmLeftUpperNoTransUnit](#)
- class [ftrsmLeftUpperTransNonUnit](#)
- class [ftrsmLeftUpperTransUnit](#)
- class [ftrsmRightLowerNoTransNonUnit](#)
- class [ftrsmRightLowerNoTransUnit](#)
- class [ftrsmRightLowerTransNonUnit](#)
- class [ftrsmRightLowerTransUnit](#)
- class [ftrsmRightUpperNoTransNonUnit](#)
- class [ftrsmRightUpperNoTransUnit](#)
- class [ftrsmRightUpperTransNonUnit](#)
- class [ftrsmRightUpperTransUnit](#)

## Functions

- template<class [Field](#) >  
double [computeFactorClassic](#) (const [Field](#) &F)
- template<> double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< double > &F)
- template<> double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< float > &F)
- template<class [Field](#) >  
size\_t [DotProdBoundClassic](#) (const [Field](#) &F, const typename [Field::Element](#) &beta)
- template<class [Field](#) >  
size\_t [TRSMBound](#) (const [Field](#) &)  
*TRSMBound.*
- template<class [Element](#) >  
size\_t [TRSMBound](#) (const [Givaro::Modular](#)< [Element](#) > &F)  
*Specialization for positive modular representation over float.*
- template<class [Element](#) >  
size\_t [TRSMBound](#) (const [Givaro::ModularBalanced](#)< [Element](#) > &F)  
*Specialization for balanced modular representation over double.*
- template<class [NewField](#) , class [Field](#) , class [FieldMode](#) >  
[Field::Element\\_ptr](#) [fgemm\\_convert](#) (const [Field](#) &F, const [FFLAS\\_TRANSPOSE](#) ta, const [FFLAS\\_TRANSPOSE](#) tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo::Winograd](#), [FieldMode](#) > &H)
- template<class [Field](#) , class [Element](#) , class [AlgoT](#) , class [ParSeqTrait](#) >  
bool [NeedPreAddReduction](#) ([Element](#) &Outmin, [Element](#) &Outmax, [Element](#) &Op1min, [Element](#) &Op1max, [Element](#) &Op2min, [Element](#) &Op2max, [MMHelper](#)< [Field](#), [AlgoT](#), [ModeCategories::LazyTag](#), [ParSeqTrait](#) > &WH)
- template<class [Field](#) , class [Element](#) , class [AlgoT](#) , class [ModeT](#) , class [ParSeqTrait](#) >  
bool [NeedPreAddReduction](#) ([Element](#) &Outmin, [Element](#) &Outmax, [Element](#) &Op1min, [Element](#) &Op1max, [Element](#) &Op2min, [Element](#) &Op2max, [MMHelper](#)< [Field](#), [AlgoT](#), [ModeT](#), [ParSeqTrait](#) > &WH)
- template<class [Field](#) , class [Element](#) , class [AlgoT](#) , class [ParSeqTrait](#) >  
bool [NeedPreSubReduction](#) ([Element](#) &Outmin, [Element](#) &Outmax, [Element](#) &Op1min, [Element](#) &Op1max, [Element](#) &Op2min, [Element](#) &Op2max, [MMHelper](#)< [Field](#), [AlgoT](#), [ModeCategories::LazyTag](#), [ParSeqTrait](#) > &WH)
- template<class [Field](#) , class [Element](#) , class [AlgoT](#) , class [ModeT](#) , class [ParSeqTrait](#) >  
bool [NeedPreSubReduction](#) ([Element](#) &Outmin, [Element](#) &Outmax, [Element](#) &Op1min, [Element](#) &Op1max, [Element](#) &Op2min, [Element](#) &Op2max, [MMHelper](#)< [Field](#), [AlgoT](#), [ModeT](#), [ParSeqTrait](#) > &WH)

- `template<class Field, class Element, class AlgoT, class ParSeqTrait >`  
`bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)`
- `template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait >`  
`bool NeedDoublePreAddReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)`
- `template<class Field, class AlgoT, class ParSeqTrait >`  
`void ScalAndReduce (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)`
- `template<class Field, class AlgoT, class ParSeqTrait >`  
`void ScalAndReduce (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)`
- `template<class Field >`  
`Field::Element_ptr fsquareCommon (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`
- `template<class Field >`  
`int WinogradThreshold (const Field &F)`  
*Computes the number of recursive levels to perform.*
- `template<> int WinogradThreshold (const Givaro::Modular< float > &F)`
- `template<> int WinogradThreshold (const Givaro::ModularBalanced< double > &F)`
- `template<> int WinogradThreshold (const Givaro::ModularBalanced< float > &F)`
- `template<class Field >`  
`int WinogradSteps (const Field &F, const size_t &m)`  
*Computes the number of recursive levels to perform.*
- `template<class Field, class FieldMode >`  
`void DynamicPeeling (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)`
- `template<class Field, class FieldMode >`  
`void DynamicPeeling2 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)`
- `template<class Field, class FieldMode >`  
`void WinogradCalc (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)`
- `template<typename FloatElement, class Field >`  
`Field::Element_ptr fgemv_convert (const Field &F, const FFLAS_TRANSPOSE ta, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr X, const size_t incX, const typename Field::Element beta, typename Field::Element_ptr Y, const size_t incY)`

- `template<class FloatElement , class Field >`  
`void fger_convert (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha,`  
`typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t`  
`incy, typename Field::Element_ptr A, const size_t lda)`
- `template<class DFE >`  
`size_t min_types (const DFE &k)`
- `template<> size_t min_types (const Reclnt::rint< 6 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 7 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 8 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 9 > &k)`
- `template<> size_t min_types (const Reclnt::rint< 10 > &k)`
- `template<> size_t min_types (const Givaro::Integer &k)`
- `template<class T >`  
`bool unfit (T x)`
- `template<> bool unfit (int64_t x)`
- `template<size_t K>`  
`bool unfit (Reclnt::rint< K > x)`
- `template<> bool unfit (Reclnt::rint< 6 > x)`
- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB>`  
`void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda,`  
`const int64_t *B, size_t ldb, int64_t *C, size_t ldc)`
- `template<enum FFLAS_TRANSPOSE tA, enum FFLAS_TRANSPOSE tB, enum number_kind alpha_kind>`  
`void igemm_colmajor (size_t rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda,`  
`const int64_t *B, size_t ldb, int64_t *C, size_t ldc)`
- `void igemm (const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_TRANSPOSE TransB, size_t`  
`rows, size_t cols, size_t depth, const int64_t alpha, const int64_t *A, size_t lda, const int64_t *B, size_t ldb,`  
`const int64_t beta, int64_t *C, size_t ldc)`
- `template<class Field >`  
`void MatF2MatD_Triangular (const Field &F, Givaro::DoubleDomain::Element_ptr S, const size_t lds, type-`  
`name Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)`
- `template<class Field >`  
`void MatF2MatFI_Triangular (const Field &F, Givaro::FloatDomain::Element_ptr S, const size_t lds, typename`  
`Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)`

## 15.12.1 Function Documentation

### 15.12.1.1 computeFactorClassic() [1/3]

```
double computeFactorClassic (
    const Field & F ) [inline]
```

### 15.12.1.2 computeFactorClassic() [2/3]

```
double computeFactorClassic (
    const Givaro::ModularBalanced< double > & F ) [inline]
```

**15.12.1.3 computeFactorClassic() [3/3]**

```
double computeFactorClassic (
    const Givaro::ModularBalanced< float > & F ) [inline]
```

**15.12.1.4 DotProdBoundClassic()**

```
size_t DotProdBoundClassic (
    const Field & F,
    const typename Field::Element & beta ) [inline]
```

**15.12.1.5 TRSMBound() [1/3]**

```
size_t TRSMBound (
    const Field & ) [inline]
```

TRSMBound.

computes the maximal size for delaying the modular reduction in a triangular system resolution

This is the default version over an arbitrary field. It is currently never used (the recursive algorithm is run until  $n=1$  in this case)

**Parameters**

|     |                                      |
|-----|--------------------------------------|
| $F$ | Finite Field/Ring of the computation |
|-----|--------------------------------------|

**15.12.1.6 TRSMBound() [2/3]**

```
size_t TRSMBound (
    const Givaro::Modular< Element > & F ) [inline]
```

Specialization for positive modular representation over float.

Computes  $n_{\max}$  s.t.  $(p-1)/2 * (p^{\{n_{\max}-1\}} + (p-2)^{\{n_{\max}-1\}}) < 2^{24}$  @pbi See [Dumas Giorgi Pernet 06, arXiv:cs/0601133]

**15.12.1.7 TRSMBound() [3/3]**

```
size_t TRSMBound (
    const Givaro::ModularBalanced< Element > & F ) [inline]
```

Specialization for balanced modular representation over double.

Computes  $n_{\max}$  s.t.  $(p-1)/2 * (((p+1)/2)^{\{n_{\max}-1\}}) < 2^{53}$

**Bibliography** • Dumas Giorgi Pernet 06, arXiv:cs/0601133

**15.12.1.8 fgemm\_convert()**

```
Field::Element_ptr fgemm_convert (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H ) [inline]
```

**15.12.1.9 NeedPreAddReduction() [1/2]**

```
bool NeedPreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]
```

**15.12.1.10 NeedPreAddReduction() [2/2]**

```
bool NeedPreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]
```

**15.12.1.11 NeedPreSubReduction() [1/2]**

```
bool NeedPreSubReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]
```

**15.12.1.12 NeedPreSubReduction()** [2/2]

```

bool NeedPreSubReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]

```

**15.12.1.13 NeedDoublePreAddReduction()** [1/2]

```

bool NeedDoublePreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    Element beta,
    MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH ) [inline]

```

**15.12.1.14 NeedDoublePreAddReduction()** [2/2]

```

bool NeedDoublePreAddReduction (
    Element & Outmin,
    Element & Outmax,
    Element & Op1min,
    Element & Op1max,
    Element & Op2min,
    Element & Op2max,
    Element beta,
    MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH ) [inline]

```

**15.12.1.15 ScalAndReduce()** [1/2]

```

void ScalAndReduce (
    const Field & F,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr X,
    const size_t incX,
    const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H ) [inline]

```

**15.12.1.16 ScalAndReduce()** [2/2]

```

void ScalAndReduce (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::Element_ptr A,
    const size_t lda,
    const MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & H ) [inline]

```

**15.12.1.17 fsquareCommon()**

```

Field::Element_ptr fsquareCommon (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t n,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc ) [inline]

```

**15.12.1.18 WinogradThreshold()** [1/4]

```

int WinogradThreshold (
    const Field & F ) [inline]

```

Computes the number of recursive levels to perform.

**Parameters**

|          |   |
|----------|---|
| <i>m</i> | the common dimension in the product AxB |
|----------|---|

**15.12.1.19 WinogradThreshold()** [2/4]

```

int WinogradThreshold (
    const Givaro::Modular< float > & F ) [inline]

```

**15.12.1.20 WinogradThreshold()** [3/4]

```
int WinogradThreshold (
    const Givaro::ModularBalanced< double > & F ) [inline]
```

**15.12.1.21 WinogradThreshold()** [4/4]

```
int WinogradThreshold (
    const Givaro::ModularBalanced< float > & F ) [inline]
```

**15.12.1.22 WinogradSteps()**

```
int WinogradSteps (
    const Field & F,
    const size_t & m ) [inline]
```

Computes the number of recursive levels to perform.

**Parameters**

|          |   |
|----------|---|
| <i>m</i> | the common dimension in the product AxB |
|----------|---|

**15.12.1.23 DynamicPeeling()**

```
void DynamicPeeling (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↔
    Field::Element Cmin,
```



```

        const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↵
Field::Element Cmax ) [inline]

```

#### 15.12.1.24 DynamicPeeling2()

```

void DynamicPeeling2 (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↵
Field::Element Cmin,
    const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↵
Field::Element Cmax ) [inline]

```

#### 15.12.1.25 WinogradCalc()

```

void WinogradCalc (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const FFLAS_TRANSPOSE tb,
    const size_t mr,
    const size_t nr,
    const size_t kr,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    const typename Field::Element beta,
    typename Field::Element_ptr C,
    const size_t ldc,
    MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > & H ) [inline]

```

**15.12.1.26 fgemv\_convert()**

```
Field::Element_ptr fgemv_convert (
    const Field & F,
    const FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr X,
    const size_t incX,
    const typename Field::Element beta,
    typename Field::Element_ptr Y,
    const size_t incY ) [inline]
```

**15.12.1.27 fger\_convert()**

```
void fger_convert (
    const Field & F,
    const size_t M,
    const size_t N,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr x,
    const size_t incx,
    typename Field::ConstElement_ptr y,
    const size_t incy,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]
```

**15.12.1.28 min\_types() [1/7]**

```
size_t min_types (
    const DFE & k ) [inline]
```

**15.12.1.29 min\_types() [2/7]**

```
size_t min_types (
    const RecInt::rint< 6 > & k ) [inline]
```

**15.12.1.30 min\_types() [3/7]**

```
size_t min_types (
    const RecInt::rint< 7 > & k ) [inline]
```

**15.12.1.31 min\_types() [4/7]**

```
size_t min_types (
    const RecInt::rint< 8 > & k )    [inline]
```

**15.12.1.32 min\_types() [5/7]**

```
size_t min_types (
    const RecInt::rint< 9 > & k )    [inline]
```

**15.12.1.33 min\_types() [6/7]**

```
size_t min_types (
    const RecInt::rint< 10 > & k )    [inline]
```

**15.12.1.34 min\_types() [7/7]**

```
size_t min_types (
    const Givaro::Integer & k )    [inline]
```

**15.12.1.35 unfit() [1/4]**

```
bool unfit (
    T x )    [inline]
```

**15.12.1.36 unfit() [2/4]**

```
bool unfit (
    int64_t x )    [inline]
```

**15.12.1.37 unfit() [3/4]**

```
bool unfit (
    RecInt::rint< K > x )    [inline]
```

**15.12.1.38 unfit()** [4/4]

```
bool unfit (
    RecInt::rint< 6 > x ) [inline]
```

**15.12.1.39 igemm\_colmajor()** [1/2]

```
void igemm_colmajor (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    int64_t * C,
    size_t ldc )
```

**15.12.1.40 igemm\_colmajor()** [2/2]

```
void igemm_colmajor (
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    int64_t * C,
    size_t ldc )
```

**15.12.1.41 igemm()**

```
void igemm (
    const enum FFLAS_TRANSPOSE TransA,
    const enum FFLAS_TRANSPOSE TransB,
    size_t rows,
    size_t cols,
    size_t depth,
    const int64_t alpha,
    const int64_t * A,
    size_t lda,
    const int64_t * B,
    size_t ldb,
    const int64_t beta,
    int64_t * C,
    size_t ldc ) [inline]
```

**Todo** use primitive (no [Field\(\)](#)) and specialise for int64.

**Todo** use primitive (no [Field\(\)](#)) and specialise for int64.

#### 15.12.1.42 MatF2MatD\_Triangular()

```
void MatF2MatD_Triangular (
    const Field & F,
    Givaro::DoubleDomain::Element_ptr S,
    const size_t lds,
    typename Field::ConstElement_ptr const E,
    const size_t lde,
    const size_t m,
    const size_t n )
```

#### 15.12.1.43 MatF2MatF1\_Triangular()

```
void MatF2MatF1_Triangular (
    const Field & F,
    Givaro::FloatDomain::Element_ptr S,
    const size_t lds,
    typename Field::ConstElement_ptr const E,
    const size_t lde,
    const size_t m,
    const size_t n )
```

**Todo** do finit(...,FFLAS\_TRANS,FFLAS\_DIAG)  
do fconvert(...,FFLAS\_TRANS,FFLAS\_DIAG)

## 15.13 FFLAS::sell\_details Namespace Reference

### Data Structures

- struct [Coo](#)
- struct [Info](#)

## 15.14 FFLAS::sparse\_details Namespace Reference

### Functions

- `template<class Field >`  
`void init_y (const Field &F, const size_t m, const typename Field::Element b, typename Field::Element_ptr y)`
- `template<class Field >`  
`void init_y (const Field &F, const size_t m, const size_t n, const typename Field::Element b, typename Field::Element_ptr y, const int ldy)`
- `template<class Field , class SM , class FC , class MZO >`  
`std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type fspmv_dispatch (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FC fc, MZO mzo)`
- `template<class Field , class SM , class FC , class MZO >`  
`std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmv_dispatch (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FC fc, MZO mzo)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value)||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`

- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value || std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< !(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value || std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value) >::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloat >::value || std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< !support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM`

- &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, [FieldCategories::UnparametricTag](#), [NotZOSparseMatrix](#))
- `template<class Field , class SM >`  
`std::enable_if< support\_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if< !support\_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if< support\_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if< !support\_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
  - `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement\_ptr x, int ldx, typename Field::Element\_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::GenericTag, std::false_type)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, std::false_type)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::ModularTag, std::false_type)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::GenericTag, std::true_type)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, std::true_type)`
  - `template<class Field , class SM >`  
`void pfspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::ModularTag, std::true_type)`
  - `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support\_simd< typenameField::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support\_simd< typenameField::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::ModularTag, NotZOSparseMatrix)`
  - `template<class Field , class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support\_simd< typenameField::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement\_ptr x, typename Field::Element\_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`

### 15.14.1 Function Documentation



**15.14.1.1 init\_y() [1/2]**

```
void init_y (
    const Field & F,
    const size_t m,
    const typename Field::Element b,
    typename Field::Element_ptr y ) [inline]
```

**15.14.1.2 init\_y() [2/2]**

```
void init_y (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element b,
    typename Field::Element_ptr y,
    const int ldy ) [inline]
```

**15.14.1.3 fspmv\_dispatch() [1/2]**

```
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt>::value)>::type fspmv_dispatch (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FC fC,
    MZO mzo ) [inline]
```

**15.14.1.4 fspmv\_dispatch() [2/2]**

```
std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt>::value >::type fspmv_dispatch (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FC fC,
    MZO mzo ) [inline]
```

**15.14.1.5 fspmv()** [1/12]

```
void fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.6 fspmv()** [2/12]

```
std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.7 fspmv()** [3/12]

```
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.8 fspmv()** [4/12]

```
std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.9 fspmv()** [5/12]

```
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.10 fspmv()** [6/12]

```
void fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.11 fspmv()** [7/12]

```
std::enable_if<!isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.12 fspmv()** [8/12]

```
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.13 fspmv()** [9/12]

```

void fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::true_type ) [inline]

```

**15.14.1.14 fspmm\_dispatch()** [1/2]

```

std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Mac
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value)>::type fspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]

```

**15.14.1.15 fspmm\_dispatch()** [2/2]

```

std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Mac
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value >::type fspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]

```

**15.14.1.16 fspmm() [1/9]**

```

void fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]

```

**15.14.1.17 fspmm() [2/9]**

```

std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```

**15.14.1.18 fspmm() [3/9]**

```

std::enable_if< !support_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```

**15.14.1.19 fspmm() [4/9]**

```

std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]

```

**15.14.1.20 fspmm()** [5/9]

```
std::enable_if<!support\_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.21 fspmm()** [6/9]

```
void fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.22 fspmm()** [7/9]

```
std::enable_if< support\_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.23 fspmm()** [8/9]

```
std::enable_if<!support\_simd< typenameField::Element >::value >::type fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.24 fspmm()** [9/9]

```

void fspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    ZOSparseMatrix ) [inline]

```

**15.14.1.25 pfspmm\_dispatch()** [1/2]

```

std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Mac
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value)>::type pfspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]

```

**15.14.1.26 pfspmm\_dispatch()** [2/2]

```

std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Mac
>::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineInt
>::value >::type pfspmm_dispatch (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FCat ,
    MZO ) [inline]

```

**15.14.1.27 pfspmm()** [1/9]

```

void pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    NotZOSparseMatrix ) [inline]

```

**15.14.1.28 pfspmm()** [2/9]

```

std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```

**15.14.1.29 pfspmm()** [3/9]

```

std::enable_if< !support_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```

**15.14.1.30 pfspmm()** [4/9]

```

std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]

```



**15.14.1.31 pfspmm()** [5/9]

```
std::enable_if<!support\_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

**15.14.1.32 pfspmm()** [6/9]

```
void pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.33 pfspmm()** [7/9]

```
std::enable_if< support\_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.34 pfspmm()** [8/9]

```
std::enable_if<!support\_simd< typenameField::Element >::value >::type pfspmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

**15.14.1.35 pfsppmm()** [9/9]

```

void pfsppmm (
    const Field & F,
    const SM & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::ModularTag ,
    ZOSparseMatrix ) [inline]

```

**15.14.1.36 pfsppmv()** [1/6]

```

void pfsppmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    std::false_type ) [inline]

```

**15.14.1.37 pfsppmv()** [2/6]

```

void pfsppmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    std::false_type ) [inline]

```

**15.14.1.38 pfsppmv()** [3/6]

```

void pfsppmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::false_type ) [inline]

```

**15.14.1.39 pfspmv()** [4/6]

```

void pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ,
    std::true_type ) [inline]

```

**15.14.1.40 pfspmv()** [5/6]

```

void pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    std::true_type ) [inline]

```

**15.14.1.41 pfspmv()** [6/6]

```

void pfspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    std::true_type ) [inline]

```

**15.14.1.42 fspmv()** [10/12]

```

std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField↵
::Element >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    NotZOSparseMatrix ) [inline]

```

#### 15.14.1.43 fspmv() [11/12]

```
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField↵
::Element >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::ModularTag ,
    NotZOSparseMatrix ) [inline]
```

#### 15.14.1.44 fspmv() [12/12]

```
std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField↵
::Element >::value >::type fspmv (
    const Field & F,
    const SM & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ,
    ZOSparseMatrix ) [inline]
```

## 15.15 FFLAS::sparse\_details\_impl Namespace Reference

### Functions

- template<class Field >  
void fspmm (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, FieldCategories::GenericTag)
- template<class Field >  
void fspmm (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, FieldCategories::UnparametricTag)
- template<class Field >  
void fspmm (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, const int64\_t kmax)
- template<class Field >  
void fspmm\_simd\_aligned (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_t block↵  
Size, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, const int64\_t kmax)
- template<class Field >  
void fspmm\_simd\_unaligned (const Field &F, const Sparse< Field, SparseMatrix\_t::COO > &A, size\_↵  
t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, const int64\_t kmax)
- template<class Field >  
void fspmm\_one (const Field &F, const Sparse< Field, SparseMatrix\_t::COO\_ZO > &A, size\_↵  
t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, FieldCategories::GenericTag)
- template<class Field >  
void fspmm\_mone (const Field &F, const Sparse< Field, SparseMatrix\_t::COO\_ZO > &A, size\_↵  
t blockSize, typename Field::ConstElement\_ptr x\_, int ldx, typename Field::Element\_ptr y\_, int ldy, FieldCategories::GenericTag)

- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

- `template<class Field >`  
`void pfspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_↵`  
`t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_task (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const index_t iStart, const index_t iStop,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename`  
`Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, index_t block_↵`  
`Size, typename Field::ConstElement_ptr x_, index_t ldx, typename Field::Element_ptr y_, index_t ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_↵`  
`_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_↵`  
`_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename`  
`Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_↵`  
`t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_↵`  
`t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,`  
`FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A,`

- ```
size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,
FieldCategories::UnparametricTag)
```
- template<class Field >

```
void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A,
size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,
FieldCategories::UnparametricTag)
```
  - template<class Field >

```
void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A,
size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,
FieldCategories::UnparametricTag)
```
  - template<class Field >

```
void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A,
size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy,
FieldCategories::UnparametricTag)
```
  - template<class Field >

```
void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr
x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
```
  - template<class Field >

```
void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr
x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
```
  - template<class Field >

```
void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr
x_, typename Field::Element_ptr y_, const int64_t kmax)
```
  - template<class Field >

```
void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename
Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
```
  - template<class Field >

```
void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename
Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
```
  - template<class Field >

```
void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename
Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
```
  - template<class Field >

```
void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename
Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)
```
  - template<class Field >

```
void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-
name Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)
```
  - template<class Field >

```
void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-
name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)
```
  - template<class Field >

```
void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-
name Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)
```
  - template<class Field >

```
void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_↵
t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,
FieldCategories::UnparametricTag)
```
  - template<class Field >

```
void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-
name Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)
```
  - template<class Field >

```
void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, type-
name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)
```
  - template<class Field >

```
void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename
Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)
```

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, const int64_t kmax)`
- `template<class Field >`  
`void pfspm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)`
- `template<class Field, class Func >`  
`void pfspm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, Func &&func)`
- `template<class Field, class Func >`  
`void pfspm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, Func &&func)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`



- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`



- [illegible]

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 15.15.1 Function Documentation

**15.15.1.1 fspmm()** [1/15]

```
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.2 fspmm()** [2/15]

```
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.3 fspmm()** [3/15]

```
void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.15.1.4 fspmm\_simd\_aligned()** [1/2]

```
void fspmm_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]
```

**15.15.1.5 fspmm\_simd\_unaligned()** [1/2]

```

void fspmm_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]

```

**15.15.1.6 fspmm\_one()** [1/4]

```

void fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.7 fspmm\_mone()** [1/4]

```

void fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.8 fspmm\_one\_simd\_aligned()** [1/3]

```

void fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.9 fspmm\_one\_simd\_unaligned()** [1/3]

```

void fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.10 fspmm\_mone\_simd\_aligned()** [1/3]

```

void fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.11 fspmm\_mone\_simd\_unaligned()** [1/3]

```

void fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.12 fspmv()** [1/21]

```

void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.13 fspmv()** [2/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.14 fspmv()** [3/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.15 fspmv\_one()** [1/10]

```
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.16 fspmv\_mone()** [1/10]

```
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.17 fspmv\_one()** [2/10]

```
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```



**15.15.1.18 fspmv\_mone()** [2/10]

```

void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::COO_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.19 pfspmm()** [1/18]

```

void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.20 pfspmm()** [2/18]

```

void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.21 pfspmm()** [3/18]

```

void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]

```

**15.15.1.22 pfspmm\_one()** [1/2]

```

void pfspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.23 pfspmm\_mone()** [1/2]

```

void pfspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.24 pfspmm\_one()** [2/2]

```

void pfspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.25 pfspmm\_mone()** [2/2]

```

void pfspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.26 pfspmv()** [1/18]

```
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.27 pfspmv\_task()**

```
void pfspmv_task (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const index_t iStart,
    const index_t iStop,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.28 pfspmv()** [2/18]

```
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.29 pfspmv()** [3/18]

```
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.15.1.30 pfspmv\_one()** [1/8]

```
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.31 pfspmv\_mone()** [1/8]

```

void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.32 pfspmv\_one()** [2/8]

```

void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.33 pfspmv\_mone()** [2/8]

```

void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.34 fspmm()** [4/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.35 fspmm()** [5/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    index_t blockSize,
    typename Field::ConstElement_ptr x_,
    index_t ldx,
    typename Field::Element_ptr y_,
    index_t ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.36 fspmm\_simd\_aligned()** [2/2]

```

void fspmm_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.37 fspmm\_simd\_unaligned()** [2/2]

```

void fspmm_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.38 fspmm()** [6/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]

```

**15.15.1.39 fspmm\_one() [2/4]**

```

void fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.40 fspmm\_mone() [2/4]**

```

void fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.41 fspmm\_one\_simd\_aligned() [2/3]**

```

void fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.42 fspmm\_one\_simd\_unaligned() [2/3]**

```

void fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.143 fspmm\_mone\_simd\_aligned()** [2/3]

```

void fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.144 fspmm\_mone\_simd\_unaligned()** [2/3]

```

void fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.145 fspmv()** [4/21]

```

void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.146 fspmv()** [5/21]

```

void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.47 fspmv()** [6/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.15.1.48 fspmv\_one()** [3/10]

```
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.49 fspmv\_mone()** [3/10]

```
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.50 fspmv\_one()** [4/10]

```
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.51 fspmv\_mone()** [4/10]

```
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```



**15.15.152 pfsppmm()** [4/18]

```
void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]
```

**15.15.153 pfsppmm()** [5/18]

```
void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]
```

**15.15.154 pfsppmm()** [6/18]

```
void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.155 pfsppmm()** [7/18]

```
void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.56 pfsppmm()** [8/18]

```

void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    const int64_t kmax ) [inline]

```

**15.15.1.57 pfsppmm()** [9/18]

```

void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    const int64_t kmax ) [inline]

```

**15.15.1.58 pfsppmv()** [4/18]

```

void pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.59 pfsppmv()** [5/18]

```

void pfsppmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.60 pfspmv()** [6/18]

```

void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]

```

**15.15.1.61 fspmm()** [7/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.62 fspmm()** [8/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.63 fspmm()** [9/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]

```

**15.15.1.64 fspmv()** [7/21]

```

void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.65 fspmv()** [8/21]

```

void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.66 fspmv()** [9/21]

```

void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::CSR_HYB > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]

```

**15.15.1.67 pfspmm()** [10/18]

```

void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.68 pfspmm()** [11/18]

```

void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.69 pfsppmm()** [12/18]

```

void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.70 pfsppmm()** [13/18]

```

void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.71 pfsppmm()** [14/18]

```

void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    const int64_t kmax ) [inline]

```

**15.15.1.72 pfsppmm()** [15/18]

```

void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    const int64_t kmax ) [inline]

```

**15.15.1.73 pfspmm\_zo()** [1/2]

```

void pfspmm_zo (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    Func && func ) [inline]

```

**15.15.1.74 pfspmm\_zo()** [2/2]

```

void pfspmm_zo (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    Func && func ) [inline]

```

**15.15.1.75 pfspmv()** [7/18]

```

void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.76 pfspmv()** [8/18]

```

void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.77 pfspmv()** [9/18]

```
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.15.1.78 pfspmv\_one()** [3/8]

```
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.79 pfspmv\_mone()** [3/8]

```
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.80 pfspmv\_one()** [4/8]

```
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.81 pfspmv\_mone()** [4/8]

```
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.82 fspmm()** [10/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.83 fspmm()** [11/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.84 fspmm()** [12/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    const int64_t kmax ) [inline]

```

**15.15.1.85 fspmm\_mone()** [3/4]

```

void fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```



**15.15.1.86 fspmm\_one()** [3/4]

```

void fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.87 fspmm\_mone()** [4/4]

```

void fspmm_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.88 fspmm\_one()** [4/4]

```

void fspmm_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.89 fspmm\_one\_simd\_aligned()** [3/3]

```

void fspmm_one_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.90 fspmm\_one\_simd\_unaligned()** [3/3]

```

void fspmm_one_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.91 fspmm\_mone\_simd\_aligned()** [3/3]

```

void fspmm_mone_simd_aligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.92 fspmm\_mone\_simd\_unaligned()** [3/3]

```

void fspmm_mone_simd_unaligned (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x_,
    int ldx,
    typename Field::Element_ptr y_,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.93 fspmv()** [10/21]

```

void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.94 fspmv()** [11/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.95 fspmv()** [12/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.96 fspmv\_one()** [5/10]

```
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.97 fspmv\_mone()** [5/10]

```
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.98 fspmv\_one()** [6/10]

```
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.99 fspmv\_mone()** [6/10]

```
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.100 pfspmv()** [10/18]

```
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.101 pfspmv()** [11/18]

```
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.102 pfspmv()** [12/18]

```
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.103 pfspmv\_one()** [5/8]

```
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.104 pfspmv\_mone()** [5/8]

```
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.105 pfspmv\_one()** [6/8]

```
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.106 pfspmv\_mone()** [6/8]

```
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.107 fspmv()** [13/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.108 fspmv\_simd()** [1/4]

```
void fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.109 fspmv()** [14/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.110 fspmv\_simd()** [2/4]

```
void fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.111 fspmv()** [15/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.112 fspmv\_one()** [7/10]

```
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.113 fspmv\_mone()** [7/10]

```
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.114 fspmv\_one()** [8/10]

```

void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.115 fspmv\_mone()** [8/10]

```

void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.116 fspmv\_one\_simd()** [1/2]

```

void fspmv_one_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.117 fspmv\_mone\_simd()** [1/2]

```

void fspmv_mone_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::ELL_simd_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.118 pfspmm()** [16/18]

```

void pfspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.119 pfsppmm()** [17/18]

```

void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.120 pfsppmm()** [18/18]

```

void pfsppmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    uint64_t kmax ) [inline]

```

**15.15.1.121 pfspmv()** [13/18]

```

void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.122 pfspmv()** [14/18]

```

void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]

```



**15.15.1.123 pfspmv()** [15/18]

```

void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    uint64_t kmax ) [inline]

```

**15.15.1.124 fspmm()** [13/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::GenericTag ) [inline]

```

**15.15.1.125 fspmm()** [14/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    FieldCategories::UnparametricTag ) [inline]

```

**15.15.1.126 fspmm()** [15/15]

```

void fspmm (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    size_t blockSize,
    typename Field::ConstElement_ptr x,
    int ldx,
    typename Field::Element_ptr y,
    int ldy,
    uint64_t kmax ) [inline]

```

**15.15.1.127 fspmv()** [16/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.128 fspmv()** [17/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.129 fspmv()** [18/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::HYB_ZO > & A,
    typename Field::ConstElement_ptr x,
    typename Field::Element_ptr y,
    uint64_t kmax ) [inline]
```

**15.15.1.130 pfspmv()** [16/18]

```
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.131 pfspmv()** [17/18]

```
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.132 pfspmv()** [18/18]

```
void pfspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const int64_t kmax ) [inline]
```

**15.15.1.133 pfspmv\_one()** [7/8]

```
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.134 pfspmv\_mone()** [7/8]

```
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.135 pfspmv\_one()** [8/8]

```
void pfspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.136 pfspmv\_mone()** [8/8]

```
void pfspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.137 fspmv()** [19/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.138 fspmv\_simd()** [3/4]

```
void fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.139 fspmv()** [20/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.140 fspmv\_simd()** [4/4]

```
void fspmv_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.141 fspmv()** [21/21]

```
void fspmv (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    const uint64_t kmax ) [inline]
```

**15.15.1.142 fspmv\_one()** [9/10]

```
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.143 fspmv\_mone()** [9/10]

```
void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::GenericTag ) [inline]
```

**15.15.1.144 fspmv\_one\_simd()** [2/2]

```
void fspmv_one_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.145 fspmv\_mone\_simd()** [2/2]

```
void fspmv_mone_simd (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.146 fspmv\_one()** [10/10]

```
void fspmv_one (
    const Field & F,
    const Sparse< Field, SparseMatrix_t::SELL_ZO > & A,
    typename Field::ConstElement_ptr x_,
    typename Field::Element_ptr y_,
    FieldCategories::UnparametricTag ) [inline]
```

**15.15.1.147 fspmv\_mone()** [10/10]

```

void fspmv_mone (
    const Field & F,
    const Sparse< Field, SparseMatrix\_t::SELL\_ZO > & A,
    typename Field::ConstElement\_ptr x_,
    typename Field::Element\_ptr y_,
    FieldCategories::UnparametricTag ) [inline]

```

**15.16 FFLAS::StrategyParameter Namespace Reference****Data Structures**

- struct [Fixed](#)
- struct [Grain](#)
- struct [Threads](#)
- struct [ThreeD](#)
- struct [ThreeDAdaptive](#)
- struct [ThreeDInPlace](#)
- struct [TwoD](#)
- struct [TwoDAdaptive](#)

**15.17 FFLAS::StructureHelper Namespace Reference**

[StructureHelper](#) for ftrsm.

**Data Structures**

- struct [Hybrid](#)
- struct [Iterative](#)
- struct [Recursive](#)

**15.17.1 Detailed Description**

[StructureHelper](#) for ftrsm.

**15.18 FFLAS::vectorised Namespace Reference****Namespaces**

- namespace [unswitch](#)

## Data Structures

- struct [HelperMod](#)
- struct [HelperMod](#)< Field, ElementCategories::MachineIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::FixedPrecIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::MachineFloatTag >

## Functions

- template<class SimdT, class Element, bool positive>  
std::enable\_if< [is\\_simd](#)< SimdT >::value, void >::type [VEC\\_ADD](#) (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
- template<bool positive, class Element, class T1, class T2 >  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< Element >::value, void >::type [addp](#) (Element \*T, const Element \*TA, const Element \*TB, size\_t n, Element p, T1 min\_, T2 max\_)
- template<class SimdT, class Element, bool positive>  
std::enable\_if< [is\\_simd](#)< SimdT >::value, void >::type [VEC\\_SUB](#) (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
- template<bool positive, class Element, class T1, class T2 >  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< Element >::value, void >::type [subp](#) (Element \*T, const Element \*TA, const Element \*TB, const size\_t n, const Element p, const T1 min\_, const T2 max\_)
- template<class Element >  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< Element >::value, void >::type [add](#) (Element \*T, const Element \*TA, const Element \*TB, size\_t n)
- template<class Element >  
std::enable\_if< [FFLAS::support\\_simd\\_add](#)< Element >::value, void >::type [sub](#) (Element \*T, const Element \*TA, const Element \*TB, size\_t n)
- template<class T >  
std::enable\_if< !std::is\_integral< T >::value, T >::type [reduce](#) (T A, T B)
- template<class T >  
std::enable\_if< std::is\_integral< T >::value, T >::type [reduce](#) (T A, T B)
- template<> Givaro::Integer [reduce](#) (Givaro::Integer A, Givaro::Integer B)
- float [reduce](#) (float A, float B, float invB, float min, float max)
- double [reduce](#) (double A, double B, double invB, double min, double max)
- [int64\\_t reduce](#) ([int64\\_t](#) A, [int64\\_t](#) p, double invp, double min, double max, [int64\\_t](#) pow50rem)
- template<class Field >  
[Field::Element reduce](#) (typename [Field::Element](#) A, [HelperMod](#)< Field, ElementCategories::MachineIntTag > &H)
- template<class Field >  
[Field::Element reduce](#) (typename [Field::Element](#) A, [HelperMod](#)< Field, ElementCategories::MachineFloatTag > &H)
- template<class Field >  
[Field::Element reduce](#) (typename [Field::Element](#) A, [HelperMod](#)< Field, ElementCategories::ArbitraryPrecIntTag > &H)
- template<class Field >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [modp](#) (const [Field](#) &F, typename [Field::ConstElement\\_ptr](#) U, const size\_t &n, typename [Field::Element\\_ptr](#) T)
- template<class Field >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [modp](#) (const [Field](#) &F, typename [Field::ConstElement\\_ptr](#) U, const size\_t &n, const size\_t &incX, typename [Field::Element\\_ptr](#) T)
- template<class Field >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [scalp](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) T, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) U, const size\_t n)

- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX)`

## 15.18.1 Function Documentation

### 15.18.1.1 VEC\_ADD()

```
std::enable_if< is_simd< SimdT >::value, void >::type VEC_ADD (
    SimdT & C,
    SimdT & A,
    SimdT & B,
    SimdT & Q,
    SimdT & T,
    SimdT & P,
    SimdT & NEGP,
    SimdT & MIN,
    SimdT & MAX ) [inline]
```

### 15.18.1.2 addp()

```
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type addp (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n,
    Element p,
    T1 min_,
    T2 max_ ) [inline]
```

### 15.18.1.3 VEC\_SUB()

```
std::enable_if< is_simd< SimdT >::value, void >::type VEC_SUB (
    SimdT & C,
    SimdT & A,
    SimdT & B,
    SimdT & Q,
    SimdT & T,
    SimdT & P,
    SimdT & NEGP,
    SimdT & MIN,
    SimdT & MAX ) [inline]
```



**15.18.1.4 subp()**

```
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type subp (
    Element * T,
    const Element * TA,
    const Element * TB,
    const size_t n,
    const Element p,
    const T1 min_,
    const T2 max_ ) [inline]
```

**15.18.1.5 add()**

```
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type add (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n ) [inline]
```

**15.18.1.6 sub()**

```
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type sub (
    Element * T,
    const Element * TA,
    const Element * TB,
    size_t n ) [inline]
```

**15.18.1.7 reduce() [1/9]**

```
std::enable_if<!std::is_integral< T >::value, T >::type reduce (
    T A,
    T B ) [inline]
```

**15.18.1.8 reduce() [2/9]**

```
std::enable_if< std::is_integral< T >::value, T >::type reduce (
    T A,
    T B ) [inline]
```

**15.18.1.9 reduce()** [3/9]

```
Givaro::Integer reduce (
    Givaro::Integer A,
    Givaro::Integer B ) [inline]
```

**15.18.1.10 reduce()** [4/9]

```
float reduce (
    float A,
    float B,
    float invB,
    float min,
    float max ) [inline]
```

**15.18.1.11 reduce()** [5/9]

```
double reduce (
    double A,
    double B,
    double invB,
    double min,
    double max ) [inline]
```

**15.18.1.12 reduce()** [6/9]

```
int64_t reduce (
    int64_t A,
    int64_t P,
    double invp,
    double min,
    double max,
    int64_t pow50rem ) [inline]
```

**15.18.1.13 reduce()** [7/9]

```
Field::Element reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::MachineIntTag > & H ) [inline]
```

**15.18.1.14 reduce()** [8/9]

```
Field::Element reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::MachineFloatTag > & H ) [inline]
```

**15.18.1.15 reduce()** [9/9]

```
Field::Element reduce (
    typename Field::Element A,
    HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > & H ) [inline]
```

**15.18.1.16 modp()** [1/2]

```
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    typename Field::Element_ptr T ) [inline]
```

**15.18.1.17 modp()** [2/2]

```
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    const size_t & incX,
    typename Field::Element_ptr T ) [inline]
```

**15.18.1.18 scalp()** [1/2]

```
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp
(
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n ) [inline]
```

### 15.18.1.19 scalp() [2/2]

```
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp
(
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    const size_t & incX ) [inline]
```

## 15.19 FFLAS::vectorised::unswitch Namespace Reference

### Functions

- template<class Field >  
std::enable\_if<!FFLAS::support\_simd\_mod< typenameField::Element >::value &&FFLAS::support\_fast\_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement\_ptr U, const size\_t &n, typename Field::Element\_ptr T, HelperMod< Field > &H)
- template<class Field >  
std::enable\_if< FFLAS::support\_fast\_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement\_ptr U, const size\_t &n, const size\_t &incX, typename Field::Element\_ptr T, HelperMod< Field > &H)
- template<class Field >  
std::enable\_if<!FFLAS::support\_simd\_mod< typenameField::Element >::value &&FFLAS::support\_fast\_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element\_ptr T, const typename Field::Element alpha, typename Field::ConstElement\_ptr U, const size\_t n, HelperMod< Field > &H)
- template<class Field >  
std::enable\_if< FFLAS::support\_fast\_mod< typenameField::Element >::value, void >::type scalp (const Field &F, typename Field::Element\_ptr T, const typename Field::Element alpha, typename Field::ConstElement\_ptr U, const size\_t n, const size\_t &incX, HelperMod< Field > &H)

### 15.19.1 Function Documentation

#### 15.19.1.1 modp() [1/2]

```
std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    typename Field::Element_ptr T,
    HelperMod< Field > & H ) [inline]
```

**15.19.1.2 modp()** [2/2]

```
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (
    const Field & F,
    typename Field::ConstElement_ptr U,
    const size_t & n,
    const size_t & incX,
    typename Field::Element_ptr T,
    HelperMod< Field > & H ) [inline]
```

**15.19.1.3 scalp()** [1/2]

```
std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod<
typenameField::Element >::value, void >::type scalp (
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    HelperMod< Field > & H ) [inline]
```

**15.19.1.4 scalp()** [2/2]

```
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp
(
    const Field & F,
    typename Field::Element_ptr T,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr U,
    const size_t n,
    const size_t & incX,
    HelperMod< Field > & H ) [inline]
```

**15.20 FFPACK Namespace Reference**

Finite Field **PACK** Set of elimination based routines for dense linear algebra.

**Namespaces**

- namespace [Protected](#)

## Data Structures

- class [callLUdivine\\_small](#)
- class [callLUdivine\\_small< double >](#)
- class [callLUdivine\\_small< float >](#)
- class [CharpolyFailed](#)
- class [CheckerImplem\\_charpoly](#)
- class [CheckerImplem\\_Det](#)
- class [CheckerImplem\\_invert](#)
- class [CheckerImplem\\_PLUQ](#)
- class [Failure](#)  
*A precondition failed.*
- struct [rns\\_double](#)
- struct [rns\\_double\\_elt](#)
- struct [rns\\_double\\_elt\\_cstptr](#)
- struct [rns\\_double\\_elt\\_ptr](#)
- struct [rns\\_double\\_extended](#)
- class [RNSInteger](#)
- class [RNSIntegerMod](#)
- class [rnsRandIter](#)

## Typedefs

- template<class [Field](#) >  
using [Checker\\_PLUQ](#) = [FFLAS::Checker\\_Empty](#)< [Field](#) >
- template<class [Field](#) >  
using [Checker\\_Det](#) = [FFLAS::Checker\\_Empty](#)< [Field](#) >
- template<class [Field](#) >  
using [Checker\\_invert](#) = [FFLAS::Checker\\_Empty](#)< [Field](#) >
- template<class [Field](#) , class Polynomial >  
using [Checker\\_charpoly](#) = [FFLAS::Checker\\_Empty](#)< [Field](#) >
- template<class [Field](#) >  
using [ForceCheck\\_PLUQ](#) = [CheckerImplem\\_PLUQ](#)< [Field](#) >
- template<class [Field](#) >  
using [ForceCheck\\_Det](#) = [CheckerImplem\\_Det](#)< [Field](#) >
- template<class [Field](#) >  
using [ForceCheck\\_invert](#) = [CheckerImplem\\_invert](#)< [Field](#) >
- template<class [Field](#) , class Polynomial >  
using [ForceCheck\\_charpoly](#) = [CheckerImplem\\_charpoly](#)< [Field](#), Polynomial >

## Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)  
*Conversion of a permutation from LAPACK format to Math format.*
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)  
*Conversion of a permutation from Maths format to LAPACK format.*
- template<class [Field](#) >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P)  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*

- template<class [Field](#) >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const [FFLAS::ParSeqHelper::Sequential](#) seq)
- template<class [Field](#) , class [Cut](#) , class [Param](#) >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const [FFLAS::ParSeqHelper::Parallel](#)< [Cut](#), [Param](#) > par)
- template<class [Field](#) >  
void [MonotonicApplyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t R)  
*Apply a R-monotonically increasing permutation P, to the matrix A.*
- template<class [Field](#) >  
void [fgetrs](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename [Field::Element\\_ptr](#) B, const size\_t ldb, int \*info)  
*Solve the system  $AX = B$  or  $XA = B$ .*
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [fgetrs](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename [Field::Element\\_ptr](#) X, const size\_t ldX, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, int \*info)  
*Solve the system  $AX = B$  or  $XA = B$ .*
- template<class [Field](#) >  
size\_t [fgesv](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, int \*info)  
*Square system solver.*
- template<class [Field](#) >  
size\_t [fgesv](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldX, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, int \*info)  
*Rectangular system solver.*
- template<class [Field](#) >  
void [ftrtri](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t threshold=\_\_FFLASFFPACK\_FTRTRI\_THRESHOLD)  
*Compute the inverse of a triangular matrix.*
- template<class [Field](#) >  
void [trinv\\_left](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) L, const size\_t ldL, typename [Field::Element\\_ptr](#) X, const size\_t ldX)
- template<class [Field](#) >  
void [ftrrm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Compute the product of two triangular matrices of opposite shape.*
- template<class [Field](#) >  
void [ftrstr](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diagA, const [FFLAS::FFLAS\\_DIAG](#) diagB, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const size\_t threshold=\_\_FFLASFFPACK\_FTRSTR\_THRESHOLD)  
*Solve a triangular system with a triangular right hand side of the same shape.*
- template<class [Field](#) >  
void [ftrssyr2k](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diagA, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const size\_t threshold=\_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD)  
*Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.*

- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field , class Cut , class Param >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr D, const size_t incD, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`  
*Compute a PLUQ factorization of the given matrix.*
- `template<class Field >`  
`size_t pPLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential &PHelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)`
- `template<class Field , class Cut , class Param >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< Cut, Param > &PHelper)`
- `template<class Field >`  
`size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive, const size_t cutoff=__FFLASFFPACK_LUDIVINE_THRESHOLD)`  
*Compute the CUP or PLE factorization of the given matrix.*
- `template<class Field >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Compute the Column Echelon form of the input matrix in-place.*
- `template<class Field >`  
`size_t pColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, bool transform=false, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FpackTileRecursive)`
- `template<class Field , class PSHelper >`  
`size_t ColumnEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag, const PSHelper &psH)`
- `template<class Field >`  
`size_t RowEchelonForm (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`  
*Compute the Row Echelon form of the input matrix in-place.*



- template<class [Field](#) >  
size\_t [pRowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class PSHelper >  
size\_t [RowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const PSHelper &psH)
- template<class [Field](#) >  
size\_t [ReducedColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))  
*Compute the Reduced Column Echelon form of the input matrix in-place.*
- template<class [Field](#) >  
size\_t [pReducedColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class PSHelper >  
size\_t [ReducedColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const PSHelper &psH)
- template<class [Field](#) >  
size\_t [ReducedRowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))  
*Compute the Reduced Row Echelon form of the input matrix in-place.*
- template<class [Field](#) >  
size\_t [pReducedRowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class PSHelper >  
size\_t [ReducedRowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const PSHelper &psH)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [Invert](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, int &nullity)  
*Invert the given matrix in place or computes its nullity if it is singular.*
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [Invert](#) (const [Field](#) &F, const size\_t M, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [Invert2](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- template<class PolRing >  
std::list< typename PolRing::Element > & [CharPoly](#) (const PolRing &R, std::list< typename PolRing::Element > &charp, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t lda, typename PolRing::Domain\_t::RandIter &G, const FFPACK\_CHARPOLY\_TAG CharpTag=[FfpackAuto](#), const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)  
*Compute the characteristic polynomial of the matrix A.*
- template<class PolRing >  
PolRing::Element & [CharPoly](#) (const PolRing &R, typename PolRing::Element &charp, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t lda, typename PolRing::Domain\_t::RandIter &G, const FFPACK\_CHARPOLY\_TAG CharpTag=[FfpackAuto](#), const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)

*Compute the characteristic polynomial of the matrix A.*

- template<class PolRing >  
PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t lda, const FFPACK\_CHARPOLY\_TAG Charp←Tag=FFpackAuto, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)

*Compute the characteristic polynomial of the matrix A.*

- template<class Field , class Polynomial >  
Polynomial & MinPoly (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda)

*Compute the minimal polynomial of the matrix A.*

- template<class Field , class Polynomial , class RandIter >  
Polynomial & MinPoly (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, RandIter &G)

*Compute the minimal polynomial of the matrix A.*

- template<class Field , class Polynomial >  
Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr v, const size\_t incv)

*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*

- template<class Field >  
size\_t Rank (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda)

*Computes the rank of the given matrix using a PLUQ factorization.*

- template<class Field >  
size\_t pRank (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t numthreads=0)
- template<class Field , class PSHelper >  
size\_t Rank (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const PSHelper &psH)
- template<class Field >  
bool IsSingular (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t lda)

*Returns true if the given matrix is singular.*

- template<class Field >  
Field::Element & Det (const Field &F, typename Field::Element &det, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P=NULL, size\_t \*Q=NULL)

*Returns the determinant of the given square matrix.*

- template<class Field >  
Field::Element & pDet (const Field &F, typename Field::Element &det, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t numthreads=0, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class Field , class PSHelper >  
Field::Element & Det (const Field &F, typename Field::Element &det, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const PSHelper &psH, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class Field >  
Field::Element\_ptr Solve (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr x, const int incx, typename Field::ConstElement\_ptr b, const int incb)

*Solves a linear system  $AX = b$  using PLUQ factorization.*

- template<class Field , class PSHelper >  
Field::Element\_ptr Solve (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr x, const int incx, typename Field::ConstElement\_ptr b, const int incb, PSHelper &psH)
- template<class Field >  
Field::Element\_ptr pSolve (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr x, const int incx, typename Field::ConstElement\_ptr b, const int incb, size\_t numthreads=0)

- template<class [Field](#) >  
 \*void [RandomNullSpaceVector](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
*Solve  $LX = B$  or  $XL = B$  in place.*
- template<class [Field](#) >  
 size\_t [NullSpaceBasis](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &NS, size\_t &ldn, size\_t &NSdim)  
*Computes a basis of the Left/Right nullspace of the matrix A.*
- template<class [Field](#) >  
 size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))  
*Computes the row rank profile of A.*
- template<class [Field](#) >  
 size\_t [pRowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class PSHelper >  
 size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)
- template<class [Field](#) >  
 size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))  
*Computes the column rank profile of A.*
- template<class [Field](#) >  
 size\_t [pColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class PSHelper >  
 size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)
- void [RankProfileFromLU](#) (const size\_t \*P, const size\_t N, const size\_t R, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag)  
*Recovers the column/row rank profile from the permutation of an LU decomposition.*
- size\_t [LeadingSubmatrixRankProfiles](#) (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP)  
*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*
- template<class [Field](#) >  
 size\_t [RowRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)  
*RowRankProfileSubmatrixIndices.*
- template<class [Field](#) >  
 size\_t [ColRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)  
*Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.*
- template<class [Field](#) >  
 size\_t [RowRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)  
*Computes the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*
- template<class [Field](#) >  
 size\_t [ColRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)  
*Compute the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*
- template<class [Field](#) >  
 void [getTriangular](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false)

*Extracts a triangular matrix from a compact storage  $A=L\backslash U$  of rank  $R$ .*

- `template<class Field >`  
`void getTriangular (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr A, const size_t Ida)`

*Cleans up a compact storage  $A=L\backslash U$  to reveal a triangular matrix of rank  $R$ .*

- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::Element_ptr T, const size_t Idt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*

- `template<class Field >`  
`void getEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t Ida, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank  $R$ .*

- `template<class Field >`  
`void getEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::Element_ptr T, const size_t Idt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*

- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::Element_ptr T, const size_t Idt, const bool OnlyNonZeroVectors=false, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*

- `template<class Field >`  
`void getReducedEchelonForm (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, typename Field::Element_ptr A, const size_t Ida, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*

- `template<class Field >`  
`void getReducedEchelonTransform (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const size_t M, const size_t N, const size_t R, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::Element_ptr T, const size_t Idt, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by RowEchelonForm or ColumnEchelonForm.*

- `void PLUQtoEchelonPermutation (const size_t N, const size_t R, const size_t *P, size_t *outPerm)`  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*

- `template<class Field >`  
`Field::Element_ptr LQUPtoInverseOfFullRankMinor (const Field &F, const size_t rank, typename Field::Element_ptr A_factors, const size_t Ida, const size_t *QtPointer, typename Field::Element_ptr X, const size_t Idx)`

*LQUPtoInverseOfFullRankMinor.*

- `template<class Field >`  
`void RandomNullSpaceVector (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr X, const size_t incX)`

*Solve  $LX = B$  or  $XL = B$  in place.*

- `template<class Field >`  
`void solveLB (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)`
- `template<class Field >`  
`void solveLB2 (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, typename Field::Element_ptr L, const size_t ldl, const size_t *Q, typename Field::Element_ptr B, const size_t ldb)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t lda)`
- `template<class Field >`  
`Field::Element_ptr buildMatrix (const Field &F, typename Field::ConstElement_ptr E, typename Field::ConstElement_ptr C, const size_t lda, const size_t *B, const size_t *T, const size_t me, const size_t mc, const size_t lambda, const size_t mu)`
- `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr CharPoly (const FFPACK::RNSInteger< FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr charp, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)`
- `template<> Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly (const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > &R, Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element &charp, const size_t N, Givaro::Integer *A, const size_t lda, Givaro::ZRing< Givaro::Integer >::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag, size_t degree)`
- `template<class PSHelper >`  
`FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det (const FFPACK::RNSInteger< FFPACK::rns_double > &F, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr &det, const size_t N, typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A, const size_t lda, const PSHelper &psH)`
- `template<class PSHelper >`  
`Givaro::Integer & Det (const Givaro::ZRing< Givaro::Integer > &F, Givaro::Integer &det, const size_t N, Givaro::Integer *A, const size_t lda, const PSHelper &psH, size_t *P, size_t *Q)`
- `template<class Field >`  
`bool fsytrf_BC_Crout (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)`
- `template<class Field >`  
`size_t fsytrf_BC_RL (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM_BC_RL (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_LOW_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM_BC_Crout (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P)`
- `template<class Field >`  
`size_t fsytrf_UP_RPM (const Field &F, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, size_t *P, size_t BCThreshold)`
- `template<class Field >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, FFLAS::ParSeqHelper::Sequential seq, size_t threshold)`
- `template<class Field , class Cut , class Param >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv, FFLAS::ParSeqHelper::Parallel< Cut, Param > par, size_t threshold)`

- `template<class Field >`  
`size_t fsytrf_RPM (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t threshold)`
- `template<class Field >`  
`void getTridiagonal (const Field &F, const size_t N, const size_t R, typename Field::ConstElement_ptr A, const size_t lda, size_t *P, typename Field::Element_ptr T, const size_t ldt)`
- `template<class Field >`  
`size_t LUdivine_gauss (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`
- `template<class Field >`  
`size_t LUdivine_small (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`
- `template<class Field >`  
`size_t LUdivine (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag, const size_t cutoff)`
- `template<> size_t LUdivine (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Givaro::Integer *A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag, const size_t cutoff)`
- `template<class Field >`  
`void MonotonicCompress (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)`
- `template<class Field >`  
`void MonotonicCompressMorePivots (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t rowstomove, const size_t lenP)`
- `template<class Field >`  
`void MonotonicCompressCycles (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t lenP)`
- `template<class Field >`  
`void MonotonicExpand (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)`
- `template<class Field >`  
`void applyP_block (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P)`
- `template<class Field >`  
`void doApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field, class Cut, class Param >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyS (T *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`



- `template<class Field >`  
`void doApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field, class Cut, class Param >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyT (T *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- `void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $MathP$  as a MathPermutation format.*
- `void composePermutationsMLM (size_t *MathP1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $MathP1 \times \text{Diag}(I_R, P2)$  where  $MathP1$  is a MathPermutation and  $P2$  a LAPACK permutation and store the result in  $MathP1$  as a MathPermutation format.*
- `void cyclic_shift_mathPerm (size_t *P, const size_t s)`
- `template<class Field >`  
`void cyclic_shift_row_col (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<class Field >`  
`void cyclic_shift_row (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<typename T >`  
`void cyclic_shift_row (const RNSIntegerMod< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<class Field >`  
`void cyclic_shift_col (const Field &F, typename Field::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<typename T >`  
`void cyclic_shift_col (const RNSIntegerMod< T > &F, typename T::Element_ptr A, size_t m, size_t n, size_t lda)`
- `template<class Field >`  
`size_t PLUQ_basecaseV3 (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element *A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ_basecaseV2 (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element *A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ_basecaseCrout (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t _PLUQ (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold)`
- `template<class Cut, class Param >`  
`size_t PLUQ (const Givaro::Modular< Givaro::Integer > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Givaro::Integer *A, const size_t lda, size_t *P, size_t *Q, size_t BCThreshold, FFLAS::ParSeqHelper::Parallel< Cut, Param > &PSHelper)`

- `template<class Field >`  
`void threads_fgemm (const size_t m, const size_t n, const size_t r, int nbthreads, size_t *W1, size_t *W2, size_t *W3, size_t gamma)`
- `template<class Field >`  
`void threads_ftrsm (const size_t m, const size_t n, int nbthreads, size_t *t1, size_t *t2)`
- `template<class Field >`  
`size_t PLUQ (const Field &Fi, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, type-name Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads > &PSHelper)`
- `template<> rns_double_elt_ptr fflas_const_cast (rns_double_elt_cstptr x)`
- `template<> rns_double_elt_cstptr fflas_const_cast (rns_double_elt_ptr x)`
- `template<typename Base_t >`  
`void cyclic_shift_row_col (Base_t *A, size_t m, size_t n, size_t lda)`
- `template INST_OR_DECL void cyclic_shift_row (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT *A, size_t m, size_t n, size_t lda)`
- `template INST_OR_DECL void cyclic_shift_col (const FFLAS_FIELD< FFLAS_ELT > &F, FFLAS_ELT *A, size_t m, size_t n, size_t lda)`
- `template INST_OR_DECL void applyP (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, FFLAS_ELT *A, const size_t lda, const size_t *P)`
- `template INST_OR_DECL void fgetrs (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t R, FFLAS_ELT *A, const size_t lda, const size_t *P, const size_t *Q, FFLAS_ELT *B, const size_t ldb, int *info)`
- `template INST_OR_DECL FFLAS_ELT * fgetrs (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, const size_t R, FFLAS_ELT *A, const size_t lda, const size_t *P, const size_t *Q, FFLAS_ELT *X, const size_t ldx, const FFLAS_ELT *B, const size_t ldb, int *info)`
- `template INST_OR_DECL size_t fgesv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *B, const size_t ldb, int *info)`
- `template INST_OR_DECL size_t fgesv (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, FFLAS_ELT *A, const size_t lda, FFLAS_ELT *X, const size_t ldx, const FFLAS_ELT *B, const size_t ldb, int *info)`
- `template INST_OR_DECL void ftrtri (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG Diag, const size_t N, FFLAS_ELT *A, const size_t lda, const size_t threshold)`
- `template INST_OR_DECL void trinv_left (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t N, const FFLAS_ELT *L, const size_t ldl, FFLAS_ELT *X, const size_t ldx)`
- `template INST_OR_DECL void ftrtrm (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_DIAG diag, const size_t N, FFLAS_ELT *A, const size_t lda)`
- `template INST_OR_DECL size_t PLUQ (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q)`
- `template INST_OR_DECL size_t LUdivine (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const FFPACK_LU_TAG LuTag, const size_t cutoff)`
- `template INST_OR_DECL size_t LUdivine_small (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q, const FFPACK_LU_TAG LuTag)`
- `template INST_OR_DECL size_t LUdivine_gauss (const FFLAS_FIELD< FFLAS_ELT > &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Q, const FFPACK_LU_TAG LuTag)`
- `template INST_OR_DECL size_t RowEchelonForm (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag)`
- `template INST_OR_DECL size_t ReducedRowEchelonForm (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t M, const size_t N, FFLAS_ELT *A, const size_t lda, size_t *P, size_t *Qt, const bool transform, const FFPACK_LU_TAG LuTag)`



- template [INST\\_OR\\_DECL](#) [size\\_t](#) [ColumnEchelonForm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) M, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, [size\\_t](#) \*P, [size\\_t](#) \*Qt, const bool transform, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) [size\\_t](#) [ReducedColumnEchelonForm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) M, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, [size\\_t](#) \*P, [size\\_t](#) \*Qt, const bool transform, const [FFPACK\\_LU\\_TAG](#) LuTag)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [Invert](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) M, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, int &nullity)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [Invert](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) M, const [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, [FFLAS\\_ELT](#) \*X, const [size\\_t](#) ldx, int &nullity)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [Invert2](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) M, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, [FFLAS\\_ELT](#) \*X, const [size\\_t](#) ldx, int &nullity)
- template [INST\\_OR\\_DECL](#) std::list< Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element > & [CharPoly](#) (const Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > > &R, std::list< Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element > &charp, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) >::RandIter &G, const [FFPACK\\_CHARPOLY\\_TAG](#) CharpTag, const [size\\_t](#) degree)
- template [INST\\_OR\\_DECL](#) Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element & [CharPoly](#) (const Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > > &R, Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element &charp, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) >::RandIter &G, const [FFPACK\\_CHARPOLY\\_TAG](#) CharpTag, const [size\\_t](#) degree)
- template [INST\\_OR\\_DECL](#) Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element & [CharPoly](#) (const Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > > &R, Givaro::Poly1Dom< [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > >::Element &charp, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, const [FFPACK\\_CHARPOLY\\_TAG](#) CharpTag, const [size\\_t](#) degree)
- template [INST\\_OR\\_DECL](#) std::vector< [FFLAS\\_ELT](#) > & [MinPoly](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, std::vector< [FFLAS\\_ELT](#) > &minP, const [size\\_t](#) N, const [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) >::RandIter &G)
- template [INST\\_OR\\_DECL](#) std::vector< [FFLAS\\_ELT](#) > & [MinPoly](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, std::vector< [FFLAS\\_ELT](#) > &minP, const [size\\_t](#) N, const [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda)
- template [INST\\_OR\\_DECL](#) std::vector< [FFLAS\\_ELT](#) > & [MatVecMinPoly](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, std::vector< [FFLAS\\_ELT](#) > &minP, const [size\\_t](#) N, const [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, const [FFLAS\\_ELT](#) \*V, const [size\\_t](#) incv)
- template [INST\\_OR\\_DECL](#) [size\\_t](#) [KrylovElim](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) M, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, [size\\_t](#) \*P, [size\\_t](#) \*Q, const [size\\_t](#) deg, [size\\_t](#) \*iterates, [size\\_t](#) \*inviterates, const [size\\_t](#) maxit, [size\\_t](#) virt)
- template [INST\\_OR\\_DECL](#) [size\\_t](#) [SpecRankProfile](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) M, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, const [size\\_t](#) deg, [size\\_t](#) \*rankProfile)
- template [INST\\_OR\\_DECL](#) [size\\_t](#) [Rank](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) M, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda)
- template [INST\\_OR\\_DECL](#) bool [IsSingular](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) M, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) & [Det](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, [FFLAS\\_ELT](#) &det, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, [size\\_t](#) \*P, [size\\_t](#) \*Q)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) & [Det](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, [FFLAS\\_ELT](#) &det, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, const [FFLAS::ParSeqHelper::Parallel](#)< [FFLAS::CuttingStrategy::Recursive](#), [FFLAS::StrategyParameter::Threads](#) > &parH, [size\\_t](#) \*P, [size\\_t](#) \*Q)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [Solve](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [size\\_t](#) M, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, [FFLAS\\_ELT](#) \*x, const int incx, const [FFLAS\\_ELT](#) \*b, const int incb)
- template [INST\\_OR\\_DECL](#) void [solveLB](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [size\\_t](#) M, const [size\\_t](#) N, const [size\\_t](#) R, [FFLAS\\_ELT](#) \*L, const [size\\_t](#) ldl, const [size\\_t](#) \*Q, [FFLAS\\_ELT](#) \*B, const [size\\_t](#) ldb)
- template [INST\\_OR\\_DECL](#) void [solveLB2](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [size\\_t](#) M, const [size\\_t](#) N, const [size\\_t](#) R, [FFLAS\\_ELT](#) \*L, const [size\\_t](#) ldl, const [size\\_t](#) \*Q, [FFLAS\\_ELT](#) \*B, const [size\\_t](#) ldb)
- template [INST\\_OR\\_DECL](#) void [RandomNullSpaceVector](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [size\\_t](#) M, const [size\\_t](#) N, [FFLAS\\_ELT](#) \*A, const [size\\_t](#) lda, [FFLAS\\_ELT](#) \*X, const [size\\_t](#) incX)

- template `INST_OR_DECL` `size_t` `NullSpaceBasis` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&NS, `size_t` &ldn, `size_t` &NSdim)
- template `INST_OR_DECL` `size_t` `RowRankProfile` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t` `ColumnRankProfile` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t` `RowRankProfileSubmatrixIndices` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rowindices, `size_t` \*&colindices, `size_t` &R)
- template `INST_OR_DECL` `size_t` `ColRankProfileSubmatrixIndices` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rowindices, `size_t` \*&colindices, `size_t` &R)
- template `INST_OR_DECL` `size_t` `RowRankProfileSubmatrix` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&X, `size_t` &R)
- template `INST_OR_DECL` `size_t` `ColRankProfileSubmatrix` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*&X, `size_t` &R)
- template `INST_OR_DECL` void `getTriangular< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const bool OnlyNonZeroVectors)
- template `INST_OR_DECL` void `getTriangular< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` void `getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const bool OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const bool OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `FFLAS_ELT` \* `LQUPtoInverseOfFullRankMinor` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` rank, `FFLAS_ELT` \*A\_factors, const `size_t` lda, const `size_t` \*QtPointer, `FFLAS_ELT` \*X, const `size_t` ldx)
- template<class T, class CT = const T>  
T `fflas_const_cast` (CT x)
- `Failure` & `failure` ()
- template<class T >  
bool `isOdd` (const T &a)
- bool `isOdd` (const float &a)
- bool `isOdd` (const double &a)

- template<class [Field](#) , class Randlter >  
[Field::Element\\_ptr NonZeroRandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda, Randlter &G)  
*Random non-zero Matrix.*
- template<class [Field](#) , class Randlter >  
[Field::Element\\_ptr NonZeroRandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda)  
*Random non-zero Matrix.*
- template<class [Field](#) , class Randlter >  
[Field::Element\\_ptr RandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda, Randlter &G)  
*Random Matrix.*
- template<class [Field](#) >  
[Field::Element\\_ptr RandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda)  
*Random Matrix.*
- template<class [Field](#) , class Randlter >  
[Field::Element\\_ptr RandomTriangularMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_DIAG](#) Diag, bool nonsingular, typename [Field::Element\\_ptr](#) A, size\_t lda, Randlter &G)  
*Random Triangular Matrix.*
- template<class [Field](#) >  
[Field::Element\\_ptr RandomTriangularMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_DIAG](#) Diag, bool nonsingular, typename [Field::Element\\_ptr](#) A, size\_t lda)  
*Random Triangular Matrix.*
- size\_t [RandInt](#) (size\_t a, size\_t b)
- template<class [Field](#) , class Randlter >  
[Field::Element\\_ptr RandomSymmetricMatrix](#) (const [Field](#) &F, size\_t n, bool nonsingular, typename [Field::Element\\_ptr](#) A, size\_t lda, Randlter &G)  
*Random Symmetric Matrix.*
- template<class [Field](#) , class Randlter >  
[Field::Element\\_ptr RandomMatrixWithRank](#) (const [Field](#) &F, size\_t m, size\_t n, size\_t r, typename [Field::Element\\_ptr](#) A, size\_t lda, Randlter &G)  
*Random Matrix with prescribed rank.*
- template<class [Field](#) >  
[Field::Element\\_ptr RandomMatrixWithRank](#) (const [Field](#) &F, size\_t m, size\_t n, size\_t r, typename [Field::Element\\_ptr](#) A, size\_t lda)  
*Random Matrix with prescribed rank.*
- size\_t \* [RandomIndexSubset](#) (size\_t N, size\_t R, size\_t \*P)  
*Pick uniformly at random a sequence of  $R$  distinct elements from the set  $\{0, \dots, N - 1\}$  using Knuth's shuffle.*
- size\_t \* [RandomPermutation](#) (size\_t N, size\_t \*P)  
*Pick uniformly at random a permutation of size  $N$  stored in LAPACK format using Knuth's shuffle.*
- void [RandomRankProfileMatrix](#) (size\_t M, size\_t N, size\_t R, size\_t \*rows, size\_t \*cols)  
*Pick uniformly at random an  $R$ -subpermutation of dimension  $M \times N$  : a matrix with only  $R$  non-zeros equal to one, in a random rook placement.*
- void [swapval](#) (size\_t k, size\_t N, size\_t \*P, size\_t val)
- void [RandomSymmetricRankProfileMatrix](#) (size\_t N, size\_t R, size\_t \*rows, size\_t \*cols)  
*Pick uniformly at random a symmetric  $R$ -subpermutation of dimension  $N \times N$  : a symmetric matrix with only  $R$  non-zeros, all equal to one, in a random rook placement.*
- template<class [Field](#) , class Randlter >  
[Field::Element\\_ptr RandomMatrixWithRankandRPM](#) (const [Field](#) &F, size\_t M, size\_t N, size\_t R, typename [Field::Element\\_ptr](#) A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, Randlter &G)  
*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*

- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRankandRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)  
*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, Randlter &G)  
*Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .*
- `template<class Field >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)  
*Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, Randlter &G)  
*Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda)  
*Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, Randlter &G)  
*Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda)  
*Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithDet` (const `Field` &F, size\_t n, const typename `Field::Element` d, typename `Field::Element_ptr` A, size\_t lda)  
*Random Matrix with prescribed det.*
- `template<class Field , class Randlter >`  
`Field::Element_ptr RandomMatrixWithDet` (const `Field` &F, size\_t n, const typename `Field::Element` d, typename `Field::Element_ptr` A, size\_t lda, Randlter &G)  
*Random Matrix with prescribed det.*
- `template<typename Field >`  
`Givaro::Integer maxFieldElt` ()
- `template<> Givaro::Integer maxFieldElt< Givaro::ZRing< Givaro::Integer > > ()`
- `template<typename Field >`  
`Field * chooseField` (Givaro::Integer q, uint64\_t b, uint64\_t seed)
- `template<> Givaro::ZRing< int32_t > * chooseField< Givaro::ZRing< int32_t > >` (Givaro::Integer q, uint64\_t b, uint64\_t seed)
- `template<> Givaro::ZRing< int64_t > * chooseField< Givaro::ZRing< int64_t > >` (Givaro::Integer q, uint64\_t b, uint64\_t seed)
- `template<> Givaro::ZRing< float > * chooseField< Givaro::ZRing< float > >` (Givaro::Integer q, uint64\_t b, uint64\_t seed)
- `template<> Givaro::ZRing< double > * chooseField< Givaro::ZRing< double > >` (Givaro::Integer q, uint64\_t b, uint64\_t seed)

### 15.20.1 Detailed Description

Finite Field **PACK** Set of elimination based routines for dense linear algebra.

This namespace enlarges the set of BLAS routines of the class [FFLAS](#), with higher level routines based on elimination.

### 15.20.2 Typedef Documentation

#### 15.20.2.1 Checker\_PLUQ

```
using Checker_PLUQ = FFLAS::Checker_Empty<Field>
```

#### 15.20.2.2 Checker\_Det

```
using Checker_Det = FFLAS::Checker_Empty<Field>
```

#### 15.20.2.3 Checker\_invert

```
using Checker_invert = FFLAS::Checker_Empty<Field>
```

#### 15.20.2.4 Checker\_charpoly

```
using Checker_charpoly = FFLAS::Checker_Empty<Field>
```

#### 15.20.2.5 ForceCheck\_PLUQ

```
using ForceCheck_PLUQ = CheckerImplem_PLUQ<Field>
```

#### 15.20.2.6 ForceCheck\_Det

```
using ForceCheck_Det = CheckerImplem_Det<Field>
```

### 15.20.2.7 ForceCheck\_invert

```
using ForceCheck_invert = CheckerImplem_invert<Field>
```

### 15.20.2.8 ForceCheck\_charpoly

```
using ForceCheck_charpoly = CheckerImplem_charpoly<Field, Polynomial>
```

## 15.20.3 Function Documentation

### 15.20.3.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
    size_t * MathP,
    const size_t * LapackP,
    const size_t N ) [inline]
```

Conversion of a permutation from LAPACK format to Math format.

### 15.20.3.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
    size_t * LapackP,
    const size_t * MathP,
    const size_t N ) [inline]
```

Conversion of a permutation from Maths format to LAPACK format.

### 15.20.3.3 applyP() [1/4]

```
void applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P ) [inline]
```

Computes  $P1 \times \text{Diag}(I\_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.

## Parameters

|                |           |                                  |
|----------------|-----------|----------------------------------|
| <i>in, out</i> | <i>P1</i> | a LAPACK permutation of size N   |
|                | <i>P2</i> | a LAPACK permutation of size N-R |

Applies a permutation P to the matrix A. Apply a permutation P, stored in the LAPACK format (a sequence of transpositions) between indices *ibeg* and *iend* of P to (*iend-ibeg*) vectors of size M stored in A (as column for NoTrans and rows for Trans). Side==FFLAS::FflasLeft for row permutation Side==FFLAS::FflasRight for a column permutation Trans==FFLAS::FflasTrans for the inverse permutation of P

## Parameters

|              |                                                                                                 |
|--------------|-------------------------------------------------------------------------------------------------|
| <i>F</i>     | base field                                                                                      |
| <i>Side</i>  | decides if rows (FflasLeft) or columns (FflasRight) are permuted                                |
| <i>Trans</i> | decides if the matrix is seen as columns (FflasTrans) or rows (FflasNoTrans)                    |
| <i>M</i>     | size of the elements to permute                                                                 |
| <i>ibeg</i>  | first index to consider in P                                                                    |
| <i>iend</i>  | last index to consider in P                                                                     |
| <i>A</i>     | input matrix                                                                                    |
| <i>lda</i>   | leading dimension of A                                                                          |
| <i>P</i>     | permutation in LAPACK format                                                                    |
| <i>psh</i>   | (optional): a sequential or parallel helper, to choose between sequential or parallel execution |

## Warning

not sure the submatrix is still a permutation and the one we expect in all cases... examples for *iend*=2, *ibeg*=1 and *P*=[2,2,2]

## 15.20.3.4 applyP() [2/4]

```
void applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t m,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const FFLAS::ParSeqHelper::Sequential seq ) [inline]
```

### 15.20.3.5 applyP() [3/4]

```
void applyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t m,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par ) [inline]
```

### 15.20.3.6 MonotonicApplyP()

```
void MonotonicApplyP (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t R ) [inline]
```

Apply a R-monotonically increasing permutation P, to the matrix A.

MonotonicApplyP Apply a permutation defined by the first R entries of the vector P (the pivots).

The permutation represented by P is defined as follows:

- the first R values of P is a LAPACK representation (a sequence of transpositions)
- the remaining iend-ibeg-R values of the permutation are in a monotonically increasing progression Side==FFLAS::FflasLeft for row permutation Side==FFLAS::FflasRight for a column permutation Trans==FFLAS::FflasTrans for the inverse permutation of P

#### Parameters

|              |                                                                       |
|--------------|-----------------------------------------------------------------------|
| <i>F</i>     | base field                                                            |
| <i>Side</i>  | selects if it is a row (FflasLeft) or column (FflasRight) permutation |
| <i>Trans</i> | inverse permutation (FflasTrans/NoTrans)                              |
| <i>M</i>     |                                                                       |
| <i>ibeg</i>  |                                                                       |
| <i>iend</i>  |                                                                       |
| <i>A</i>     | input matrix                                                          |
| <i>lda</i>   | leading dimension of A                                                |
| <i>P</i>     | LAPACK permuation                                                     |
| <i>R</i>     | first values of P                                                     |



The non pivot elements, are located in montonically increasing order.

### 15.20.3.7 fgetrs() [1/4]

```
void fgetrs (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb,
    int * info )
```

Solve the system  $AX = B$  or  $XA = B$ .

Solving using the PLUQ decomposition of A already computed inplace with PLUQ (FFLAS::FflasNonUnit). Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

#### Parameters

|             |                                                                                    |
|-------------|------------------------------------------------------------------------------------|
| <i>F</i>    | base field                                                                         |
| <i>Side</i> | Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking. |
| <i>M</i>    | row dimension of B                                                                 |
| <i>N</i>    | col dimension of B                                                                 |
| <i>R</i>    | rank of A                                                                          |
| <i>A</i>    | input matrix                                                                       |
| <i>lda</i>  | leading dimension of A                                                             |
| <i>P</i>    | row permutation of the PLUQ decomposition of A                                     |
| <i>Q</i>    | column permutation of the PLUQ decomposition of A                                  |
| <i>B</i>    | Right/Left hand side matrix. Initially stores B, finally stores the solution X.    |
| <i>ldb</i>  | leading dimension of B                                                             |
| <i>info</i> | Success of the computation: 0 if successfull, >0 if system is inconsistent         |

### 15.20.3.8 fgetrs() [2/4]

```
Field::Element_ptr fgetrs (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
```

```

    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    typename Field::Element_ptr X,
    const size_t idx,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    int * info )

```

Solve the system  $A X = B$  or  $X A = B$ .

Solving using the PLUQ decomposition of A already computed inplace with PLUQ(FFLAS::FflasNonUnit). Version for A rectangular. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

#### Parameters

|             |                                                                                                             |
|-------------|-------------------------------------------------------------------------------------------------------------|
| <i>F</i>    | base field                                                                                                  |
| <i>Side</i> | Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.                          |
| <i>M</i>    | row dimension of A                                                                                          |
| <i>N</i>    | col dimension of A                                                                                          |
| <i>NRHS</i> | number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B |
| <i>R</i>    | rank of A                                                                                                   |
| <i>A</i>    | input matrix                                                                                                |
| <i>lda</i>  | leading dimension of A                                                                                      |
| <i>P</i>    | row permutation of the PLUQ decomposition of A                                                              |
| <i>Q</i>    | column permutation of the PLUQ decomposition of A                                                           |
| <i>X</i>    | solution matrix                                                                                             |
| <i>idx</i>  | leading dimension of X                                                                                      |
| <i>B</i>    | Right/Left hand side matrix.                                                                                |
| <i>ldb</i>  | leading dimension of B                                                                                      |
| <i>info</i> | Succes of the computation: 0 if successfull, >0 if system is inconsistent                                   |

#### 15.20.3.9 fgesv() [1/4]

```

size_t fgesv (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    int * info )

```

Square system solver.

## Parameters

|             |                                                                                     |
|-------------|-------------------------------------------------------------------------------------|
| <i>F</i>    | The computation domain                                                              |
| <i>Side</i> | Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking   |
| <i>M</i>    | row dimension of B                                                                  |
| <i>N</i>    | col dimension of B                                                                  |
| <i>A</i>    | input matrix                                                                        |
| <i>lda</i>  | leading dimension of A                                                              |
| <i>B</i>    | Right/Left hand side matrix. Initially contains B, finally contains the solution X. |
| <i>ldb</i>  | leading dimension of B                                                              |
| <i>info</i> | Success of the computation: 0 if successfull, >0 if system is inconsistent          |

## Returns

the rank of the system

Solve the system  $A X = B$  or  $X A = B$ . Version for A square. If A is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

## 15.20.3.10 fgesv() [2/4]

```
size_t fgesv (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    int * info )
```

Rectangular system solver.

## Parameters

|             |                                                                                                             |
|-------------|-------------------------------------------------------------------------------------------------------------|
| <i>F</i>    | The computation domain                                                                                      |
| <i>Side</i> | Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking                           |
| <i>M</i>    | row dimension of A                                                                                          |
| <i>N</i>    | col dimension of A                                                                                          |
| <i>NRHS</i> | number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B |
| <i>A</i>    | input matrix                                                                                                |
| <i>lda</i>  | leading dimension of A                                                                                      |
| <i>B</i>    | Right/Left hand side matrix. Initially contains B, finally contains the solution X.                         |
| <i>ldb</i>  | leading dimension of B                                                                                      |
| <i>X</i>    |                                                                                                             |
| <i>ldx</i>  |                                                                                                             |
| <i>info</i> | Success of the computation: 0 if successfull, >0 if system is inconsistent                                  |

**Returns**

the rank of the system

Solve the system  $A X = B$  or  $X A = B$ . Version for  $A$  square. If  $A$  is rank deficient, a solution is returned if the system is consistent, Otherwise an info is 1

**15.20.3.11 ftrtri() [1/2]**

```
void ftrtri (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t threshold = __FFLASFFPACK_FTRTRI_THRESHOLD )
```

Compute the inverse of a triangular matrix.

**Parameters**

|             |                                                        |
|-------------|--------------------------------------------------------|
| <i>F</i>    | base field                                             |
| <i>Uplo</i> | whether the matrix is upper or lower triangular        |
| <i>Diag</i> | whether the matrix is unit diagonal (FflasUnit/NoUnit) |
| <i>N</i>    | input matrix order                                     |
| <i>A</i>    | the input matrix                                       |
| <i>lda</i>  | leading dimension of A                                 |

**15.20.3.12 trinv\_left() [1/2]**

```
void trinv_left (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr L,
    const size_t ldl,
    typename Field::Element_ptr X,
    const size_t ldx )
```

**15.20.3.13 ftrtrm() [1/2]**

```
void ftrtrm (
    const Field & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_DIAG diag,
    const size_t N,
```

```

typename Field::Element_ptr A,
const size_t lda )

```

Compute the product of two triangular matrices of opposite shape.

Product UL or LU of the upper, resp lower triangular matrices U and L stored one above the other in the square matrix A.

#### Parameters

|             |                                                                      |
|-------------|----------------------------------------------------------------------|
| <i>F</i>    | base field                                                           |
| <i>Side</i> | set to FflasLeft to compute the product UL, FflasRight to compute LU |
| <i>diag</i> | whether the matrix U is unit diagonal (FflasUnit/NoUnit)             |
| <i>N</i>    | input matrix order                                                   |
| <i>A</i>    | the input matrix                                                     |
| <i>lda</i>  | leading dimension of A                                               |

#### 15.20.3.14 ftrstr()

```

void ftrstr (
    const Field & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diagA,
    const FFLAS::FFLAS_DIAG diagB,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const size_t threshold = __FFLASFFPACK_FTRSTR_THRESHOLD ) [inline]

```

Solve a triangular system with a triangular right hand side of the same shape.

#### Parameters

|              |                                                                                                                       |
|--------------|-----------------------------------------------------------------------------------------------------------------------|
| <i>F</i>     | base field                                                                                                            |
| <i>Side</i>  | set to FflasLeft to compute $U1^{-1} * U2$ or $L1^{-1} * L2$ , FflasRight to compute $U1 * U2^{-1}$ or $L1 * L2^{-1}$ |
| <i>Uplo</i>  | whether the matrix A is upper or lower triangular                                                                     |
| <i>diag1</i> | whether the matrix U1 or L2 is unit diagonal (FflasUnit/NoUnit)                                                       |
| <i>diag2</i> | whether the matrix U2 or L2 is unit diagonal (FflasUnit/NoUnit)                                                       |
| <i>N</i>     | order of the input matrices                                                                                           |
| <i>A</i>     | the input matrix to be inverted (U1 or L1)                                                                            |
| <i>lda</i>   | leading dimension of A                                                                                                |
| <i>B</i>     | the input right hand side (U2 or L2)                                                                                  |
| <i>ldb</i>   | leading dimension of B                                                                                                |

### 15.20.3.15 ftrssyr2k()

```
void ftrssyr2k (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diagA,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr B,
    const size_t ldb,
    const size_t threshold = __FFLASFFPACK_FTRSSYR2K_THRESHOLD ) [inline]
```

Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.

C is overwritten by B.

#### Parameters

|         |              |                                                                                                                       |
|---------|--------------|-----------------------------------------------------------------------------------------------------------------------|
|         | <i>F</i>     | base field                                                                                                            |
|         | <i>Side</i>  | set to FflasLeft to compute $U1^{-1} * U2$ or $L1^{-1} * L2$ , FflasRight to compute $U1 * U2^{-1}$ or $L1 * L2^{-1}$ |
|         | <i>Uplo</i>  | whether the matrix A is upper or lower triangular                                                                     |
|         | <i>diagA</i> | whether the matrix A is unit diagonal (FflasUnit/NoUnit)                                                              |
|         | <i>N</i>     | order of the input matrices                                                                                           |
|         | <i>A</i>     | the input matrix                                                                                                      |
|         | <i>lda</i>   | leading dimension of A                                                                                                |
| in, out | <i>B</i>     | the input right hand side where the output is written                                                                 |
|         | <i>ldb</i>   | leading dimension of B                                                                                                |

### 15.20.3.16 fsytrf() [1/3]

```
bool fsytrf (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )
```

Triangular factorization of symmetric matrices.

#### Parameters

|         |             |                                                                                          |
|---------|-------------|------------------------------------------------------------------------------------------|
|         | <i>F</i>    | The computation domain                                                                   |
|         | <i>UpLo</i> | Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor |
|         | <i>N</i>    | order of the matrix A                                                                    |
| in, out | <i>A</i>    | input matrix                                                                             |
|         | <i>lda</i>  | leading dimension of A                                                                   |

## Returns

false if the  $A$  does not have generic rank profile, making the computation fail.

Compute the a triangular factorization of the matrix  $A$ :  $A = L \times D \times L^T$  if UpLo = FflasLower or  $A = U^T \times D \times U$  otherwise.  $D$  is a diagonal matrix. The matrices  $L$  and  $U$  are unit diagonal lower (resp. upper) triangular and overwrite the input matrix  $A$ . The matrix  $D$  is stored on the diagonal of  $A$ , as the diagonal of  $L$  or  $U$  is known to be all ones. If  $A$  does not have generic rank profile, the LDLT or UTDU factorizations is not defined, and the algorithm returns false.

## 15.20.3.17 fsytrf() [2/3]

```
bool fsytrf (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Sequential seq,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD ) [inline]
```

## 15.20.3.18 fsytrf() [3/3]

```
bool fsytrf (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD ) [inline]
```

## 15.20.3.19 fsytrf\_nonunit() [1/3]

```
bool fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr D,
    const size_t incD,
    const size_t threshold = __FFLASFFPACK_FSYTRF_THRESHOLD )
```

Triangular factorization of symmetric matrices.

## Parameters

|                      |        |                                                                                          |
|----------------------|--------|------------------------------------------------------------------------------------------|
|                      | $F$    | The computation domain                                                                   |
|                      | $UpLo$ | Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor |
| Generated by Doxygen | $N$    | order of the matrix $A$                                                                  |
| in, out              | $A$    | input matrix                                                                             |
| in, out              | $D$    |                                                                                          |
|                      | $lda$  | leading dimension of $A$                                                                 |

## Returns

false if the  $A$  does not have generic rank profile, making the computation fail.

Compute the a triangular factorization of the matrix  $A$ :  $A = L \times D_{inv} \times L^T$  if UpLo = FflasLower or  $A = U^T \times D \times U$  otherwise.  $D$  is a diagonal matrix. The matrices  $L$  and  $U$  are lower (resp. upper) triangular and overwrite the input matrix  $A$ . The matrix  $D$  need to be stored separately, as the diagonal of  $L$  or  $U$  are not unit. If  $A$  does not have generic rank profile, the LDLT or UTDU factorizations is not defined, and the algorithm returns false.

## 15.20.3.20 PLUQ() [1/6]

```
size_t PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

Compute a PLUQ factorization of the given matrix.

Return its rank. The permutations P and Q are represented using LAPACK's convention.

## Parameters

|        |                                                                        |
|--------|------------------------------------------------------------------------|
| $F$    | base field                                                             |
| $Diag$ | whether U should have a unit diagonal (FflasUnit) or not (FflasNoUnit) |
| $M$    | matrix row dimension                                                   |
| $N$    | matrix column dimension                                                |
| $A$    | input matrix                                                           |
| $lda$  | leading dimension of $A$                                               |
| $P$    | the row permutation                                                    |
| $Q$    | the column permutation                                                 |

## Returns

the rank of  $A$

## Bibliography

- Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13, 2013

## 15.20.3.21 pPLUQ()

```
size_t pPLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
```



```

    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]

```

### 15.20.3.22 PLUQ() [2/6]

```

size_t PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Sequential & PSHelper,
    size_t BCThreshold = __FFLASFFPACK_PLUQ_THRESHOLD ) [inline]

```

### 15.20.3.23 PLUQ() [3/6]

```

size_t PLUQ (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > & PSHelper )

```

### 15.20.3.24 LUdivine() [1/4]

```

size_t LUdivine (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive,
    const size_t cutoff = __FFLASFFPACK_LUDIVINE_THRESHOLD )

```

Compute the CUP or PLE factorization of the given matrix.

Using a block algorithm and return its rank. The permutations P and Q are represented using LAPACK's convention.

## Parameters

|               |                                                                                                                             |
|---------------|-----------------------------------------------------------------------------------------------------------------------------|
| <i>F</i>      | base field                                                                                                                  |
| <i>Diag</i>   | whether the transformation matrix (U of the CUP, L of the PLE) should have a unit diagonal (FflasUnit) or not (FflasNoUnit) |
| <i>trans</i>  | whether to compute the CUP decomposition (FflasNoTrans) or the PLE decomposition (FflasTrans)                               |
| <i>M</i>      | matrix row dimension                                                                                                        |
| <i>N</i>      | matrix column dimension                                                                                                     |
| <i>A</i>      | input matrix                                                                                                                |
| <i>lda</i>    | leading dimension of A                                                                                                      |
| <i>P</i>      | the factor of CUP or PLE                                                                                                    |
| <i>Q</i>      | a permutation indicating the pivot position in the echelon form C or E in its first r positions                             |
| <i>LuTag</i>  | flag for setting the earling termination if the matrix is singular                                                          |
| <i>cutoff</i> | threshold to basecase                                                                                                       |

## Returns

the rank of A

## Bibliography

- Jeannerod C-P, Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013
- Pernet C, Brassel M *LUdivine, une divine factorisation LU*, 2002

## 15.20.3.25 ColumnEchelonForm() [1/3]

```
size_t ColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Compute the Column Echelon form of the input matrix in-place.

If LuTag == FfpackTileRecursive, then after the computation  $A = [M \setminus V]$  such that  $AU = C$  is a column echelon decomposition of A, with  $U = P^T [V]$  and  $C = M + Q [I_r] [0 \text{ } I_{n-r}] [0]$  If LuTag == FfpackTileRecursive then  $A = [N \setminus V]$  such that the same holds with  $M = QN$

$Q^t = Q^T$  If transform=false, the matrix V is not computed. See also test-colechelon for an example of use

## Parameters

|    |                  |                                                    |
|----|------------------|----------------------------------------------------|
|    | <i>F</i>         | base field                                         |
|    | <i>M</i>         | number of rows                                     |
|    | <i>N</i>         | number of columns                                  |
| in | <i>A</i>         | input matrix                                       |
|    | <i>lda</i>       | leading dimension of A                             |
|    | <i>P</i>         | the column permutation                             |
|    | <i>Qt</i>        | the row position of the pivots in the echelon form |
|    | <i>transform</i> | decides whether V is computed                      |

**15.20.3.26 pColumnEchelonForm()**

```

size_t pColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

**15.20.3.27 ColumnEchelonForm() [2/3]**

```

size_t ColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]

```

**15.20.3.28 RowEchelonForm() [1/3]**

```

size_t RowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Compute the Row Echelon form of the input matrix in-place.

If  $\text{LuTag} == \text{FfpackTileRecursive}$ , then after the computation  $A = [L \setminus M]$  such that  $XA = R$  is a row echelon decomposition of  $A$ , with  $X = [L \ 0]P$  and  $R = M + [I \ 0]Q^T$ . If  $\text{LuTag} == \text{FfpackSlabRecursive}$  then  $A = [L \setminus N]$  such that the same holds with  $M = NQ^TQt = Q^T$ . If  $\text{transform} = \text{false}$ , the matrix  $L$  is not computed. See also `test-rowechelon` for an example of use

## Parameters

|    |                  |                                                                                       |
|----|------------------|---------------------------------------------------------------------------------------|
|    | $F$              | base field                                                                            |
|    | $M$              | number of rows                                                                        |
|    | $N$              | number of columns                                                                     |
| in | $A$              | the input matrix                                                                      |
|    | $lda$            | leading dimension of A                                                                |
|    | $P$              | the row permutation                                                                   |
|    | $Qt$             | the column position of the pivots in the echelon form                                 |
|    | <i>transform</i> | decides whether L is computed                                                         |
|    | <i>LuTag</i>     | chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ |

## 15.20.3.29 pRowEchelonForm()

```

size_t pRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

## 15.20.3.30 RowEchelonForm() [2/3]

```

size_t RowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psh ) [inline]

```

**15.20.3.31 ReducedColumnEchelonForm()** [1/3]

```

size_t ReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Compute the Reduced Column Echelon form of the input matrix in-place.

After the computation  $A = [V]$  such that  $AX = R$  is a reduced col echelon  $[M \ 0]$  decomposition of  $A$ , where  $X = P^T [V]$  and  $R = Q [I_r] [0 \ I_{n-r}] [M \ 0] Q^T = Q^T I$  If transform=false, the matrix  $X$  is not computed and the matrix  $A = R$

**Parameters**

|    |             |                                                                                       |
|----|-------------|---------------------------------------------------------------------------------------|
|    | $F$         | base field                                                                            |
|    | $M$         | number of rows                                                                        |
|    | $N$         | number of columns                                                                     |
| in | $A$         | input matrix                                                                          |
|    | $lda$       | leading dimension of $A$                                                              |
|    | $P$         | the column permutation                                                                |
|    | $Qt$        | the row position of the pivots in the echelon form                                    |
|    | $transform$ | decides whether $X$ is computed                                                       |
|    | $LuTag$     | chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ |

**15.20.3.32 pReducedColumnEchelonForm()**

```

size_t pReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

**15.20.3.33 ReducedColumnEchelonForm()** [2/3]

```

size_t ReducedColumnEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & pSH ) [inline]

```

**15.20.3.34 ReducedRowEchelonForm()** [1/3]

```

size_t ReducedRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Compute the Reduced Row Echelon form of the input matrix in-place.

After the computation  $A = [V1 \ M]$  such that  $X A = R$  is a reduced row echelon  $[V2 \ 0]$  decomposition of  $A$ , where  $X = [V1 \ 0] P$  and  $R = [I_r \ M] Q^T [V2 \ In-r] [0] Qt = Q^T$  If transform=false, the matrix  $X$  is not computed and the matrix  $A = R$

**Parameters**

|    |             |                                                                                       |
|----|-------------|---------------------------------------------------------------------------------------|
|    | $F$         | base field                                                                            |
|    | $M$         | number of rows                                                                        |
|    | $N$         | number of columns                                                                     |
| in | $A$         | input matrix                                                                          |
|    | $lda$       | leading dimension of $A$                                                              |
|    | $P$         | the row permutation                                                                   |
|    | $Qt$        | the column position of the pivots in the echelon form                                 |
|    | $transform$ | decides whether $X$ is computed                                                       |
|    | $LuTag$     | chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ |

**15.20.3.35 pReducedRowEchelonForm()**

```

size_t pReducedRowEchelonForm (
    const Field & F,

```

```

const size_t M,
const size_t N,
typename Field::Element_ptr A,
const size_t lda,
size_t * P,
size_t * Qt,
const bool transform = false,
size_t numthreads = 0,
const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]

```

### 15.20.3.36 ReducedRowEchelonForm() [2/3]

```

size_t ReducedRowEchelonForm (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag,
    const PSHelper & psH ) [inline]

```

### 15.20.3.37 Invert() [1/4]

```

Field::Element_ptr Invert (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    int & nullity )

```

Invert the given matrix in place or computes its nullity if it is singular.

An inplace  $2n^3$  algorithm is used.

#### Parameters

|         |           |                               |
|---------|-----------|-------------------------------|
|         | $F$       | The computation domain        |
|         | $M$       | order of the matrix           |
| in, out | $A$       | input matrix ( $M \times M$ ) |
|         | $lda$     | leading dimension of A        |
|         | $nullity$ | dimension of the kernel of A  |

#### Returns

pointer to  $A$  and  $A \leftarrow A^{-1}$

**15.20.3.38 Invert()** [2/4]

```
Field::Element_ptr Invert (
    const Field & F,
    const size_t M,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldX,
    int & nullity )
```

Invert the given matrix or computes its nullity if it is singular.

**Precondition**

X is preallocated and should be large enough to store the  $m \times m$  matrix A.

**Parameters**

|     |           |                                                                                                    |
|-----|-----------|----------------------------------------------------------------------------------------------------|
|     | $F$       | The computation domain                                                                             |
|     | $M$       | order of the matrix                                                                                |
| in  | $A$       | input matrix ( $M \times M$ )                                                                      |
|     | $lda$     | leading dimension of A                                                                             |
| out | $X$       | this is the inverse of A if A is invertible (non NULL and nullity = 0). It is untouched otherwise. |
|     | $ldx$     | leading dimension of X                                                                             |
|     | $nullity$ | dimension of the kernel of A                                                                       |

**Returns**

pointer to  $X = A^{-1}$

**15.20.3.39 Invert2()** [1/2]

```
Field::Element_ptr Invert2 (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldX,
    int & nullity )
```

Invert the given matrix or computes its nullity if it is singular.

An  $2n^3f$  algorithm is used. This routine can be % faster than [FFPACK::Invert](#) but is not totally inplace.



**Precondition**

$X$  is preallocated and should be large enough to store the  $m \times m$  matrix  $A$ .

**Warning**

$A$  is overwritten here !

**Bug** not tested.

**Parameters**

|         |           |                                                                                                             |
|---------|-----------|-------------------------------------------------------------------------------------------------------------|
|         | $F$       | the computation domain                                                                                      |
|         | $M$       | order of the matrix                                                                                         |
| in, out | $A$       | input matrix ( $M \times M$ ). On output, $A$ is modified and represents a "psycological" factorisation LU. |
|         | $lda$     | leading dimension of $A$                                                                                    |
| out     | $X$       | this is the inverse of $A$ if $A$ is invertible (non NULL and nullity = 0). It is untouched otherwise.      |
|         | $ldx$     | leading dimension of $X$                                                                                    |
|         | $nullity$ | dimension of the kernel of $A$                                                                              |

**Returns**

pointer to  $X = A^{-1}$

**Todo** this init is not all necessary (done after ftrtri)

**Todo** this init is not all necessary (done after ftrtri)

**15.20.3.40 CharPoly() [1/8]**

```
std::list< typename PolRing::Element > & CharPoly (
    const PolRing & R,
    std::list< typename PolRing::Element > & charp,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    typename PolRing::Domain_t::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]
```

Compute the characteristic polynomial of the matrix  $A$ .

**Parameters**

|                            |            |                                                                                       |
|----------------------------|------------|---------------------------------------------------------------------------------------|
|                            | $R$        | the polynomial ring of charp (contains the base field)                                |
| out                        | $charp$    | the characteristic polynomial of $A$ as a list of factors                             |
|                            | $N$        | order of the matrix $A$                                                               |
| Generated by Doxygen<br>in | $A$        | the input matrix ( $N \times N$ ) (could be overwritten in some algorithmic variants) |
|                            | $lda$      | leading dimension of $A$                                                              |
|                            | $CharpTag$ | the algorithmic variant                                                               |
|                            | $G$        | a random iterator (required for the randomized variants LUKernel and ArithProg)       |

**15.20.3.41 CharPoly() [2/8]**

```

PolRing::Element & CharPoly (
    const PolRing & R,
    typename PolRing::Element & charp,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    typename PolRing::Domain_t::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]

```

Compute the characteristic polynomial of the matrix A.

**Parameters**

|     |                 |                                                                                       |
|-----|-----------------|---------------------------------------------------------------------------------------|
|     | <i>R</i>        | the polynomial ring of charp (contains the base field)                                |
| out | <i>charp</i>    | the characteristic polynomial of as a single polynomial                               |
|     | <i>N</i>        | order of the matrix A                                                                 |
| in  | <i>A</i>        | the input matrix ( $N \times N$ ) (could be overwritten in some algorithmic variants) |
|     | <i>lda</i>      | leading dimension of A                                                                |
|     | <i>CharpTag</i> | the algorithmic variant                                                               |
|     | <i>G</i>        | a random iterator (required for the randomized variants LUKrylov and ArithProg)       |

**15.20.3.42 CharPoly() [3/8]**

```

PolRing::Element & CharPoly (
    const PolRing & R,
    typename PolRing::Element & charp,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    const FFPACK_CHARPOLY_TAG CharpTag = FfpackAuto,
    const size_t degree = __FFLASFFPACK_ARITHPROG_THRESHOLD ) [inline]

```

Compute the characteristic polynomial of the matrix A.

**Parameters**

|     |                 |                                                                                       |
|-----|-----------------|---------------------------------------------------------------------------------------|
|     | <i>R</i>        | the polynomial ring of charp (contains the base field)                                |
| out | <i>charp</i>    | the characteristic polynomial of as a single polynomial                               |
|     | <i>N</i>        | order of the matrix A                                                                 |
| in  | <i>A</i>        | the input matrix ( $N \times N$ ) (could be overwritten in some algorithmic variants) |
|     | <i>lda</i>      | leading dimension of A                                                                |
|     | <i>CharpTag</i> | the algorithmic variant                                                               |

**15.20.3.43 MinPoly()** [1/4]

```
Polynomial & MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda ) [inline]
```

Compute the minimal polynomial of the matrix A.

The algorithm is randomized probabilistic, and computes the minimal polynomial of the Krylov iterates of a random vector:  $(v, Av, \dots, A^k v)$

**Parameters**

|     |             |                                   |
|-----|-------------|-----------------------------------|
|     | <i>F</i>    | the base field                    |
| out | <i>minP</i> | the minimal polynomial of A       |
|     | <i>N</i>    | order of the matrix A             |
| in  | <i>A</i>    | the input matrix ( $N \times N$ ) |
|     | <i>lda</i>  | leading dimension of A            |

**15.20.3.44 MinPoly()** [2/4]

```
Polynomial & MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    RandIter & G ) [inline]
```

Compute the minimal polynomial of the matrix A.

The algorithm is randomized probabilistic, and computes the minimal polynomial of the Krylov iterates of a random vector:  $(v, Av, \dots, A^k v)$

**Parameters**

|     |             |                                   |
|-----|-------------|-----------------------------------|
|     | <i>F</i>    | the base field                    |
| out | <i>minP</i> | the minimal polynomial of A       |
|     | <i>N</i>    | order of the matrix A             |
| in  | <i>A</i>    | the input matrix ( $N \times N$ ) |
|     | <i>lda</i>  | leading dimension of A            |
|     | <i>G</i>    | a random iterator                 |

**15.20.3.45 MatVecMinPoly()** [1/2]

```

Polynomial & MatVecMinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr v,
    const size_t incv ) [inline]

```

Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .

**Parameters**

|     |        |                                                                                 |
|-----|--------|---------------------------------------------------------------------------------|
|     | $F$    | the base field                                                                  |
| out | $minP$ | the minimal polynomial of A and v                                               |
|     | $N$    | order of the matrix A                                                           |
| in  | $A$    | the input matrix ( $N \times N$ )                                               |
|     | $lda$  | leading dimension of A                                                          |
|     | $K$    | an $N \times (N + 1)$ matrix containing the vector v on its first row           |
|     | $ldk$  | leading dimension of K                                                          |
|     | $P$    | [out] (optional) the permutation used in the elimination of the Krylov matrix K |

**15.20.3.46 Rank()** [1/3]

```

size_t Rank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )

```

Computes the rank of the given matrix using a PLUQ factorization.

The input matrix is modified.

**Parameters**

|    |       |                                                                               |
|----|-------|-------------------------------------------------------------------------------|
|    | $F$   | base field                                                                    |
|    | $M$   | row dimension of the matrix                                                   |
|    | $N$   | column dimension of the matrix                                                |
| in | $A$   | input matrix                                                                  |
|    | $lda$ | leading dimension of A                                                        |
|    | $psH$ | (optional) a ParSeqHelper to choose between sequential and parallel execution |

**15.20.3.47 pRank()**

```

size_t pRank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t numthreads = 0 )

```

**15.20.3.48 Rank()** [2/3]

```

size_t Rank (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const PSHelper & pSH )

```

**15.20.3.49 IsSingular()** [1/2]

```

bool IsSingular (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )

```

Returns true if the given matrix is singular.

The method is a block elimination with early termination

using LQUP factorization with early termination. If  $M \neq N$ , then the matrix is virtually padded with zeros to make it square and it's determinant is zero.

**Warning**

The input matrix is modified.

**Parameters**

|         |       |                                 |
|---------|-------|---------------------------------|
|         | $F$   | base field                      |
|         | $M$   | row dimension of the matrix     |
|         | $N$   | column dimension of the matrix. |
| in, out | $A$   | input matrix                    |
|         | $lda$ | leading dimension of $A$        |

### 15.20.3.50 Det() [1/6]

```
Field::Element & Det (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P = NULL,
    size_t * Q = NULL ) [inline]
```

Returns the determinant of the given square matrix.

The method is a block elimination using PLUQ factorization. The input matrix A is overwritten.

#### Warning

The input matrix is modified.

#### Parameters

|         |            |                                                                                                                                                             |
|---------|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
|         | <i>F</i>   | base field                                                                                                                                                  |
| out     | <i>det</i> | the determinant of A                                                                                                                                        |
|         | <i>N</i>   | the order of the square matrix A.                                                                                                                           |
| in, out | <i>A</i>   | input matrix                                                                                                                                                |
|         | <i>lda</i> | leading dimension of A                                                                                                                                      |
|         | <i>psH</i> | (optional) a ParSeqHelper to choose between sequential and parallel execution                                                                               |
|         | <i>P,Q</i> | (optional) row and column permutations to be used by the PLUQ factorization. randomized checkers (see cherckes/checker_det.inl) need them for certification |

### 15.20.3.51 pDet()

```
Field::Element & pDet (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t numthreads = 0,
    size_t * P = NULL,
    size_t * Q = NULL ) [inline]
```

**15.20.3.52 Det()** [2/6]

```
Field::Element & Det (
    const Field & F,
    typename Field::Element & det,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const PSHelper & pSH,
    size_t * P = NULL,
    size_t * Q = NULL )
```

**15.20.3.53 Solve()** [1/3]

```
Field::Element_ptr Solve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr x,
    const int incx,
    typename Field::ConstElement_ptr b,
    const int incb ) [inline]
```

Solves a linear system  $AX = b$  using PLUQ factorization.

@oaram F base field @oaram M matrix order

**Parameters**

|     |             |                                     |
|-----|-------------|-------------------------------------|
| in  | <i>A</i>    | input matrix                        |
|     | <i>lda</i>  | leading dimension of A              |
| out | <i>x</i>    | output solution vector              |
|     | <i>incx</i> | increment of x                      |
|     | <i>b</i>    | input right hand side of the system |
|     | <i>incb</i> | increment of b                      |

**15.20.3.54 Solve()** [2/3]

```
Field::Element_ptr Solve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr x,
    const int incx,
    typename Field::ConstElement_ptr b,
```

```
const int incb,
PSHelper & psH )
```

### 15.20.3.55 pSolve()

```
Field::Element_ptr pSolve (
    const Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr x,
    const int incx,
    typename Field::ConstElement_ptr b,
    const int incb,
    size_t numthreads = 0 ) [inline]
```

### 15.20.3.56 RandomNullSpaceVector() [1/3]

```
*void RandomNullSpaceVector (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t incX )
```

Solve  $LX = B$  or  $XL = B$  in place.

$L$  is  $M \times M$  if `Side == FFLAS::FflasLeft` and  $N \times N$  if `Side == FFLAS::FflasRight`,  $B$  is  $M \times N$ . Only the  $R$  non trivial column of  $L$  are stored in the  $M \times R$  matrix  $L$  Requirement : so that  $L$  could be expanded in-place Computes a vector of the Left/Right nullspace of the matrix  $A$ .

#### Parameters

|         |             |                                                                                  |
|---------|-------------|----------------------------------------------------------------------------------|
|         | <i>F</i>    | The computation domain                                                           |
|         | <i>Side</i> | decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace |
|         | <i>M</i>    | number of rows                                                                   |
|         | <i>N</i>    | number of columns                                                                |
| in, out | <i>A</i>    | input matrix of dimension $M \times N$ , $A$ is modified to its LU version       |
|         | <i>lda</i>  | leading dimension of $A$                                                         |
| out     | <i>X</i>    | output vector                                                                    |
|         | <i>incX</i> | increment of $X$                                                                 |



**15.20.3.57 NullSpaceBasis()** [1/2]

```

size_t NullSpaceBasis (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & NS,
    size_t & ldn,
    size_t & NSdim )

```

Computes a basis of the Left/Right nullspace of the matrix A.

return the dimension of the nullspace.

**Parameters**

|         |              |                                                                                  |
|---------|--------------|----------------------------------------------------------------------------------|
|         | <i>F</i>     | The computation domain                                                           |
|         | <i>Side</i>  | decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace |
|         | <i>M</i>     | number of rows                                                                   |
|         | <i>N</i>     | number of columns                                                                |
| in, out | <i>A</i>     | input matrix of dimension M x N, A is modified                                   |
|         | <i>lda</i>   | leading dimension of A                                                           |
| out     | <i>NS</i>    | output matrix of dimension N x NSdim (allocated here)                            |
| out     | <i>ldn</i>   | leading dimension of NS                                                          |
| out     | <i>NSdim</i> | the dimension of the Nullspace (N-rank(A))                                       |

**15.20.3.58 RowRankProfile()** [1/3]

```

size_t RowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Computes the row rank profile of A.

**Parameters**

|     |                  |                                                                                       |
|-----|------------------|---------------------------------------------------------------------------------------|
|     | <i>F</i>         | base field                                                                            |
|     | <i>M</i>         | number of rows                                                                        |
|     | <i>N</i>         | number of columns                                                                     |
| in  | <i>A</i>         | input matrix of dimension M x N                                                       |
|     | <i>lda</i>       | leading dimension of A                                                                |
| out | <i>rkprofile</i> | return the rank profile as an array of row indexes, of dimension r=rank(A)            |
|     | <i>LuTag</i>     | chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ |

A modified rkprofile is allocated during the computation.

#### Returns

R

#### 15.20.3.59 pRowRankProfile()

```
size_t pRowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]
```

#### 15.20.3.60 RowRankProfile() [2/3]

```
size_t RowRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag,
    PSHelper & psH ) [inline]
```

#### 15.20.3.61 ColumnRankProfile() [1/3]

```
size_t ColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Computes the column rank profile of A.

#### Parameters

|     |             |                                                                                     |                      |
|-----|-------------|-------------------------------------------------------------------------------------|----------------------|
|     | $F$         | base field                                                                          | Generated by Doxygen |
|     | $M$         | number of rows                                                                      |                      |
|     | $N$         | number of columns                                                                   |                      |
| in  | $A$         | input matrix of dimension                                                           |                      |
|     | $lda$       | leading dimension of A                                                              |                      |
| out | $rkprofile$ | return the rank profile as an array of row indexes. of dimension $r=\text{rank}(A)$ |                      |

A is modified rkprofile is allocated during the computation.

#### Returns

R

#### 15.20.3.62 pColumnRankProfile()

```
size_t pColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *rkprofile,
    size_t numthreads = 0,
    const FFPACK_LU_TAG LuTag = FfpackTileRecursive ) [inline]
```

#### 15.20.3.63 ColumnRankProfile() [2/3]

```
size_t ColumnRankProfile (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *rkprofile,
    const FFPACK_LU_TAG LuTag,
    PSHelper & psH ) [inline]
```

#### 15.20.3.64 RankProfileFromLU()

```
void RankProfileFromLU (
    const size_t * P,
    const size_t N,
    const size_t R,
    size_t * rkprofile,
    const FFPACK_LU_TAG LuTag ) [inline]
```

Recovers the column/row rank profile from the permutation of an LU decomposition.

Works with both the CUP/PLE decompositions (obtained by LUdivine) or the PLUQ decomposition. Assumes that the output vector containing the rank profile is already allocated.

## Parameters

|     |             |                                                                                       |
|-----|-------------|---------------------------------------------------------------------------------------|
|     | $P$         | the permutation carrying the rank profile information                                 |
|     | $N$         | the row/col dimension for a row/column rank profile                                   |
|     | $R$         | the rank of the matrix                                                                |
| out | $rkprofile$ | return the rank profile as an array of indices                                        |
|     | $LuTag$     | chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ |

## 15.20.3.65 LeadingSubmatrixRankProfiles()

```
size_t LeadingSubmatrixRankProfiles (
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t LSm,
    const size_t LSn,
    const size_t * P,
    const size_t * Q,
    size_t * RRP,
    size_t * CRP ) [inline]
```

Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.

Only works with the PLUQ decomposition Assumes that the output vectors containing the rank profiles are already allocated.

## Parameters

|       |                                                          |
|-------|----------------------------------------------------------|
| $P$   | the permutation carrying the rank profile information    |
| $M$   | the row dimension of the initial matrix                  |
| $N$   | the column dimension of the initial matrix               |
| $R$   | the rank of the initial matrix                           |
| $LSm$ | the row dimension of the leading submatrix considered    |
| $LSn$ | the column dimension of the leading submatrix considered |
| $P$   | the row permutation of the PLUQ decomposition            |
| $Q$   | the column permutation of the PLUQ decomposition         |
| $RRP$ | return the row rank profile of the leading submatrix     |

## Returns

the rank of the  $LSm \times LSn$  leading submatrix

A is modified

## Bibliography

- Dumas J-G., Pernet C., and Sultan Z. *Simultaneous computation of the row and column rank profiles*, ISSAC'13.

**15.20.3.66 RowRankProfileSubmatrixIndices()** [1/2]

```

size_t RowRankProfileSubmatrixIndices (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )

```

RowRankProfileSubmatrixIndices.

Computes the indices of the submatrix  $r \times r$  X of A whose rows correspond to the row rank profile of A.

**Parameters**

|     |                   |                                    |
|-----|-------------------|------------------------------------|
|     | $F$               | base field                         |
|     | $M$               | number of rows                     |
|     | $N$               | number of columns                  |
| in  | $A$               | input matrix of dimension          |
|     | <i>rowindices</i> | array of the row indices of X in A |
|     | <i>colindices</i> | array of the col indices of X in A |
|     | <i>lda</i>        | leading dimension of A             |
| out | $R$               | list of indices                    |

rowindices and colindices are allocated during the computation. A is modified

**Returns**

$R$

**15.20.3.67 ColRankProfileSubmatrixIndices()** [1/2]

```

size_t ColRankProfileSubmatrixIndices (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )

```

Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.

**Parameters**

|  |     |                |
|--|-----|----------------|
|  | $F$ | base field     |
|  | $M$ | number of rows |

## Parameters

|     |                   |                                    |
|-----|-------------------|------------------------------------|
|     | $N$               | number of columns                  |
| in  | $A$               | input matrix of dimension          |
|     | <i>rowindices</i> | array of the row indices of X in A |
|     | <i>colindices</i> | array of the col indices of X in A |
|     | <i>lda</i>        | leading dimension of A             |
| out | $R$               | list of indices                    |

rowindices and colindices are allocated during the computation.

## Warning

A is modified

## Returns

R

## 15.20.3.68 RowRankProfileSubmatrix() [1/2]

```
size_t RowRankProfileSubmatrix (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & X,
    size_t & R )
```

Computes the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.

## Parameters

|     |            |                                 |
|-----|------------|---------------------------------|
|     | $F$        | base field                      |
|     | $M$        | number of rows                  |
|     | $N$        | number of columns               |
| in  | $A$        | input matrix of dimension M x N |
|     | <i>lda</i> | leading dimension of A          |
| out | $X$        | the output matrix               |
| out | $R$        | list of indices                 |

A is not modified X is allocated during the computation.

## Returns

R

**15.20.3.69 ColRankProfileSubmatrix()** [1/2]

```
size_t ColRankProfileSubmatrix (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr & X,
    size_t & R )
```

Compute the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.

**Parameters**

|     |       |                                 |
|-----|-------|---------------------------------|
|     | $F$   | base field                      |
|     | $M$   | number of rows                  |
|     | $N$   | number of columns               |
| in  | $A$   | input matrix of dimension M x N |
|     | $lda$ | leading dimension of A          |
| out | $X$   | the output matrix               |
| out | $R$   | list of indices                 |

A is not modified X is allocated during the computation.

**Returns**

R

**15.20.3.70 getTriangular()** [1/2]

```
void getTriangular (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const bool OnlyNonZeroVectors = false ) [inline]
```

Extracts a triangular matrix from a compact storage A=L\U of rank R.

if OnlyNonZeroVectors is false, then T and A have the same dimensions Otherwise, T is R x N if UpLo = FflasUpper, else T is M x R

## Parameters

|     |                           |                                                                                       |
|-----|---------------------------|---------------------------------------------------------------------------------------|
|     | <i>F</i>                  | base field                                                                            |
|     | <i>UpLo</i>               | selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned |
|     | <i>diag</i>               | selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)                     |
|     | <i>M</i>                  | row dimension of T                                                                    |
|     | <i>N</i>                  | column dimension of T                                                                 |
|     | <i>R</i>                  | rank of the triangular matrix (how many rows/columns need to be copied)               |
| in  | <i>A</i>                  | input matrix                                                                          |
|     | <i>lda</i>                | leading dimension of A                                                                |
| out | <i>T</i>                  | output matrix                                                                         |
|     | <i>ldt</i>                | leading dimension of T                                                                |
|     | <i>OnlyNonZeroVectors</i> | decides whether the last zero rows/columns should be ignored                          |

**Todo** just one triangular fzero+fassign ?

**Todo** just one triangular fzero+fassign ?

### 15.20.3.71 getTriangular() [2/2]

```
void getTriangular (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr A,
    const size_t lda ) [inline]
```

Cleans up a compact storage  $A=LU$  to reveal a triangular matrix of rank  $R$ .

## Parameters

|         |             |                                                                                       |
|---------|-------------|---------------------------------------------------------------------------------------|
|         | <i>F</i>    | base field                                                                            |
|         | <i>UpLo</i> | selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is revealed |
|         | <i>diag</i> | selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)                     |
|         | <i>M</i>    | row dimension of A                                                                    |
|         | <i>N</i>    | column dimension of A                                                                 |
|         | <i>R</i>    | rank of the triangular matrix                                                         |
| in, out | <i>A</i>    | input/output matrix                                                                   |
|         | <i>lda</i>  | leading dimension of A                                                                |

**Todo** just one triangular fzero+fassign ?



**Todo** just one triangular fzero+fassign ?

### 15.20.3.72 getEchelonForm() [1/2]

```
void getEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const bool OnlyNonZeroVectors = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]
```

Extracts a matrix in echelon form from a compact storage A=LU of rank R obtained by RowEchelonForm or ColumnEchelonForm.

Either L or U is in Echelon form (depending on Uplo) The echelon structure is defined by the first R values of the array P. row and column dimension of T are greater or equal to that of A

#### Parameters

|     |                           |                                                                                       |
|-----|---------------------------|---------------------------------------------------------------------------------------|
|     | <i>F</i>                  | base field                                                                            |
|     | <i>UpLo</i>               | selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned |
|     | <i>diag</i>               | selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)                      |
|     | <i>M</i>                  | row dimension of T                                                                    |
|     | <i>N</i>                  | column dimension of T                                                                 |
|     | <i>R</i>                  | rank of the triangular matrix (how many rows/columns need to be copied)               |
|     | <i>P</i>                  | positions of the R pivots                                                             |
| in  | <i>A</i>                  | input matrix                                                                          |
|     | <i>lda</i>                | leading dimension of A                                                                |
| out | <i>T</i>                  | output matrix                                                                         |
|     | <i>ldt</i>                | leading dimension of T                                                                |
|     | <i>OnlyNonZeroVectors</i> | decides whether the last zero rows/columns should be ignored                          |
|     | <i>LuTag</i>              | which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)   |

### 15.20.3.73 getEchelonForm() [2/2]

```
void getEchelonForm (
    const Field & F,
```

```

const FFLAS::FFLAS_UPLO Uplo,
const FFLAS::FFLAS_DIAG diag,
const size_t M,
const size_t N,
const size_t R,
const size_t * P,
typename Field::Element_ptr A,
const size_t lda,
const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank  $R$ .

Either  $L$  or  $U$  is in Echelon form (depending on Uplo) The echelon structure is defined by the first  $R$  values of the array  $P$ .

#### Parameters

|         |         |                                                                                       |
|---------|---------|---------------------------------------------------------------------------------------|
|         | $F$     | base field                                                                            |
|         | $UpLo$  | selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned |
|         | $diag$  | selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)                      |
|         | $M$     | row dimension of $A$                                                                  |
|         | $N$     | column dimension of $A$                                                               |
|         | $R$     | rank of the triangular matrix (how many rows/columns need to be copied)               |
|         | $P$     | positions of the $R$ pivots                                                           |
| in, out | $A$     | input/output matrix                                                                   |
|         | $lda$   | leading dimension of $A$                                                              |
|         | $LuTag$ | which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)   |

#### 15.20.3.74 getEchelonTransform()

```

void getEchelonTransform (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by Row↔ EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower:  $T$  is  $N \times N$  (already allocated) such that  $A T = C$  is a transformation of  $A$  in Column echelon form Else  $T$  is  $M \times M$  (already allocated) such that  $T A = E$  is a transformation of  $A$  in Row Echelon form

## Parameters

|     |              |                                                                                                         |
|-----|--------------|---------------------------------------------------------------------------------------------------------|
|     | <i>F</i>     | base field                                                                                              |
|     | <i>UpLo</i>  | Lower (FflasLower) means Transformation to Column Echelon Form, Upper (FflasUpper), to Row Echelon Form |
|     | <i>diag</i>  | selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)                                        |
|     | <i>M</i>     | row dimension of A                                                                                      |
|     | <i>N</i>     | column dimension of A                                                                                   |
|     | <i>R</i>     | rank of the triangular matrix                                                                           |
|     | <i>P</i>     | permutation matrix                                                                                      |
| in  | <i>A</i>     | input matrix                                                                                            |
|     | <i>lda</i>   | leading dimension of A                                                                                  |
| out | <i>T</i>     | output matrix                                                                                           |
|     | <i>ldt</i>   | leading dimension of T                                                                                  |
|     | <i>LuTag</i> | which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)                     |

15.20.3.75 `getReducedEchelonForm()` [1/2]

```

void getReducedEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr T,
    const size_t ldt,
    const bool OnlyNonZeroVectors = false,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `ReducedRowEchelonForm` or `ReducedColumnEchelonForm` with `transform = true`.

Either  $L$  or  $U$  is in Echelon form (depending on `Uplo`) The echelon structure is defined by the first  $R$  values of the array `P`. row and column dimension of `T` are greater or equal to that of `A`

## Parameters

|    |             |                                                                                       |
|----|-------------|---------------------------------------------------------------------------------------|
|    | <i>F</i>    | base field                                                                            |
|    | <i>UpLo</i> | selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned |
|    | <i>diag</i> | selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)                      |
|    | <i>M</i>    | row dimension of T                                                                    |
|    | <i>N</i>    | column dimension of T                                                                 |
|    | <i>R</i>    | rank of the triangular matrix (how many rows/columns need to be copied)               |
|    | <i>P</i>    | positions of the $R$ pivots                                                           |
| in | <i>A</i>    | input matrix                                                                          |
|    | <i>lda</i>  | leading dimension of A                                                                |

## Parameters

|  |                           |                                                                                     |
|--|---------------------------|-------------------------------------------------------------------------------------|
|  | <i>ldt</i>                | leading dimension of T                                                              |
|  | <i>LuTag</i>              | which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive) |
|  | <i>OnlyNonZeroVectors</i> | decides whether the last zero rows/columns should be ignored                        |

15.20.3.76 `getReducedEchelonForm()` [2/2]

```

void getReducedEchelonForm (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    typename Field::Element_ptr A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Cleans up a compact storage  $A=L\backslash U$  of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.

Either L or U is in Echelon form (depending on Uplo) The echelon structure is defined by the first R values of the array P.

## Parameters

|                |              |                                                                                       |
|----------------|--------------|---------------------------------------------------------------------------------------|
|                | <i>F</i>     | base field                                                                            |
|                | <i>UpLo</i>  | selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned |
|                | <i>diag</i>  | selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)                      |
|                | <i>M</i>     | row dimension of A                                                                    |
|                | <i>N</i>     | column dimension of A                                                                 |
|                | <i>R</i>     | rank of the triangular matrix (how many rows/columns need to be copied)               |
|                | <i>P</i>     | positions of the R pivots                                                             |
| <i>in, out</i> | <i>A</i>     | input/output matrix                                                                   |
|                | <i>lda</i>   | leading dimension of A                                                                |
|                | <i>LuTag</i> | which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)   |

15.20.3.77 `getReducedEchelonTransform()`

```

void getReducedEchelonTransform (
    const Field & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,

```

```

const size_t N,
const size_t R,
const size_t * P,
const size_t * Q,
typename Field::ConstElement_ptr A,
const size_t lda,
typename Field::Element_ptr T,
const size_t ldt,
const FFPACK_LU_TAG LuTag = FfpackSlabRecursive ) [inline]

```

Extracts a transformation matrix to echelon form from a compact storage  $A=LU$  of rank  $R$  obtained by Row↔EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower:  $T$  is  $N \times N$  (already allocated) such that  $A T = C$  is a transformation of  $A$  in Column echelon form Else  $T$  is  $M \times M$  (already allocated) such that  $T A = E$  is a transformation of  $A$  in Row Echelon form

#### Parameters

|     |              |                                                                                     |
|-----|--------------|-------------------------------------------------------------------------------------|
|     | <i>F</i>     | base field                                                                          |
|     | <i>UpLo</i>  | selects Col (FflasLower) or Row (FflasUpper) Echelon Form                           |
|     | <i>diag</i>  | selects if the echelon matrix has unit pivots (FflasUnit/NoUnit)                    |
|     | <i>M</i>     | row dimension of A                                                                  |
|     | <i>N</i>     | column dimension of A                                                               |
|     | <i>R</i>     | rank of the triangular matrix                                                       |
|     | <i>P</i>     | permutation matrix                                                                  |
| in  | <i>A</i>     | input matrix                                                                        |
|     | <i>lda</i>   | leading dimension of A                                                              |
| out | <i>T</i>     | output matrix                                                                       |
|     | <i>ldt</i>   | leading dimension of T                                                              |
|     | <i>LuTag</i> | which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive) |

#### 15.20.3.78 PLUQtoEchelonPermutation()

```

void PLUQtoEchelonPermutation (
    const size_t N,
    const size_t R,
    const size_t * P,
    size_t * outPerm ) [inline]

```

Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.

#### 15.20.3.79 LQUPtoInverseOfFullRankMinor() [1/2]

```

Field::Element_ptr LQUPtoInverseOfFullRankMinor (
    const Field & F,
    const size_t rank,

```

```

typename Field::Element_ptr A_factors,
const size_t lda,
const size_t * QtPointer,
typename Field::Element_ptr X,
const size_t ldx )

```

LQUPtoInverseOfFullRankMinor.

Suppose A has been factorized as L.Q.U.P, with rank r. Then Qt.A.Pt has an invertible leading principal  $r \times r$  submatrix This procedure efficiently computes the inverse of this minor and puts it into X.

#### Note

It changes the lower entries of A\_factors in the process (NB: unless A was nonsingular and square)

#### Parameters

|                  |                                                            |
|------------------|------------------------------------------------------------|
| <i>F</i>         | base field                                                 |
| <i>rank</i>      | rank of the matrix.                                        |
| <i>A_factors</i> | matrix containing the L and U entries of the factorization |
| <i>lda</i>       | leading dimension of A                                     |
| <i>QtPointer</i> | theLQUP->getQ()->getPointer() (note: getQ returns Qt!)     |
| <i>X</i>         | desired location for output                                |
| <i>ldx</i>       | leading dimension of X                                     |

### 15.20.3.80 RandomNullSpaceVector() [2/3]

```

void RandomNullSpaceVector (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t incX )

```

Solve  $LX = B$  or  $XL = B$  in place.

L is  $M \times M$  if Side == FFLAS::FflasLeft and  $N \times N$  if Side == FFLAS::FflasRight, B is  $M \times N$ . Only the R non trivial column of L are stored in the  $M \times R$  matrix L Requirement : so that L could be expanded in-place Computes a vector of the Left/Right nullspace of the matrix A.

#### Parameters

|         |             |                                                                                  |
|---------|-------------|----------------------------------------------------------------------------------|
|         | <i>F</i>    | The computation domain                                                           |
|         | <i>Side</i> | decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace |
|         | <i>M</i>    | number of rows                                                                   |
|         | <i>N</i>    | number of columns                                                                |
| in, out | <i>A</i>    | input matrix of dimension $M \times N$ , A is modified to its LU version         |
|         | <i>lda</i>  | leading dimension of A                                                           |
| out     | <i>X</i>    | output vector                                                                    |
|         | <i>incX</i> | increment of X                                                                   |

**15.20.3.81 solveLB()** [1/2]

```
void solveLB (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr L,
    const size_t ldl,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb )
```

**15.20.3.82 solveLB2()** [1/2]

```
void solveLB2 (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    typename Field::Element_ptr L,
    const size_t ldl,
    const size_t * Q,
    typename Field::Element_ptr B,
    const size_t ldb )
```

**15.20.3.83 Danilevski()**

```
std::list< Polynomial > & Danilevski (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

**15.20.3.84 buildMatrix()**

```
Field::Element_ptr buildMatrix (
    const Field & F,
    typename Field::ConstElement_ptr E,
    typename Field::ConstElement_ptr C,
    const size_t lda,
    const size_t * B,
    const size_t * T,
    const size_t me,
    const size_t mc,
    const size_t lambda,
    const size_t mu )
```

**Bug** is this :

**15.20.3.85 CharPoly()** [4/8]

```
FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr CharPoly (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr charp,
    const size_t N,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A,
    const size_t lda,
    Givaro::ZRing< Givaro::Integer >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    size_t degree ) [inline]
```

**15.20.3.86 CharPoly()** [5/8]

```
Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly (
    const Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > & R,
    Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & charp,
    const size_t N,
    Givaro::Integer * A,
    const size_t lda,
    Givaro::ZRing< Givaro::Integer >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    size_t degree ) [inline]
```

**15.20.3.87 Det()** [3/6]

```
FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det (
    const FFPACK::RNSInteger< FFPACK::rns_double > & F,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & det,
    const size_t N,
    typename FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr A,
    const size_t lda,
    const PSHelper & psH ) [inline]
```



**15.20.3.88 Det() [4/6]**

```
Givaro::Integer & Det (
    const Givaro::ZRing< Givaro::Integer > & F,
    Givaro::Integer & det,
    const size_t N,
    Givaro::Integer * A,
    const size_t lda,
    const PSHelper & psH,
    size_t * P,
    size_t * Q ) [inline]
```

**15.20.3.89 fsytrf\_BC\_Crout()**

```
bool fsytrf_BC_Crout (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv ) [inline]
```

**15.20.3.90 fsytrf\_BC\_RL()**

```
size_t fsytrf_BC_RL (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv ) [inline]
```

**15.20.3.91 fsytrf\_UP\_RPM\_BC\_RL()**

```
size_t fsytrf_UP_RPM_BC_RL (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]
```

**15.20.3.92 fsytrf\_LOW\_RPM\_BC\_Crout()**

```
size_t fsytrf_LOW_RPM_BC_Crout (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]
```

**15.20.3.93 fsytrf\_UP\_RPM\_BC\_Crout()**

```
size_t fsytrf_UP_RPM_BC_Crout (
    const Field & F,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P ) [inline]
```

**15.20.3.94 fsytrf\_UP\_RPM()**

```
size_t fsytrf_UP_RPM (
    const Field & Fi,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    size_t * P,
    size_t BCThreshold ) [inline]
```

MathP <- [ [ I ] x P1 ] [ [ I\_(N1+R2) ] [ P2^T ] ] x [ P3^T ] [ ----- | --- ] [ [ Q2^T ]

Changing [ U1 V1 | E1 E21 E22 ] into [ U1 E11 E12 V1 E\* E\* ] [ 0 | L2 \ U2 V21 V22 ] [ U4 V41 0 V42 V43 ] [ 0 | M2 0 0 ] [ U3 0 0 V3 ] [ ----- | ----- ] [ [ 0 0 0 ] [ 0 | H1 H21 H22 ] [ 0 | U3 V3 ] [ 0 | 0 ] where U4 is the 2R2 x 2R2 matrix formed by interleaving U2, L2^T and H1

**15.20.3.95 fsytrf\_nonunit() [2/3]**

```
bool fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    FFLAS::ParSeqHelper::Sequential seq,
    size_t threshold ) [inline]
```

**15.20.3.96 fsytrf\_nonunit()** [3/3]

```
bool fsytrf_nonunit (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr Dinv,
    const size_t incDinv,
    FFLAS::ParSeqHelper::Parallel< Cut, Param > par,
    size_t threshold ) [inline]
```

**15.20.3.97 fsytrf\_RPM()**

```
size_t fsytrf_RPM (
    const Field & F,
    const FFLAS::FFLAS_UPLO UpLo,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t threshold ) [inline]
```

**15.20.3.98 getTridiagonal()**

```
void getTridiagonal (
    const Field & F,
    const size_t N,
    const size_t R,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    size_t * P,
    typename Field::Element_ptr T,
    const size_t ldt ) [inline]
```

**15.20.3.99 LUdivine\_gauss()** [1/2]

```
size_t LUdivine_gauss (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

**15.20.3.100 LUdivine\_small() [1/2]**

```

size_t LUdivine_small (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]

```

**15.20.3.101 LUdivine() [2/4]**

```

size_t LUdivine (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag,
    const size_t cutoff ) [inline]

```

**Todo** std::swap ?

**15.20.3.102 LUdivine() [3/4]**

```

size_t LUdivine (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Givaro::Integer * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag,
    const size_t cutoff ) [inline]

```

### 15.20.3.103 MonotonicCompress()

```
void MonotonicCompress (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t maxpiv,
    const size_t rowstomove,
    const std::vector< bool > & ispiv ) [inline]
```

### 15.20.3.104 MonotonicCompressMorePivots()

```
void MonotonicCompressMorePivots (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t rowstomove,
    const size_t lenP ) [inline]
```

### 15.20.3.105 MonotonicCompressCycles()

```
void MonotonicCompressCycles (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t lenP ) [inline]
```

**15.20.3.106 MonotonicExpand()**

```

void MonotonicExpand (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t incA,
    const size_t * MathP,
    const size_t R,
    const size_t maxpiv,
    const size_t rowstomove,
    const std::vector< bool > & ispiv )

```

**15.20.3.107 applyP\_block()**

```

void applyP_block (
    const Field & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t * P ) [inline]

```

**15.20.3.108 doApplyS()**

```

void doApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]

```

**15.20.3.109 MatrixApplyS() [1/3]**

```
void MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]
```

**15.20.3.110 MatrixApplyS() [2/3]**

```
void MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Sequential seq ) [inline]
```

**15.20.3.111 MatrixApplyS() [3/3]**

```
void MatrixApplyS (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par ) [inline]
```

**15.20.3.112 PermApplyS()**

```

void PermApplyS (
    T * A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]

```

**15.20.3.113 doApplyT()**

```

void doApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]

```

**15.20.3.114 MatrixApplyT() [1/3]**

```

void MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]

```

**15.20.3.115 MatrixApplyT() [2/3]**

```

void MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t N2,

```



```

const size_t R1,
const size_t R2,
const size_t R3,
const size_t R4,
const FFLAS::ParSeqHelper::Sequential seq ) [inline]

```

### 15.20.3.116 MatrixApplyT() [3/3]

```

void MatrixApplyT (
    const Field & F,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    const FFLAS::ParSeqHelper::Parallel< Cut, Param > par ) [inline]

```

### 15.20.3.117 PermApplyT()

```

void PermApplyT (
    T * A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 ) [inline]

```

### 15.20.3.118 composePermutationsLLL()

```

void composePermutationsLLL (
    size_t * P1,
    const size_t * P2,
    const size_t R,
    const size_t N ) [inline]

```

Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.

#### Parameters

|         |      |                                  |
|---------|------|----------------------------------|
| in, out | $P1$ | a LAPACK permutation of size N   |
|         | $P2$ | a LAPACK permutation of size N-R |

**15.20.3.119 composePermutationsLLM()**

```
void composePermutationsLLM (
    size_t * MathP,
    const size_t * P1,
    const size_t * P2,
    const size_t R,
    const size_t N ) [inline]
```

Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $\text{MathP}$  as a  $\text{MathPermutation}$  format.

**Parameters**

|     |  |  |
|-----|--|--|
| out |  |  |
|-----|--|--|

a  $\text{MathPermutation}$  of size  $N$

**Parameters**

|      |                                    |
|------|------------------------------------|
| $P1$ | a LAPACK permutation of size $N$   |
| $P2$ | a LAPACK permutation of size $N-R$ |

**15.20.3.120 composePermutationsMLM()**

```
void composePermutationsMLM (
    size_t * MathP1,
    const size_t * P2,
    const size_t R,
    const size_t N ) [inline]
```

Computes  $\text{MathP1} \times \text{Diag}(I_R, P2)$  where  $\text{MathP1}$  is a  $\text{MathPermutation}$  and  $P2$  a LAPACK permutation and store the result in  $\text{MathP1}$  as a  $\text{MathPermutation}$  format.

**Parameters**

|         |                 |                                        |
|---------|-----------------|----------------------------------------|
| in, out | $\text{MathP1}$ | a $\text{MathPermutation}$ of size $N$ |
|         | $P2$            | a LAPACK permutation of size $N-R$     |

**15.20.3.121 cyclic\_shift\_mathPerm()**

```
void cyclic_shift_mathPerm (
    size_t * P,
    const size_t s ) [inline]
```

**15.20.3.122 cyclic\_shift\_row\_col() [1/2]**

```
void cyclic_shift_row_col (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.20.3.123 cyclic\_shift\_row() [1/3]**

```
void cyclic_shift_row (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.20.3.124 cyclic\_shift\_row() [2/3]**

```
void cyclic_shift_row (
    const RNSIntegerMod< T > & F,
    typename T::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.20.3.125 cyclic\_shift\_col() [1/3]**

```
void cyclic_shift_col (
    const Field & F,
    typename Field::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.20.3.126 cyclic\_shift\_col()** [2/3]

```
void cyclic_shift_col (
    const RNSIntegerMod< T > & F,
    typename T::Element_ptr A,
    size_t m,
    size_t n,
    size_t lda ) [inline]
```

**15.20.3.127 PLUQ\_basecaseV3()**

```
size_t PLUQ_basecaseV3 (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element * A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

**15.20.3.128 PLUQ\_basecaseV2()**

```
size_t PLUQ_basecaseV2 (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element * A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

**15.20.3.129 PLUQ\_basecaseCrout()**

```
size_t PLUQ_basecaseCrout (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q ) [inline]
```

**15.20.3.130 \_PLUQ()**

```
size_t _PLUQ (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    size_t BCThreshold ) [inline]
```

**15.20.3.131 PLUQ() [4/6]**

```
size_t PLUQ (
    const Givaro::Modular< Givaro::Integer > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Givaro::Integer * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    size_t BCThreshold,
    FFLAS::ParSeqHelper::Parallel< Cut, Param > & PSHelper ) [inline]
```

**15.20.3.132 threads\_fgemm()**

```
void threads_fgemm (
    const size_t m,
    const size_t n,
    const size_t r,
    int nbthreads,
    size_t * W1,
    size_t * W2,
    size_t * W3,
    size_t gamma )
```

**15.20.3.133 threads\_ftrsm()**

```
void threads_ftrsm (
    const size_t m,
    const size_t n,
    int nbthreads,
    size_t * t1,
    size_t * t2 )
```

**15.20.3.134 PLUQ()** [5/6]

```

size_t PLUQ (
    const Field & Fi,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter
> & PSHelper ) [inline]

```

**15.20.3.135 fflas\_const\_cast()** [1/3]

```

rns_double_elt_ptr fflas_const_cast (
    rns_double_elt_cstptr x ) [inline]

```

**15.20.3.136 fflas\_const\_cast()** [2/3]

```

rns_double_elt_cstptr fflas_const_cast (
    rns_double_elt_ptr x ) [inline]

```

**15.20.3.137 cyclic\_shift\_row\_col()** [2/2]

```

void cyclic_shift_row_col (
    Base_t * A,
    size_t m,
    size_t n,
    size_t lda )

```

**15.20.3.138 cyclic\_shift\_row()** [3/3]

```

template INST_OR_DECL void cyclic_shift_row (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT * A,
    size_t m,
    size_t n,
    size_t lda )

```

**15.20.3.139 cyclic\_shift\_col()** [3/3]

```
template INST_OR_DECL void cyclic_shift_col (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT * A,
    size_t m,
    size_t n,
    size_t lda )
```

**15.20.3.140 applyP()** [4/4]

```
template INST_OR_DECL void applyP (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t * P )
```

**15.20.3.141 fgetrs()** [3/4]

```
template INST_OR_DECL void fgetrs (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb,
    int * info )
```

**15.20.3.142 fgetrs()** [4/4]

```
template INST_OR_DECL FFLAS_ELT * fgetrs (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    const size_t R,
```

```

FFLAS_ELT * A,
const size_t lda,
const size_t * P,
const size_t * Q,
FFLAS_ELT * X,
const size_t ldx,
const FFLAS_ELT * B,
const size_t ldb,
int * info )

```

#### 15.20.3.143 fgesv() [3/4]

```

template INST_OR_DECL size_t fgesv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * B,
    const size_t ldb,
    int * info )

```

#### 15.20.3.144 fgesv() [4/4]

```

template INST_OR_DECL size_t fgesv (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldx,
    const FFLAS_ELT * B,
    const size_t ldb,
    int * info )

```

#### 15.20.3.145 ftrtri() [2/2]

```

template INST_OR_DECL void ftrtri (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t threshold )

```



**15.20.3.146 trinv\_left()** [2/2]

```
template INST_OR_DECL void trinv_left (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t N,
    const FFLAS_ELT * L,
    const size_t ldl,
    FFLAS_ELT * X,
    const size_t ldX )
```

**15.20.3.147 ftrtrm()** [2/2]

```
template INST_OR_DECL void ftrtrm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_DIAG diag,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.20.3.148 PLUQ()** [6/6]

```
template INST_OR_DECL size_t PLUQ (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q )
```

**15.20.3.149 LUdivine()** [4/4]

```
template INST_OR_DECL size_t LUdivine (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const FFPACK_LU_TAG LuTag,
    const size_t cutoff )
```

**15.20.3.150 LUdivine\_small() [2/2]**

```
template INST_OR_DECL size_t LUdivine_small (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.151 LUdivine\_gauss() [2/2]**

```
template INST_OR_DECL size_t LUdivine_gauss (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.152 RowEchelonForm() [3/3]**

```
template INST_OR_DECL size_t RowEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.153 ReducedRowEchelonForm()** [3/3]

```
template INST_OR_DECL size_t ReducedRowEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.154 ColumnEchelonForm()** [3/3]

```
template INST_OR_DECL size_t ColumnEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.155 ReducedColumnEchelonForm()** [3/3]

```
template INST_OR_DECL size_t ReducedColumnEchelonForm (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.156 Invert()** [3/4]

```
template INST_OR_DECL FFLAS_ELT * Invert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    int & nullity )
```

**15.20.3.157 Invert()** [4/4]

```
template INST_OR_DECL FFLAS_ELT * Invert (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldx,
    int & nullity )
```

**15.20.3.158 Invert2()** [2/2]

```
template INST_OR_DECL FFLAS_ELT * Invert2 (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t ldx,
    int & nullity )
```

**15.20.3.159 CharPoly()** [6/8]

```
template INST_OR_DECL std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > &
CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

**15.20.3.160 CharPoly()** [7/8]

```
template INST_OR_DECL Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

**15.20.3.161 CharPoly()** [8/8]

```
template INST_OR_DECL Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly (
    const Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > & R,
    Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & charp,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_CHARPOLY_TAG CharpTag,
    const size_t degree )
```

**15.20.3.162 MinPoly()** [3/4]

```
template INST_OR_DECL std::vector< FFLAS_ELT > & MinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_FIELD< FFLAS_ELT >::RandIter & G )
```

**15.20.3.163 MinPoly()** [4/4]

```
template INST_OR_DECL std::vector< FFLAS_ELT > & MinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda )
```

**15.20.3.164 MatVecMinPoly()** [2/2]

```
template INST_OR_DECL std::vector< FFLAS_ELT > & MatVecMinPoly (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    std::vector< FFLAS_ELT > & minP,
    const size_t N,
    const FFLAS_ELT * A,
    const size_t lda,
    const FFLAS_ELT * V,
    const size_t incv )
```

**15.20.3.165 KrylovElim()**

```
template INST_OR_DECL size_t KrylovElim (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const size_t deg,
    size_t * iterates,
    size_t * inviterates,
    const size_t maxit,
    size_t virt )
```

**15.20.3.166 SpecRankProfile()**

```
template INST_OR_DECL size_t SpecRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const size_t deg,
    size_t * rankProfile )
```

**15.20.3.167 Rank() [3/3]**

```
template INST_OR_DECL size_t Rank (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.20.3.168 IsSingular() [2/2]**

```
template INST_OR_DECL bool IsSingular (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.20.3.169 Det()** [5/6]

```
template INST_OR_DECL FFLAS_ELT & Det (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT & det,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t * P,
    size_t * Q )
```

**15.20.3.170 Det()** [6/6]

```
template INST_OR_DECL FFLAS_ELT & Det (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    FFLAS_ELT & det,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    const FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter
> & parH,
    size_t * P,
    size_t * Q )
```

**15.20.3.171 Solve()** [3/3]

```
template INST_OR_DECL FFLAS_ELT * Solve (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * x,
    const int incx,
    const FFLAS_ELT * b,
    const int incb )
```

**15.20.3.172 solveLB()** [2/2]

```
template INST_OR_DECL void solveLB (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * L,
    const size_t ldl,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb )
```

**15.20.3.173 solveLB2()** [2/2]

```
template INST_OR_DECL void solveLB2 (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * L,
    const size_t ldl,
    const size_t * Q,
    FFLAS_ELT * B,
    const size_t ldb )
```

**15.20.3.174 RandomNullSpaceVector()** [3/3]

```
template INST_OR_DECL void RandomNullSpaceVector (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * X,
    const size_t incX )
```

**15.20.3.175 NullSpaceBasis()** [2/2]

```
template INST_OR_DECL size_t NullSpaceBasis (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT *& NS,
    size_t & ldn,
    size_t & NSdim )
```

**15.20.3.176 RowRankProfile()** [3/3]

```
template INST_OR_DECL size_t RowRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag )
```



**15.20.3.177 ColumnRankProfile()** [3/3]

```
template INST_OR_DECL size_t ColumnRankProfile (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rkprofile,
    const FFPACK_LU_TAG LuTag )
```

**15.20.3.178 RowRankProfileSubmatrixIndices()** [2/2]

```
template INST_OR_DECL size_t RowRankProfileSubmatrixIndices (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )
```

**15.20.3.179 ColRankProfileSubmatrixIndices()** [2/2]

```
template INST_OR_DECL size_t ColRankProfileSubmatrixIndices (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    size_t *& rowindices,
    size_t *& colindices,
    size_t & R )
```

**15.20.3.180 RowRankProfileSubmatrix()** [2/2]

```
template INST_OR_DECL size_t RowRankProfileSubmatrix (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT *& X,
    size_t & R )
```

**15.20.3.181 ColRankProfileSubmatrix() [2/2]**

```
template INST_OR_DECL size_t ColRankProfileSubmatrix (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t M,
    const size_t N,
    FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT *& X,
    size_t & R )
```

**15.20.3.182 getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >() [1/2]**

```
template INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors )
```

**15.20.3.183 getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >() [2/2]**

```
template INST_OR_DECL void getTriangular< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    FFLAS_ELT * A,
    const size_t lda )
```

**15.20.3.184 getEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >() [1/2]**

```
template INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
```

```

const size_t * P,
const FFLAS_ELT * A,
const size_t lda,
FFLAS_ELT * T,
const size_t ldt,
const bool OnlyNonZeroVectors,
const FFPACK_LU_TAG LuTag )

```

### 15.20.3.185 getEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >() [2/2]

```

template INST_OR_DECL void getEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag )

```

### 15.20.3.186 getEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >()

```

template INST_OR_DECL void getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const FFLAS::FFLAS_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag )

```

### 15.20.3.187 getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >() [1/2]

```

template INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,

```

```

const size_t * P,
const FFLAS_ELT * A,
const size_t lda,
FFLAS_ELT * T,
const size_t ldt,
const bool OnlyNonZeroVectors,
const FFPACK_LU_TAG LuTag )

```

### 15.20.3.188 getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >() [2/2]

```

template INST_OR_DECL void getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    FFLAS_ELT * A,
    const size_t lda,
    const FFPACK_LU_TAG LuTag )

```

### 15.20.3.189 getReducedEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >()

```

template INST_OR_DECL void getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > > (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const FFLAS_ELT * A,
    const size_t lda,
    FFLAS_ELT * T,
    const size_t ldt,
    const FFPACK_LU_TAG LuTag )

```

### 15.20.3.190 LQUPtoInverseOfFullRankMinor() [2/2]

```

template INST_OR_DECL FFLAS_ELT * LQUPtoInverseOfFullRankMinor (
    const FFLAS_FIELD< FFLAS_ELT > & F,
    const size_t rank,
    FFLAS_ELT * A_factors,
    const size_t lda,
    const size_t * QtPointer,
    FFLAS_ELT * X,
    const size_t ldx )

```

**15.20.3.191 fflas\_const\_cast()** [3/3]

```
T fflas_const_cast (
    CT x )
```

**15.20.3.192 failure()**

```
Failure & failure ( ) [inline]
```

**15.20.3.193 isOdd()** [1/3]

```
bool isOdd (
    const T & a ) [inline]
```

**15.20.3.194 isOdd()** [2/3]

```
bool isOdd (
    const float & a ) [inline]
```

**15.20.3.195 isOdd()** [3/3]

```
bool isOdd (
    const double & a ) [inline]
```

**15.20.3.196 NonZeroRandomMatrix()** [1/2]

```
Field::Element_ptr NonZeroRandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random non-zero Matrix.

Creates a  $m \times n$  matrix with random entries, and at least one of them is non zero.

## Parameters

|     |              |                                                                            |
|-----|--------------|----------------------------------------------------------------------------|
|     | $F$          | field                                                                      |
|     | $m$          | number of rows in $A$                                                      |
|     | $n$          | number of cols in $A$                                                      |
| out | $A$          | the matrix (preallocated to at least $m \times \text{lda}$ field elements) |
|     | $\text{lda}$ | leading dimension of $A$                                                   |
|     | $G$          | a random iterator                                                          |

## Returns

$A$ .

## 15.20.3.197 NonZeroRandomMatrix() [2/2]

```
Field::Element_ptr NonZeroRandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random non-zero Matrix.

Creates a  $m \times n$  matrix with random entries, and at least one of them is non zero.

## Parameters

|     |              |                                                                            |
|-----|--------------|----------------------------------------------------------------------------|
|     | $F$          | field                                                                      |
|     | $m$          | number of rows in $A$                                                      |
|     | $n$          | number of cols in $A$                                                      |
| out | $A$          | the matrix (preallocated to at least $m \times \text{lda}$ field elements) |
|     | $\text{lda}$ | leading dimension of $A$                                                   |

## Returns

$A$ .

## 15.20.3.198 RandomMatrix() [1/2]

```
Field::Element_ptr RandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
```

```
size_t lda,
RandIter & G ) [inline]
```

Random Matrix.

Creates a  $m \times n$  matrix with random entries.

#### Parameters

|     |       |                                                                     |
|-----|-------|---------------------------------------------------------------------|
|     | $F$   | field                                                               |
|     | $m$   | number of rows in $A$                                               |
|     | $n$   | number of cols in $A$                                               |
| out | $A$   | the matrix (preallocated to at least $m \times lda$ field elements) |
|     | $lda$ | leading dimension of $A$                                            |
|     | $G$   | a random iterator                                                   |

#### Returns

$A$ .

#### 15.20.3.199 RandomMatrix() [2/2]

```
Field::Element_ptr RandomMatrix (
    const Field & F,
    size_t m,
    size_t n,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix.

Creates a  $m \times n$  matrix with random entries.

#### Parameters

|     |       |                                                                     |
|-----|-------|---------------------------------------------------------------------|
|     | $F$   | field                                                               |
|     | $m$   | number of rows in $A$                                               |
|     | $n$   | number of cols in $A$                                               |
| out | $A$   | the matrix (preallocated to at least $m \times lda$ field elements) |
|     | $lda$ | leading dimension of $A$                                            |

#### Returns

$A$ .

**15.20.3.200 RandomTriangularMatrix() [1/2]**

```
Field::Element_ptr RandomTriangularMatrix (
    const Field & F,
    size_t m,
    size_t n,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_DIAG Diag,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Triangular Matrix.

Creates a  $m \times n$  triangular matrix with random entries. The `UpLo` parameter defines whether it is upper or lower triangular.

**Parameters**

|     |             |                                                                     |
|-----|-------------|---------------------------------------------------------------------|
|     | <i>F</i>    | field                                                               |
|     | <i>m</i>    | number of rows in A                                                 |
|     | <i>n</i>    | number of cols in A                                                 |
|     | <i>UpLo</i> | whether A is upper or lower triangular                              |
| out | <i>A</i>    | the matrix (preallocated to at least $m \times lda$ field elements) |
|     | <i>lda</i>  | leading dimension of A                                              |
|     | <i>G</i>    | a random iterator                                                   |

**Returns**

A.

**15.20.3.201 RandomTriangularMatrix() [2/2]**

```
Field::Element_ptr RandomTriangularMatrix (
    const Field & F,
    size_t m,
    size_t n,
    const FFLAS::FFLAS_UPLO UpLo,
    const FFLAS::FFLAS_DIAG Diag,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Triangular Matrix.

Creates a  $m \times n$  triangular matrix with random entries. The `UpLo` parameter defines whether it is upper or lower triangular.



## Parameters

|     |        |                                                                     |
|-----|--------|---------------------------------------------------------------------|
|     | $F$    | field                                                               |
|     | $m$    | number of rows in $A$                                               |
|     | $n$    | number of cols in $A$                                               |
|     | $UpLo$ | whether $A$ is upper or lower triangular                            |
| out | $A$    | the matrix (preallocated to at least $m \times lda$ field elements) |
|     | $lda$  | leading dimension of $A$                                            |

## Returns

$A$ .

## 15.20.3.202 RandInt()

```
size_t RandInt (
    size_t a,
    size_t b ) [inline]
```

## 15.20.3.203 RandomSymmetricMatrix()

```
Field::Element_ptr RandomSymmetricMatrix (
    const Field & F,
    size_t n,
    bool nonsingular,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Symmetric Matrix.

Creates a  $m \times n$  triangular matrix with random entries. The  $UpLo$  parameter defines whether it is upper or lower triangular.

## Parameters

|     |       |                                                                     |
|-----|-------|---------------------------------------------------------------------|
|     | $F$   | field                                                               |
|     | $n$   | order of $A$                                                        |
| out | $A$   | the matrix (preallocated to at least $n \times lda$ field elements) |
|     | $lda$ | leading dimension of $A$                                            |
|     | $G$   | a random iterator                                                   |

## Returns

$A$ .

**15.20.3.204 RandomMatrixWithRank()** [1/2]

```
Field::Element_ptr RandomMatrixWithRank (
    const Field & F,
    size_t m,
    size_t n,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Matrix with prescribed rank.

Creates an  $m \times n$  matrix with random entries and rank  $r$ .

**Parameters**

|       |                                                                     |
|-------|---------------------------------------------------------------------|
| $F$   | field                                                               |
| $m$   | number of rows in $A$                                               |
| $n$   | number of cols in $A$                                               |
| $r$   | rank of the matrix to build                                         |
| $A$   | the matrix (preallocated to at least $m \times lda$ field elements) |
| $lda$ | leading dimension of $A$                                            |
| $G$   | a random iterator                                                   |

**Returns**

$A$ .

**15.20.3.205 RandomMatrixWithRank()** [2/2]

```
Field::Element_ptr RandomMatrixWithRank (
    const Field & F,
    size_t m,
    size_t n,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix with prescribed rank.

Creates an  $m \times n$  matrix with random entries and rank  $r$ .

**Parameters**

|     |       |                                                                     |
|-----|-------|---------------------------------------------------------------------|
|     | $F$   | field                                                               |
|     | $m$   | number of rows in $A$                                               |
|     | $n$   | number of cols in $A$                                               |
|     | $r$   | rank of the matrix to build                                         |
| out | $A$   | the matrix (preallocated to at least $m \times lda$ field elements) |
|     | $lda$ | leading dimension of $A$                                            |

## Returns

A.

**15.20.3.206 RandomIndexSubset()**

```
size_t * RandomIndexSubset (
    size_t N,
    size_t R,
    size_t * P ) [inline]
```

Pick uniformly at random a sequence of  $R$  distinct elements from the set  $\{0, \dots, N - 1\}$  using Knuth's shuffle.

## Parameters

|     |     |                                                             |
|-----|-----|-------------------------------------------------------------|
|     | $N$ | the cardinality of the sampling set                         |
|     | $R$ | the number of elements to sample                            |
| out | $P$ | the output sequence (pre-allocated to at least $R$ indices) |

**15.20.3.207 RandomPermutation()**

```
size_t * RandomPermutation (
    size_t N,
    size_t * P ) [inline]
```

Pick uniformly at random a permutation of size  $N$  stored in LAPACK format using Knuth's shuffle.

## Parameters

|     |     |                                                                |
|-----|-----|----------------------------------------------------------------|
|     | $N$ | the length of the permutation                                  |
| out | $P$ | the output permutation (pre-allocated to at least $N$ indices) |

**15.20.3.208 RandomRankProfileMatrix()**

```
void RandomRankProfileMatrix (
    size_t M,
    size_t N,
    size_t R,
    size_t * rows,
    size_t * cols ) [inline]
```

Pick uniformly at random an  $R$ -subpermutation of dimension  $M \times N$  : a matrix with only  $R$  non-zeros equal to one, in a random rook placement.

## Parameters

|     |             |                                                              |
|-----|-------------|--------------------------------------------------------------|
|     | $M$         | row dimension                                                |
|     | $N$         | column dimension                                             |
| out | <i>rows</i> | the row position of each non zero element (pre-allocated)    |
| out | <i>cols</i> | the column position of each non zero element (pre-allocated) |

**15.20.3.209 swapval()**

```
void swapval (
    size_t k,
    size_t N,
    size_t * P,
    size_t val ) [inline]
```

**15.20.3.210 RandomSymmetricRankProfileMatrix()**

```
void RandomSymmetricRankProfileMatrix (
    size_t N,
    size_t R,
    size_t * rows,
    size_t * cols ) [inline]
```

Pick uniformly at random a symmetric R-subpermutation of dimension  $N \times N$  : a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.

## Parameters

|     |             |                                                              |
|-----|-------------|--------------------------------------------------------------|
|     | $N$         | matrix order                                                 |
| out | <i>rows</i> | the row position of each non zero element (pre-allocated)    |
| out | <i>cols</i> | the column position of each non zero element (pre-allocated) |

**15.20.3.211 RandomMatrixWithRankandRPM()** [1/2]

```
Field::Element_ptr RandomMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
```

```
const size_t * CRP,
RandIter & G ) [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .

#### Parameters

|            |                                                                                  |
|------------|----------------------------------------------------------------------------------|
| <i>F</i>   | field                                                                            |
| <i>m</i>   | number of rows in A                                                              |
| <i>n</i>   | number of cols in A                                                              |
| <i>r</i>   | rank of the matrix to build                                                      |
| <i>A</i>   | the matrix (preallocated to at least $m \times lda$ field elements)              |
| <i>lda</i> | leading dimension of A                                                           |
| <i>RRP</i> | the R dimensional array with row positions of the rank profile matrix' pivots    |
| <i>CRP</i> | the R dimensional array with column positions of the rank profile matrix' pivots |
| <i>G</i>   | a random iterator                                                                |

#### Returns

A.

#### 15.20.3.212 RandomMatrixWithRankandRPM() [2/2]

```
Field::Element_ptr RandomMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP ) [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .

#### Parameters

|            |                                                                                  |
|------------|----------------------------------------------------------------------------------|
| <i>F</i>   | field                                                                            |
| <i>m</i>   | number of rows in A                                                              |
| <i>n</i>   | number of cols in A                                                              |
| <i>r</i>   | rank of the matrix to build                                                      |
| <i>A</i>   | the matrix (preallocated to at least $m \times lda$ field elements)              |
| <i>lda</i> | leading dimension of A                                                           |
| <i>RRP</i> | the R dimensional array with row positions of the rank profile matrix' pivots    |
| <i>CRP</i> | the R dimensional array with column positions of the rank profile matrix' pivots |

## Returns

A.

### 15.20.3.213 RandomSymmetricMatrixWithRankandRPM() [1/2]

```
Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (
    const Field & F,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP,
    RandIter & G ) [inline]
```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .

## Parameters

|       |                                                                                  |
|-------|----------------------------------------------------------------------------------|
| $F$   | field                                                                            |
| $n$   | order of A                                                                       |
| $r$   | rank of A                                                                        |
| $A$   | the matrix (preallocated to at least $n \times lda$ field elements)              |
| $lda$ | leading dimension of A                                                           |
| $RRP$ | the R dimensional array with row positions of the rank profile matrix' pivots    |
| $CRP$ | the R dimensional array with column positions of the rank profile matrix' pivots |
| $G$   | a random iterator                                                                |

## Returns

A.

### 15.20.3.214 RandomSymmetricMatrixWithRankandRPM() [2/2]

```
Field::Element_ptr RandomSymmetricMatrixWithRankandRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    const size_t * RRP,
    const size_t * CRP ) [inline]
```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .

## Parameters

|            |                                                                                  |
|------------|----------------------------------------------------------------------------------|
| <i>F</i>   | field                                                                            |
| <i>n</i>   | order of A                                                                       |
| <i>r</i>   | rank of A                                                                        |
| <i>A</i>   | the matrix (preallocated to at least $n \times lda$ field elements)              |
| <i>lda</i> | leading dimension of A                                                           |
| <i>RRP</i> | the R dimensional array with row positions of the rank profile matrix' pivots    |
| <i>CRP</i> | the R dimensional array with column positions of the rank profile matrix' pivots |

## Returns

A.

**15.20.3.215 RandomMatrixWithRankandRandomRPM()** [1/2]

```
Field::Element_ptr RandomMatrixWithRankandRandomRPM (
    const Field & F,
    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

## Parameters

|            |                                                                     |
|------------|---------------------------------------------------------------------|
| <i>F</i>   | field                                                               |
| <i>m</i>   | number of rows in A                                                 |
| <i>n</i>   | number of cols in A                                                 |
| <i>r</i>   | rank of the matrix to build                                         |
| <i>A</i>   | the matrix (preallocated to at least $m \times lda$ field elements) |
| <i>lda</i> | leading dimension of A                                              |
| <i>G</i>   | a random iterator                                                   |

## Returns

A.

**15.20.3.216 RandomMatrixWithRankandRandomRPM()** [2/2]

```
Field::Element_ptr RandomMatrixWithRankandRandomRPM (
    const Field & F,
```

```

    size_t M,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda ) [inline]

```

Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

#### Parameters

|       |                                                                     |
|-------|---------------------------------------------------------------------|
| $F$   | field                                                               |
| $m$   | number of rows in $A$                                               |
| $n$   | number of cols in $A$                                               |
| $r$   | rank of the matrix to build                                         |
| $A$   | the matrix (preallocated to at least $m \times lda$ field elements) |
| $lda$ | leading dimension of $A$                                            |

#### Returns

$A$ .

### 15.20.3.217 RandomSymmetricMatrixWithRankandRandomRPM() [1/2]

```

Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (
    const Field & F,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]

```

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

#### Parameters

|       |                                                                     |
|-------|---------------------------------------------------------------------|
| $F$   | field                                                               |
| $n$   | order of $A$                                                        |
| $r$   | rank of $A$                                                         |
| $A$   | the matrix (preallocated to at least $n \times lda$ field elements) |
| $lda$ | leading dimension of $A$                                            |
| $G$   | a random iterator                                                   |

#### Returns

$A$ .



**15.20.3.218 RandomSymmetricMatrixWithRankandRandomRPM()** [2/2]

```
Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM (
    const Field & F,
    size_t N,
    size_t R,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.

**Parameters**

|       |                                                                     |
|-------|---------------------------------------------------------------------|
| $F$   | field                                                               |
| $n$   | order of $A$                                                        |
| $r$   | rank of $A$                                                         |
| $A$   | the matrix (preallocated to at least $n \times lda$ field elements) |
| $lda$ | leading dimension of $A$                                            |

**Returns**

$A$ .

**15.20.3.219 RandomMatrixWithDet()** [1/2]

```
Field::Element_ptr RandomMatrixWithDet (
    const Field & F,
    size_t n,
    const typename Field::Element d,
    typename Field::Element_ptr A,
    size_t lda ) [inline]
```

Random Matrix with prescribed det.

Creates a  $m \times n$  matrix with random entries and rank  $r$ .

**Parameters**

|       |                                                                                     |
|-------|-------------------------------------------------------------------------------------|
| $F$   | field                                                                               |
| $d$   | the prescribed value for the determinant of $A$                                     |
| $n$   | number of cols in $A$                                                               |
| $A$   | the matrix to be generated (preallocated to at least $n \times lda$ field elements) |
| $lda$ | leading dimension of $A$                                                            |

**Returns**

$A$ .

**15.20.3.220 RandomMatrixWithDet()** [2/2]

```
Field::Element_ptr RandomMatrixWithDet (
    const Field & F,
    size_t n,
    const typename Field::Element d,
    typename Field::Element_ptr A,
    size_t lda,
    RandIter & G ) [inline]
```

Random Matrix with prescribed det.

Creates a  $m \times n$  matrix with random entries and rank  $r$ .

**Parameters**

|       |                                                                                     |
|-------|-------------------------------------------------------------------------------------|
| $F$   | field                                                                               |
| $d$   | the prescribed value for the determinant of A                                       |
| $n$   | number of cols in A                                                                 |
| $A$   | the matrix to be generated (preallocated to at least $n \times lda$ field elements) |
| $lda$ | leading dimension of A                                                              |

**Returns**

A.

**15.20.3.221 maxFieldElt()**

```
Givaro::Integer maxFieldElt ( )
```

**15.20.3.222 maxFieldElt< Givaro::ZRing< Givaro::Integer > >()**

```
Givaro::Integer maxFieldElt< Givaro::ZRing< Givaro::Integer > > ( )
```

**15.20.3.223 chooseField()**

```
Field * chooseField (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

**15.20.3.224 chooseField< Givaro::ZRing< int32\_t > >()**

```
Givaro::ZRing< int32_t > * chooseField< Givaro::ZRing< int32_t > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

**15.20.3.225 chooseField< Givaro::ZRing< int64\_t > >()**

```
Givaro::ZRing< int64_t > * chooseField< Givaro::ZRing< int64_t > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

**15.20.3.226 chooseField< Givaro::ZRing< float > >()**

```
Givaro::ZRing< float > * chooseField< Givaro::ZRing< float > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

**15.20.3.227 chooseField< Givaro::ZRing< double > >()**

```
Givaro::ZRing< double > * chooseField< Givaro::ZRing< double > > (
    Givaro::Integer q,
    uint64_t b,
    uint64_t seed )
```

**15.21 FFPACK::Protected Namespace Reference****Functions**

- template<class [Field](#) >  
size\_t [LUdivine\\_construct](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, typename [Field::Element\\_ptr](#) u, const size\_t incu, size\_t \*P, bool computeX, const FFPACK\_MINPOLY\_TAG MinTag=[FfpackDense](#), const size\_t kg\_mc=0, const size\_t kg\_mb=0, const size\_t kg\_j=0)
- template<class [Field](#) >  
size\_t [GaussJordan](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t colbeg, const size\_t rowbeg, const size\_t colsize, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag)

*Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.*

- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size_t N, type-`  
`name Field::ConstElement_ptr A, const size_t lda)`
- `template<class Field , class Polynomial >`  
`int KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A,`  
`const size_t lda, size_t *kg_mc, size_t *kg_mb, size_t *kg_j)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & KGFast_generalized (const Field &F, std::list< Polynomial > &charp, const size_t`  
`N, typename Field::Element_ptr A, const size_t lda)`
- `template<class Field >`  
`void fgemv_kgf (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, type-`  
`name Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, const`  
`size_t kg_mc, const size_t kg_mb, const size_t kg_j)`
- `template<class Field , class Polynomial , class RandIter >`  
`std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size_t N, typename`  
`Field::Element_ptr A, const size_t lda, typename Field::Element_ptr U, const size_t ldu, RandIter &G)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, type-`  
`name Field::Element_ptr A, const size_t lda)`
- `template<class PolRing >`  
`void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completed,`  
`Factors, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda, size_t &Nb, typename`  
`PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing::Domain_t::RandIter &g, const size_t`  
`t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`
- `template<class PolRing >`  
`std::list< typename PolRing::Element > & ArithProg (const PolRing &PR, std::list< typename PolRing::`  
`Element > &frobeniusForm, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t lda,`  
`const size_t degree)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & LUKrylov_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N,`  
`typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx)`
- `template<class Field , class Polynomial >`  
`Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr`  
`A, const size_t lda, typename Field::Element_ptr v, const size_t incv, typename Field::Element_ptr K, const`  
`size_t ldk, size_t *P)`
- `template<class Field , class Polynomial >`  
`Polynomial & Hybrid_KGF_LUK_MinPoly (const Field &F, Polynomial &minP, const size_t N, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, size_t *P,`  
`const FFPACK_MINPOLY_TAG MinTag=FFPACK::FfpackDense, const size_t kg_mc=0, const size_t kg_`  
`mb=0, const size_t kg_j=0)`
- `template<class Field >`  
`size_t updatedD (const Field &F, size_t *d, size_t k, std::vector< std::vector< typename Field::Element > >`  
`&minpt)`
- `template<class Field >`  
`size_t newD (const Field &F, size_t *d, bool &KeepOn, const size_t l, const size_t N, typename`  
`Field::Element_ptr X, const size_t *Q, std::vector< std::vector< typename Field::Element > > &minpt)`
- `template<class Field >`  
`void CompressRows (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename`  
`Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void CompressRowsQK (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename`  
`Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRows (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t`  
`lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
- `template<class Field >`  
`void DeCompressRowsQK (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const`

- ```
size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t deg, const
size_t nb_blocs)
```
- `template<class Field >`  
`void CompressRowsQA (Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename`  
`Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
  - `template<class Field >`  
`void DeCompressRowsQA (Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const`  
`size_t lda, typename Field::Element_ptr tmp, const size_t ldtmp, const size_t *d, const size_t nb_blocs)`
  - `template<class Field >`  
`size_t LUdivine_construct (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t`  
`N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t`  
`ldx, typename Field::Element_ptr u, const size_t incu, size_t *P, bool computeX, const FFPACK::FFPACK_`  
`_MINPOLY_TAG MinTag, const size_t kg_mc, const size_t kg_mb, const size_t kg_j)`

## 15.21.1 Function Documentation

### 15.21.1.1 LUdivine\_construct() [1/2]

```
size_t LUdivine_construct (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    typename Field::Element_ptr u,
    const size_t incu,
    size_t * P,
    bool computeX,
    const FFPACK_MINPOLY_TAG MinTag = FfpackDense,
    const size_t kg_mc = 0,
    const size_t kg_mb = 0,
    const size_t kg_j = 0 )
```

### 15.21.1.2 GaussJordan()

```
size_t GaussJordan (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    const size_t colbeg,
    const size_t rowbeg,
    const size_t colsize,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.

**Bibliography**

- Algorithm 2.8 of A. Storjohann Thesis 2000,
- Algorithm 11 of Jeannerod C-P., Pernet, C. and Storjohann, A. *Rank-profile revealing Gaussian elimination and the CUP matrix decomposition*, J. of Symbolic Comp., 2013

**Parameters**

|         |         |  |
|---------|---------|--|
|         | $M$     | row dimension of A   |
|         | $N$     | column dimension of A  |
| in, out | $A$     | an m x n matrix  |
|         | $lda$   | leading dimension of A   |
|         | $P$     | row permutation  |
|         | $Q$     | column permutation   |
|         | $LuTag$ | set the base case to a Tile (FfpackGaussJordanTile) or Slab (FfpackGaussJordanSlab) recursive RedEchelon |

where the transformation matrix is stored at the pivot column position

**15.21.1.3 KellerGehrig()**

```
std::list< Polynomial > & KellerGehrig (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda )
```

**15.21.1.4 KGFast()**

```
int KGFast (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * kg_mc,
    size_t * kg_mb,
    size_t * kg_j )
```

**15.21.1.5 KGFast\_generalized()**

```
std::list< Polynomial > & KGFast_generalized (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )
```

**15.21.1.6 fgemv\_kgf()**

```
void fgemv_kgf (
    const Field & F,
    const size_t N,
    typename Field::ConstElement_ptr A,
```

```

const size_t lda,
typename Field::ConstElement_ptr X,
const size_t incX,
typename Field::Element_ptr Y,
const size_t incY,
const size_t kg_mc,
const size_t kg_mb,
const size_t kg_j )

```

### 15.21.1.7 LUKrylov()

```

std::list< Polynomial > & LUKrylov (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr U,
    const size_t ldu,
    RandIter & G )

```

### 15.21.1.8 Danilevski()

```

std::list< Polynomial > & Danilevski (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda )

```

### 15.21.1.9 RandomKrylovPrecond()

```

void RandomKrylovPrecond (
    const PolRing & PR,
    std::list< typename PolRing::Element > & completedFactors,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    size_t & Nb,
    typename PolRing::Domain_t::Element_ptr & B,
    size_t & ldb,
    typename PolRing::Domain_t::RandIter & g,
    const size_t degree = __FFPACK_ARITHPROG_THRESHOLD ) [inline]

```

**Todo** swap to save space ??

**Todo**

**Todo** don't assing K2 c\*noc x N but only mas (c,noc) x N and store each one after the other

**Todo** swap to save space ??

**Todo**

**Todo** don't assing K2 c\*noc x N but only mas (c,noc) x N and store each one after the other

**15.21.1.10 ArithProg()**

```
std::list< typename PolRing::Element > & ArithProg (
    const PolRing & PR,
    std::list< typename PolRing::Element > & frobeniusForm,
    const size_t N,
    typename PolRing::Domain_t::Element_ptr A,
    const size_t lda,
    const size_t degree ) [inline]
```

**15.21.1.11 LUKrylov\_KGFast()**

```
std::list< Polynomial > & LUKrylov_KGFast (
    const Field & F,
    std::list< Polynomial > & charp,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx )
```

**15.21.1.12 MatVecMinPoly()**

```
Polynomial & MatVecMinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr v,
    const size_t incv,
    typename Field::Element_ptr K,
    const size_t ldK,
    size_t * P ) [inline]
```

**15.21.1.13 Hybrid\_KGF\_LUK\_MinPoly()**

```
Polynomial & Hybrid_KGF_LUK_MinPoly (
    const Field & F,
    Polynomial & minP,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    size_t * P,
    const FFPACK_MINPOLY_TAG MinTag = FFPACK::FfpackDense,
    const size_t kg_mc = 0,
    const size_t kg_mb = 0,
    const size_t kg_j = 0 )
```

**15.21.1.14 updateD()**

```
size_t updateD (
    const Field & F,
```



```

size_t * d,
size_t k,
std::vector< std::vector< typename Field::Element > > & minpt )

```

#### 15.21.1.15 newD()

```

size_t newD (
    const Field & F,
    size_t * d,
    bool & KeepOn,
    const size_t l,
    const size_t N,
    typename Field::Element_ptr X,
    const size_t * Q,
    std::vector< std::vector< typename Field::Element > > & minpt )

```

#### 15.21.1.16 CompressRows()

```

void CompressRows (
    Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]

```

#### 15.21.1.17 CompressRowsQK()

```

void CompressRowsQK (
    Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t deg,
    const size_t nb_blocs ) [inline]

```

#### 15.21.1.18 DeCompressRows()

```

void DeCompressRows (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]

```

**15.21.1.19 DeCompressRowsQK()**

```

void DeCompressRowsQK (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t deg,
    const size_t nb_blocs ) [inline]

```

**15.21.1.20 CompressRowsQA()**

```

void CompressRowsQA (
    Field & F,
    const size_t M,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]

```

**15.21.1.21 DeCompressRowsQA()**

```

void DeCompressRowsQA (
    Field & F,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    typename Field::Element_ptr tmp,
    const size_t ldtmp,
    const size_t * d,
    const size_t nb_blocs ) [inline]

```

**15.21.1.22 LUdivine\_construct() [2/2]**

```

size_t LUdivine_construct (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::Element_ptr X,
    const size_t ldx,
    typename Field::Element_ptr u,
    const size_t incu,
    size_t * P,
    bool computeX,
    const FFPACK::FFPACK_MINPOLY_TAG MinTag,
    const size_t kg_mc,

```

```
const size_t kg_mb,  
const size_t kg_j )
```

## 15.22 Givaro Namespace Reference

### Data Structures

- class [ModularBalanced](#)
- class [Montgomery](#)

## 15.23 MKL\_CONFIG Namespace Reference

## 15.24 Reclnt Namespace Reference

### Data Structures

- class [rint](#)
- class [ruint](#)



## Chapter 16

# Data Structure Documentation

### 16.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference

#### Public Types

- typedef [MMHelperAlgo::Winograd value](#)

#### 16.1.1 Member Typedef Documentation

##### 16.1.1.1 value

typedef [MMHelperAlgo::Winograd value](#)

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

### 16.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference

#### Public Types

- typedef [MMHelperAlgo::Classic value](#)

#### 16.2.1 Member Typedef Documentation

##### 16.2.1.1 value

typedef [MMHelperAlgo::Classic value](#)

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

### 16.3 ArbitraryPrecIntTag Struct Reference

Arbitrary precision integers: GMP.

```
#include <field-traits.h>
```

### 16.3.1 Detailed Description

Arbitrary precision integers: GMP.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.4 `AreEqual< X, Y >` Class Template Reference

```
#include <fflas_enum.h>
```

### Static Public Attributes

- static const bool [value](#) = false

### 16.4.1 Field Documentation

#### 16.4.1.1 `value`

```
const bool value = false [static]
```

The documentation for this class was generated from the following file:

- [fflas\\_enum.h](#)

## 16.5 `AreEqual< X, X >` Class Template Reference

```
#include <fflas_enum.h>
```

### Static Public Attributes

- static const bool [value](#) = true

### 16.5.1 Field Documentation

#### 16.5.1.1 `value`

```
const bool value = true [static]
```

The documentation for this class was generated from the following file:

- [fflas\\_enum.h](#)

## 16.6 Argument Struct Reference

```
#include <args-parser.h>
```

### Data Fields

- char [c](#)
- const char \* [example](#)
- const char \* [helpString](#)
- [ArgumentType](#) type
- void \* [data](#)

## 16.6.1 Field Documentation

### 16.6.1.1 c

char c

### 16.6.1.2 example

const char\* example

### 16.6.1.3 helpString

const char\* helpString

### 16.6.1.4 type

[ArgumentType](#) type

### 16.6.1.5 data

void\* data

The documentation for this struct was generated from the following file:

- [args-parser.h](#)

## 16.7 associatedDelayedField< Field > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [Field](#) field
- typedef [Field](#) & [type](#)

### 16.7.1 Member Typedef Documentation

#### 16.7.1.1 field

typedef [Field](#) field

#### 16.7.1.2 type

typedef [Field](#)& [type](#)

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.8 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [FFPACK::RNSInteger< RNS >](#) [field](#)
- typedef [FFPACK::RNSInteger< RNS >](#) [type](#)

### 16.8.1 Member Typedef Documentation

#### 16.8.1.1 field

```
typedef FFPACK::RNSInteger<RNS> field
```

#### 16.8.1.2 type

```
typedef FFPACK::RNSInteger<RNS> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.9 associatedDelayedField< const Givaro::Modular< T, X > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)

### 16.9.1 Member Typedef Documentation

#### 16.9.1.1 field

```
typedef Givaro::ZRing<T> field
```

#### 16.9.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.10 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)



## 16.10.1 Member Typedef Documentation

### 16.10.1.1 field

```
typedef Givaro::ZRing<T> field
```

### 16.10.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.11 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef Givaro::ZRing< T > [field](#)
- typedef Givaro::ZRing< T > [type](#)

## 16.11.1 Member Typedef Documentation

### 16.11.1.1 field

```
typedef Givaro::ZRing<T> field
```

### 16.11.1.2 type

```
typedef Givaro::ZRing<T> type
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.12 Auto Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.13 Bini Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.14 Block Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.15 callLUdivine\_small< Element > Class Template Reference

### Public Member Functions

- `template<class Field >`  
`size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`

### 16.15.1 Member Function Documentation

#### 16.15.1.1 operator>()()

```
size_t operator() (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

The documentation for this class was generated from the following file:

- [ffpack\\_ludivine.inl](#)

## 16.16 callLUdivine\_small< double > Class Reference

### Public Member Functions

- `template<class Field >`  
`size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`

### 16.16.1 Member Function Documentation

#### 16.16.1.1 operator>()()

```
size_t operator() (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

The documentation for this class was generated from the following file:

- [ffpack\\_ludivine.inl](#)

## 16.17 callLUdivine\_small< float > Class Reference

### Public Member Functions

- `template<class Field >`  
`size_t operator() (const Field &F, const FFLAS::FFLAS_DIAG Diag, const FFLAS::FFLAS_TRANSPOSE trans, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFPACK::FFPACK_LU_TAG LuTag)`

### 16.17.1 Member Function Documentation

#### 16.17.1.1 operator>()()

```
size_t operator() (
    const Field & F,
    const FFLAS::FFLAS_DIAG Diag,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    typename Field::Element_ptr A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const FFPACK::FFPACK_LU_TAG LuTag ) [inline]
```

The documentation for this class was generated from the following file:

- [ffpack\\_ludivine.inl](#)

## 16.18 CharpolyFailed Class Reference

```
#include <ffpack.h>
```

The documentation for this class was generated from the following file:

- [ffpack.h](#)

## 16.19 Checker\_Empty< Field > Struct Template Reference

```
#include <checker_empty.h>
```

### Public Member Functions

- `template<typename... Params>`  
`Checker_Empty (Params... parameters)`
- `template<typename... Params>`  
`bool check (Params... parameters)`

### 16.19.1 Constructor & Destructor Documentation

#### 16.19.1.1 Checker\_Empty()

```
Checker_Empty (
    Params... parameters ) [inline]
```

## 16.19.2 Member Function Documentation

### 16.19.2.1 check()

```
bool check (
    Params... parameters ) [inline]
```

The documentation for this struct was generated from the following file:

- [checker\\_empty.h](#)

## 16.20 CheckerImplem\_charpoly< Field, Polynomial > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_charpoly](#) (const [Field](#) &F\_, const size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda\_)
- [CheckerImplem\\_charpoly](#) (typename [Field::RandIter](#) &G, const size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda\_)
- [~CheckerImplem\\_charpoly](#) ()
- bool [check](#) (Polynomial &g)

### 16.20.1 Constructor & Destructor Documentation

#### 16.20.1.1 CheckerImplem\_charpoly() [1/2]

```
CheckerImplem_charpoly (
    const Field & F_,
    const size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda_ ) [inline]
```

#### 16.20.1.2 CheckerImplem\_charpoly() [2/2]

```
CheckerImplem_charpoly (
    typename Field::RandIter & G,
    const size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda_ ) [inline]
```

#### 16.20.1.3 ~CheckerImplem\_charpoly()

```
~CheckerImplem_charpoly ( ) [inline]
```

## 16.20.2 Member Function Documentation

### 16.20.2.1 check()

```
bool check (
    Polynomial & g ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_charpoly.inl](#)

## 16.21 CheckerImplem\_Det< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_Det](#) (const [Field](#) &F\_, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [CheckerImplem\\_Det](#) (typename [Field::RandIter](#) &G, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [~CheckerImplem\\_Det](#) ()
- bool [check](#) (const typename [Field::Element](#) &det, typename [Field::ConstElement\\_ptr](#) LU, size\_t lda, size\_t \*P, size\_t \*Q) const

*check if the Det factorization is correct.*

### 16.21.1 Constructor & Destructor Documentation

#### 16.21.1.1 CheckerImplem\_Det() [1/2]

```
CheckerImplem_Det (
    const Field & F_,
    size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda ) [inline]
```

#### 16.21.1.2 CheckerImplem\_Det() [2/2]

```
CheckerImplem_Det (
    typename Field::RandIter & G,
    size_t n_,
    typename Field::ConstElement\_ptr A,
    size_t lda ) [inline]
```

#### 16.21.1.3 ~CheckerImplem\_Det()

```
~CheckerImplem_Det ( ) [inline]
```

### 16.21.2 Member Function Documentation

#### 16.21.2.1 check()

```
bool check (
    const typename Field::Element & det,
    typename Field::ConstElement\_ptr LU,
    size_t lda,
    size_t * P,
    size_t * Q ) const [inline]
```

check if the Det factorization is correct.

Needs matrix in LU form

#### Parameters

|                   |             |
|-------------------|-------------|
| <i>LU,storage</i> | for L and U |
|-------------------|-------------|

## Parameters

|            |  |
|------------|--|
| <i>det</i> |  |
| <i>P</i>   |  |
| <i>Q</i>   |  |

The documentation for this class was generated from the following file:

- [checker\\_det.inl](#)

## 16.22 CheckerImplem\_fgemm< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_fgemm](#) (const [Field](#) &F\_, const size\_t m\_, const size\_t n\_, const size\_t k\_, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc\_)
- [CheckerImplem\\_fgemm](#) (typename [Field::RandIter](#) &G, const size\_t m\_, const size\_t n\_, const size\_t k\_, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc\_)
- [~CheckerImplem\\_fgemm](#) ()
- bool [check](#) (const [FFLAS::FFLAS\\_TRANSPOSE](#) ta, const [FFLAS::FFLAS\\_TRANSPOSE](#) tb, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::ConstElement\\_ptr](#) C)

### 16.22.1 Constructor & Destructor Documentation

#### 16.22.1.1 CheckerImplem\_fgemm() [1/2]

```
CheckerImplem_fgemm (
    const Field & F_,
    const size_t m_,
    const size_t n_,
    const size_t k_,
    const typename Field::Element beta,
    typename Field::Element\_ptr C,
    const size_t ldc_ ) [inline]
```

#### 16.22.1.2 CheckerImplem\_fgemm() [2/2]

```
CheckerImplem_fgemm (
    typename Field::RandIter & G,
    const size_t m_,
    const size_t n_,
    const size_t k_,
    const typename Field::Element beta,
    typename Field::Element\_ptr C,
    const size_t ldc_ ) [inline]
```

#### 16.22.1.3 ~CheckerImplem\_fgemm()

```
~CheckerImplem_fgemm ( ) [inline]
```

### 16.22.2 Member Function Documentation

### 16.22.2.1 check()

```
bool check (
    const FFLAS::FFLAS_TRANSPOSE ta,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const typename Field::Element alpha,
    typename Field::ConstElement_ptr A,
    const size_t lda,
    typename Field::ConstElement_ptr B,
    const size_t ldb,
    typename Field::ConstElement_ptr C ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_fgemm.inl](#)

## 16.23 CheckerImplem\_ftrsm< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_ftrsm](#) (const [Field](#) &F\_, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)
- [CheckerImplem\\_ftrsm](#) (typename [Field::RandIter](#) &G, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)
- [~CheckerImplem\\_ftrsm](#) ()
- bool [check](#) (const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_UPLO](#) uplo, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda, const typename [Field::ConstElement\\_ptr](#) X, size\_t ldx)

### 16.23.1 Constructor & Destructor Documentation

#### 16.23.1.1 CheckerImplem\_ftrsm() [1/2]

```
CheckerImplem_ftrsm (
    const Field & F_,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr B,
    const size_t ldb ) [inline]
```

#### 16.23.1.2 CheckerImplem\_ftrsm() [2/2]

```
CheckerImplem_ftrsm (
    typename Field::RandIter & G,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const typename Field::ConstElement_ptr B,
    const size_t ldb ) [inline]
```

#### 16.23.1.3 ~CheckerImplem\_ftrsm()

```
~CheckerImplem_ftrsm ( ) [inline]
```

## 16.23.2 Member Function Documentation

### 16.23.2.1 check()

```
bool check (
    const FFLAS::FFLAS_SIDE side,
    const FFLAS::FFLAS_UPLO uplo,
    const FFLAS::FFLAS_TRANSPOSE trans,
    const FFLAS::FFLAS_DIAG diag,
    const size_t m,
    const size_t n,
    typename Field::Element_ptr A,
    size_t lda,
    const typename Field::ConstElement_ptr X,
    size_t ldx ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_ftsm.inl](#)

## 16.24 CheckerImplem\_invert< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_invert](#) (const [Field](#) &F\_, const size\_t m\_, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda\_)
- [CheckerImplem\\_invert](#) (typename [Field::RandIter](#) &G, const size\_t m\_, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda\_)
- [~CheckerImplem\\_invert](#) ()
- bool [check](#) (typename [Field::ConstElement\\_ptr](#) A, int nullity)

### 16.24.1 Constructor & Destructor Documentation

#### 16.24.1.1 CheckerImplem\_invert() [1/2]

```
CheckerImplem_invert (
    const Field & F_,
    const size_t m_,
    typename Field::ConstElement_ptr A,
    const size_t lda_ ) [inline]
```

#### 16.24.1.2 CheckerImplem\_invert() [2/2]

```
CheckerImplem_invert (
    typename Field::RandIter & G,
    const size_t m_,
    typename Field::ConstElement_ptr A,
    const size_t lda_ ) [inline]
```

#### 16.24.1.3 ~CheckerImplem\_invert()

```
~CheckerImplem_invert ( ) [inline]
```



## 16.24.2 Member Function Documentation

### 16.24.2.1 check()

```
bool check (
    typename Field::ConstElement_ptr A,
    int nullity ) [inline]
```

The documentation for this class was generated from the following file:

- [checker\\_invert.inl](#)

## 16.25 CheckerImplem\_PLUQ< Field > Class Template Reference

### Public Member Functions

- [CheckerImplem\\_PLUQ](#) (const [Field](#) &F\_, size\_t m\_, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [CheckerImplem\\_PLUQ](#) (typename [Field::RandIter](#) &G, size\_t m\_, size\_t n\_, typename [Field::ConstElement\\_ptr](#) A, size\_t lda)
- [~CheckerImplem\\_PLUQ](#) ()
- bool [check](#) (typename [Field::ConstElement\\_ptr](#) A, size\_t lda, const [FFLAS::FFLAS\\_DIAG](#) Diag, size\_t r, size\_t \*P, size\_t \*Q) const  
*check if the PLUQ factorization is correct.*

### 16.25.1 Constructor & Destructor Documentation

#### 16.25.1.1 CheckerImplem\_PLUQ() [1/2]

```
CheckerImplem_PLUQ (
    const Field & F_,
    size_t m_,
    size_t n_,
    typename Field::ConstElement_ptr A,
    size_t lda ) [inline]
```

#### 16.25.1.2 CheckerImplem\_PLUQ() [2/2]

```
CheckerImplem_PLUQ (
    typename Field::RandIter & G,
    size_t m_,
    size_t n_,
    typename Field::ConstElement_ptr A,
    size_t lda ) [inline]
```

#### 16.25.1.3 ~CheckerImplem\_PLUQ()

```
~CheckerImplem_PLUQ ( ) [inline]
```

## 16.25.2 Member Function Documentation

### 16.25.2.1 check()

```
bool check (
    typename Field::ConstElement_ptr A,
    size_t lda,
    const FFLAS::FFLAS_DIAG Diag,
    size_t r,
    size_t * P,
    size_t * Q ) const [inline]
```

check if the PLUQ factorization is correct.

Returns true if  $w - P(L(U(Q.v))) == 0$

#### Parameters

|          |  |
|----------|--|
| <i>A</i> |  |
| <i>r</i> |  |
| <i>P</i> |  |
| <i>Q</i> |  |

The documentation for this class was generated from the following file:

- [checker\\_pluq.inl](#)

## 16.26 Classic Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.27 Column Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.28 CompactElement< Element > Struct Template Reference

### Public Types

- typedef Element [type](#)

### 16.28.1 Member Typedef Documentation

#### 16.28.1.1 type

```
typedef Element type
```

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.29 CompactElement< double > Struct Reference

### Public Types

- typedef [int32\\_t](#) [type](#)

## 16.29.1 Member Typedef Documentation

### 16.29.1.1 type

typedef [int32\\_t](#) type

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.30 CompactElement< float > Struct Reference

### Public Types

- typedef [int16\\_t](#) type

## 16.30.1 Member Typedef Documentation

### 16.30.1.1 type

typedef [int16\\_t](#) type

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.31 CompactElement< int16\_t > Struct Reference

### Public Types

- typedef [int8\\_t](#) type

## 16.31.1 Member Typedef Documentation

### 16.31.1.1 type

typedef [int8\\_t](#) type

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.32 CompactElement< int32\_t > Struct Reference

### Public Types

- typedef [int16\\_t](#) type

## 16.32.1 Member Typedef Documentation

### 16.32.1.1 type

typedef [int16\\_t](#) type

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.33 CompactElement< int64\_t > Struct Reference

### Public Types

- typedef [int32\\_t](#) [type](#)

### 16.33.1 Member Typedef Documentation

#### 16.33.1.1 type

typedef [int32\\_t](#) [type](#)

The documentation for this struct was generated from the following file:

- [test-io.C](#)

## 16.34 compatible\_data\_type< Field > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = true

### 16.34.1 Field Documentation

#### 16.34.1.1 value

constexpr bool value = true [static], [constexpr]

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.35 compatible\_data\_type< Givaro::ZRing< double > > Struct Reference

### Static Public Attributes

- static constexpr bool [value](#) = false

### 16.35.1 Field Documentation

#### 16.35.1.1 value

constexpr bool value = false [static], [constexpr]

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.36 compatible\_data\_type< Givaro::ZRing< float > > Struct Reference

### Static Public Attributes

- static constexpr bool [value](#) = false

## 16.36.1 Field Documentation

### 16.36.1.1 value

constexpr bool value = false [static], [constexpr]

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.37 Compose< H1, H2 > Struct Template Reference

### Public Member Functions

- [Compose](#) ()
- [Compose](#) (const [Compose](#) &other)
- [Compose](#) (const [Sequential](#) &S)
- [Compose](#) (size\_t th1, size\_t th2)
- [Compose](#) (const H1 &o1, const H2 &o2)
- H1 [first\\_component](#) () const
- H2 [second\\_component](#) () const

### Friends

- std::ostream & [operator<<](#) (std::ostream &o, const [Compose](#) &c)

## 16.37.1 Constructor & Destructor Documentation

### 16.37.1.1 Compose() [1/5]

[Compose](#) ( ) [inline]

### 16.37.1.2 Compose() [2/5]

[Compose](#) (   
 const [Compose](#)< H1, H2 > & other ) [inline]

### 16.37.1.3 Compose() [3/5]

[Compose](#) (   
 const [Sequential](#) & S ) [inline]

### 16.37.1.4 Compose() [4/5]

[Compose](#) (   
 size\_t th1,   
 size\_t th2 ) [inline]

### 16.37.1.5 Compose() [5/5]

[Compose](#) (   
 const H1 & o1,   
 const H2 & o2 ) [inline]

## 16.37.2 Member Function Documentation

### 16.37.2.1 first\_component()

```
H1 first_component ( ) const [inline]
```

### 16.37.2.2 second\_component()

```
H2 second_component ( ) const [inline]
```

## 16.37.3 Friends And Related Function Documentation

### 16.37.3.1 operator<<

```
std::ostream & operator<< (
    std::ostream & o,
    const Compose< H1, H2 > & c ) [friend]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.38 Const\_int\_t< n > Class Template Reference

```
#include <instrset.h>
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

## 16.39 Const\_uint\_t< n > Class Template Reference

```
#include <instrset.h>
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

## 16.40 Simd128\_impl< true, true, false, 2 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t t](#) [vect\_size]

### 16.40.1 Field Documentation

#### 16.40.1.1 v

```
vect\_t v
```

### 16.40.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128\\_int16.inl](#)

## 16.41 Simd128\_impl< true, true, false, 4 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_tt\[vect\\_size\]](#)

### 16.41.1 Field Documentation

#### 16.41.1.1 v

`vect_t v`

#### 16.41.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128\\_int32.inl](#)

## 16.42 Simd128\_impl< true, true, false, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_tt\[vect\\_size\]](#)

### 16.42.1 Field Documentation

#### 16.42.1.1 v

`vect_t v`

#### 16.42.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- [simd128\\_int64.inl](#)

## 16.43 Simd128\_impl< true, true, true, 2 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_tt\[vect\\_size\]](#)

### 16.43.1 Field Documentation

#### 16.43.1.1 v

`vect_t` v

#### 16.43.1.2 t

`scalar_t` t[vect\_size]

The documentation for this union was generated from the following file:

- [simd128\\_int16.inl](#)

## 16.44 Simd128\_impl< true, true, true, 4 >::Converter Union Reference

### Data Fields

- `vect_t` v
- `scalar_t` t[vect\_size]

### 16.44.1 Field Documentation

#### 16.44.1.1 v

`vect_t` v

#### 16.44.1.2 t

`scalar_t` t[vect\_size]

The documentation for this union was generated from the following file:

- [simd128\\_int32.inl](#)

## 16.45 Simd128\_impl< true, true, true, 8 >::Converter Union Reference

### Data Fields

- `vect_t` v
- `scalar_t` t[vect\_size]

### 16.45.1 Field Documentation

#### 16.45.1.1 v

`vect_t` v

#### 16.45.1.2 t

`scalar_t` t[vect\_size]

The documentation for this union was generated from the following file:

- [simd128\\_int64.inl](#)



## 16.46 Simd256\_impl< true, true, false, 2 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_tt](#) [vect\_size]

#### 16.46.1 Field Documentation

##### 16.46.1.1 v

[vect\\_t](#) v

##### 16.46.1.2 t

[scalar\\_t](#) t [vect\_size]

The documentation for this union was generated from the following file:

- [simd256\\_int16.inl](#)

## 16.47 Simd256\_impl< true, true, false, 4 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_tt](#) [vect\_size]

#### 16.47.1 Field Documentation

##### 16.47.1.1 v

[vect\\_t](#) v

##### 16.47.1.2 t

[scalar\\_t](#) t

The documentation for this union was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

## 16.48 Simd256\_impl< true, true, false, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_tt](#) [vect\_size]

#### 16.48.1 Field Documentation

**16.48.1.1 v**

`vect_t` v

**16.48.1.2 t**

`scalar_t` t[vect\_size]

The documentation for this union was generated from the following file:

- [simd256\\_int64.inl](#)

**16.49 Simd256\_impl< true, true, true, 2 >::Converter Union Reference****Data Fields**

- `vect_t` v
- `scalar_t` t[vect\_size]

**16.49.1 Field Documentation****16.49.1.1 v**

`vect_t` v

**16.49.1.2 t**

`scalar_t` t[vect\_size]

The documentation for this union was generated from the following file:

- [simd256\\_int16.inl](#)

**16.50 Simd256\_impl< true, true, true, 4 >::Converter Union Reference****Data Fields**

- `vect_t` v
- `scalar_t` t[vect\_size]

**16.50.1 Field Documentation****16.50.1.1 v**

`vect_t` v

**16.50.1.2 t**

`scalar_t` t

The documentation for this union was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

## 16.51 Simd256\_impl< true, true, true, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_tt](#) [vect\_size]

#### 16.51.1 Field Documentation

##### 16.51.1.1 v

[vect\\_t](#) v

##### 16.51.1.2 t

[scalar\\_t](#) t [vect\_size]

The documentation for this union was generated from the following file:

- [simd256\\_int64.inl](#)

## 16.52 Simd512\_impl< true, true, false, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_t](#) t [vect\_size]

#### 16.52.1 Field Documentation

##### 16.52.1.1 v

[vect\\_t](#) v

##### 16.52.1.2 t

[scalar\\_t](#) t [vect\_size]

The documentation for this union was generated from the following file:

- [simd512\\_int64.inl](#)

## 16.53 Simd512\_impl< true, true, true, 8 >::Converter Union Reference

### Data Fields

- [vect\\_t v](#)
- [scalar\\_tt](#) [vect\_size]

#### 16.53.1 Field Documentation

**16.53.1.1 v**

`vect_t` [v](#)

**16.53.1.2 t**

`scalar_t` [t](#)[`vect_size`]

The documentation for this union was generated from the following file:

- [simd512\\_int64.inl](#)

**16.54 ConvertTo< T > Struct Template Reference**

Force conversion to appropriate element type of ElementCategory T.

`#include <field-traits.h>`

**16.54.1 Detailed Description**

`template<class T>`

`struct FFLAS::ModeCategories::ConvertTo< T >`

Force conversion to appropriate element type of ElementCategory T.

e.g.

- `ConvertTo<ElementCategories::MachineFloatTag>` tries conversion of `Modular<int>` to `Modular<double>`
- `ConvertTo<ElementCategories::FixedPrecIntTag>` tries conversion of `Modular<Integer>` to `Modular<RecInt<K>>`
- `ConvertTo<ElementCategories::ArbitraryPrecIntTag>` tries conversion of `Modular<Integer>` to `RNSInteger`

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.55 Coo< ValT, IdxT > Struct Template Reference****Public Types**

- using `Self` = `Coo< ValT, IdxT >`

**Public Member Functions**

- `Coo` (`ValT` v, `IdxT` r, `IdxT` c)
- `Coo` ()=default
- `Coo` (const `Self` &)=default
- `Coo` (`Self` &&)=default
- `Self` & `operator=` (const `Self` &)=default
- `Self` & `operator=` (`Self` &&)=default

**Data Fields**

- `ValT` `val` = 0
- `IdxT` `row` = 0
- `IdxT` `col` = 0

**16.55.1 Member Typedef Documentation**

### 16.55.1.1 Self

```
using Self = Coo<ValT, IdxT>
```

## 16.55.2 Constructor & Destructor Documentation

### 16.55.2.1 `Coo()` [1/4]

```
Coo (
    ValT v,
    IdxT r,
    IdxT c ) [inline]
```

### 16.55.2.2 `Coo()` [2/4]

```
Coo ( ) [default]
```

### 16.55.2.3 `Coo()` [3/4]

```
Coo (
    const Self & ) [default]
```

### 16.55.2.4 `Coo()` [4/4]

```
Coo (
    Self && ) [default]
```

## 16.55.3 Member Function Documentation

### 16.55.3.1 `operator=()` [1/2]

```
Self & operator= (
    const Self & ) [default]
```

### 16.55.3.2 `operator=()` [2/2]

```
Self & operator= (
    Self && ) [default]
```

## 16.55.4 Field Documentation

### 16.55.4.1 `val`

```
ValT val = 0
```

### 16.55.4.2 `row`

```
IdxT row = 0
```

### 16.55.4.3 col

```
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- [csr\\_hyb\\_utils.inl](#)

## 16.56 **Coo**< **Field** > Struct Template Reference

```
#include <read_sparse.h>
```

### Public Member Functions

- **Coo** ()=default
- **Coo** (typename [Field::Element](#) v, [index\\_t](#) r, [index\\_t](#) c)
- **Coo** (const [Self](#) &)=default
- **Coo** ([Self](#) &&)=default
- [Self](#) & **operator=** (const [Self](#) &)=default
- [Self](#) & **operator=** ([Self](#) &&)=default

### Data Fields

- [Field::Element](#) val = 0
- [index\\_t](#) col = 0
- [index\\_t](#) row = 0
- bool [deleted](#) = false

## 16.56.1 Constructor & Destructor Documentation

### 16.56.1.1 **Coo**() [1/4]

```
Coo ( ) [default]
```

### 16.56.1.2 **Coo**() [2/4]

```
Coo (
    typename Field::Element v,
    index\_t r,
    index\_t c ) [inline]
```

### 16.56.1.3 **Coo**() [3/4]

```
Coo (
    const Self & ) [default]
```

### 16.56.1.4 **Coo**() [4/4]

```
Coo (
    Self && ) [default]
```

## 16.56.2 Member Function Documentation

**16.56.2.1 `operator=()` [1/2]**

```
Self & operator= (
    const Self & ) [default]
```

**16.56.2.2 `operator=()` [2/2]**

```
Self & operator= (
    Self && ) [default]
```

**16.56.3 Field Documentation****16.56.3.1 `val`**

```
Field::Element val = 0
```

**16.56.3.2 `col`**

```
index_t col = 0
```

**16.56.3.3 `row`**

```
index_t row = 0
```

**16.56.3.4 `deleted`**

```
bool deleted = false
```

The documentation for this struct was generated from the following file:

- [read\\_sparse.h](#)

**16.57 `Coo< ValT, IdxT >` Struct Template Reference****Public Types**

- using `Self` = `Coo< ValT, IdxT >`

**Public Member Functions**

- `Coo` (`ValT` v, `IdxT` r, `IdxT` c)
- `Coo` ()=default
- `Coo` (const `Self` &)=default
- `Coo` (`Self` &&)=default
- `Self` & `operator=` (const `Self` &)=default
- `Self` & `operator=` (`Self` &&)=default

**Data Fields**

- `ValT` `val` = 0
- `IdxT` `row` = 0
- `IdxT` `col` = 0

## 16.57.1 Member Typedef Documentation

### 16.57.1.1 Self

```
using Self = Coo<ValT, IdxT>
```

## 16.57.2 Constructor & Destructor Documentation

### 16.57.2.1 Coo() [1/4]

```
Coo (
    ValT v,
    IdxT r,
    IdxT c ) [inline]
```

### 16.57.2.2 Coo() [2/4]

```
Coo ( ) [default]
```

### 16.57.2.3 Coo() [3/4]

```
Coo (
    const Self & ) [default]
```

### 16.57.2.4 Coo() [4/4]

```
Coo (
    Self && ) [default]
```

## 16.57.3 Member Function Documentation

### 16.57.3.1 operator=() [1/2]

```
Self & operator= (
    const Self & ) [default]
```

### 16.57.3.2 operator=() [2/2]

```
Self & operator= (
    Self && ) [default]
```

## 16.57.4 Field Documentation

### 16.57.4.1 val

```
ValT val = 0
```



**16.57.4.2 row**

```
IdxT row = 0
```

**16.57.4.3 col**

```
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- [sell\\_utils.inl](#)

**16.58 CooMat< Field > Struct Template Reference**

```
#include <fflas_sparse.h>
```

**Data Fields**

- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO, int16\\_t > \\* \\_coo16](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO, int32\\_t > \\* \\_coo32](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO, int64\\_t > \\* \\_coo64](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO\\_ZO, int16\\_t > \\* \\_coo16\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO\\_ZO, int32\\_t > \\* \\_coo32\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::COO\\_ZO, int64\\_t > \\* \\_coo64\\_zo](#) = nullptr

**16.58.1 Field Documentation****16.58.1.1 \_coo16**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO, int16_t>* _coo16 = nullptr
```

**16.58.1.2 \_coo32**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO, int32_t>* _coo32 = nullptr
```

**16.58.1.3 \_coo64**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO, int64_t>* _coo64 = nullptr
```

**16.58.1.4 \_coo16\_zo**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int16_t>* _coo16_zo = nullptr
```

**16.58.1.5 \_coo32\_zo**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int32_t>* _coo32_zo = nullptr
```

**16.58.1.6 \_coo64\_zo**

```
FFLAS::Sparse<Field, SparseMatrix_t::COO_ZO, int64_t>* _coo64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.59 CsrMat< Field > Struct Template Reference

```
#include <fflas_sparse.h>
```

### Data Fields

- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR, int16\\_t > \\* \\_csr16](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR, int32\\_t > \\* \\_csr32](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR, int64\\_t > \\* \\_csr64](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR\\_ZO, int16\\_t > \\* \\_csr16\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR\\_ZO, int32\\_t > \\* \\_csr32\\_zo](#) = nullptr
- [FFLAS::Sparse< Field, SparseMatrix\\_t::CSR\\_ZO, int64\\_t > \\* \\_csr64\\_zo](#) = nullptr

### 16.59.1 Field Documentation

#### 16.59.1.1 \_csr16

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int16_t>* _csr16 = nullptr
```

#### 16.59.1.2 \_csr32

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int32_t>* _csr32 = nullptr
```

#### 16.59.1.3 \_csr64

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR, int64_t>* _csr64 = nullptr
```

#### 16.59.1.4 \_csr16\_zo

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int16_t>* _csr16_zo = nullptr
```

#### 16.59.1.5 \_csr32\_zo

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int32_t>* _csr32_zo = nullptr
```

#### 16.59.1.6 \_csr64\_zo

```
FFLAS::Sparse<Field, SparseMatrix_t::CSR_ZO, int64_t>* _csr64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.60 DefaultBoundedTag Struct Reference

Use standard field operations, but keeps track of bounds on input and output.

```
#include <field-traits.h>
```

### 16.60.1 Detailed Description

Use standard field operations, but keeps track of bounds on input and output.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.61 DefaultTag Struct Reference

No specific mode of action: use standard field operations.

```
#include <field-traits.h>
```

### 16.61.1 Detailed Description

No specific mode of action: use standard field operations.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.62 DelayedTag Struct Reference

Performs field operations with delayed mod reductions. Ensures result is reduced.

```
#include <field-traits.h>
```

### 16.62.1 Detailed Description

Performs field operations with delayed mod reductions. Ensures result is reduced.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.63 ElementTraits< Element > Struct Template Reference

[ElementTraits](#).

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::GenericTag](#) value

### 16.63.1 Detailed Description

```
template<class Element>
```

```
struct FFLAS::ElementTraits< Element >
```

[ElementTraits](#).

### 16.63.2 Member Typedef Documentation

#### 16.63.2.1 value

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.64 ElementTraits< double > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineFloatTag](#) value

## 16.64.1 Member Typedef Documentation

### 16.64.1.1 value

typedef [ElementCategories::MachineFloatTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.65 ElementTraits< FFPACK::rns\_double\_elt > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::RNSElementTag](#) value

## 16.65.1 Member Typedef Documentation

### 16.65.1.1 value

typedef [ElementCategories::RNSElementTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.66 ElementTraits< float > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineFloatTag](#) value

## 16.66.1 Member Typedef Documentation

### 16.66.1.1 value

typedef [ElementCategories::MachineFloatTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.67 ElementTraits< Givaro::Integer > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::ArbitraryPrecIntTag](#) value

## 16.67.1 Member Typedef Documentation

### 16.67.1.1 value

typedef [ElementCategories::ArbitraryPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.68 ElementTraits< int16\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.68.1 Member Typedef Documentation

### 16.68.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.69 ElementTraits< int32\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.69.1 Member Typedef Documentation

### 16.69.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.70 ElementTraits< int64\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.70.1 Member Typedef Documentation

### 16.70.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.71 ElementTraits< int8\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.71.1 Member Typedef Documentation

### 16.71.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.72 ElementTraits< RecInt::rint< K > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

## 16.72.1 Member Typedef Documentation

### 16.72.1.1 value

typedef [ElementCategories::FixedPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.73 ElementTraits< RecInt::rmint< K, MG > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

### 16.73.1 Member Typedef Documentation

#### 16.73.1.1 value

typedef [ElementCategories::FixedPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.74 ElementTraits< RecInt::ruint< K > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::FixedPrecIntTag](#) value

### 16.74.1 Member Typedef Documentation

#### 16.74.1.1 value

typedef [ElementCategories::FixedPrecIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.75 ElementTraits< uint16\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

### 16.75.1 Member Typedef Documentation

#### 16.75.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.76 ElementTraits< uint32\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.76.1 Member Typedef Documentation

### 16.76.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.77 ElementTraits< uint64\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.77.1 Member Typedef Documentation

### 16.77.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.78 ElementTraits< uint8\_t > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ElementCategories::MachineIntTag](#) value

## 16.78.1 Member Typedef Documentation

### 16.78.1.1 value

typedef [ElementCategories::MachineIntTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.79 EIMat< Field > Struct Template Reference

```
#include <fflas_sparse.h>
```



## Data Fields

- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#), [int16\\_t](#) > \* [\\_ell16](#) = nullptr
- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#), [int32\\_t](#) > \* [\\_ell32](#) = nullptr
- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL](#), [int64\\_t](#) > \* [\\_ell64](#) = nullptr
- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_ZO](#), [int16\\_t](#) > \* [\\_ell16\\_zo](#) = nullptr
- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_ZO](#), [int32\\_t](#) > \* [\\_ell32\\_zo](#) = nullptr
- [FFLAS::Sparse](#)< [Field](#), [SparseMatrix\\_t::ELL\\_ZO](#), [int64\\_t](#) > \* [\\_ell64\\_zo](#) = nullptr

## 16.79.1 Field Documentation

### 16.79.1.1 [\\_ell16](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int16_t>* _ell16 = nullptr
```

### 16.79.1.2 [\\_ell32](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int32_t>* _ell32 = nullptr
```

### 16.79.1.3 [\\_ell64](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL, int64_t>* _ell64 = nullptr
```

### 16.79.1.4 [\\_ell16\\_zo](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int16_t>* _ell16_zo = nullptr
```

### 16.79.1.5 [\\_ell32\\_zo](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int32_t>* _ell32_zo = nullptr
```

### 16.79.1.6 [\\_ell64\\_zo](#)

```
FFLAS::Sparse<Field, SparseMatrix_t::ELL_ZO, int64_t>* _ell64_zo = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.80 Failure Class Reference

A precondition failed.

```
#include <debug.h>
```

## Public Member Functions

- [Failure](#) ()
- void [operator\(\)](#) (const char \*function, int line, const char \*check)
- void [operator\(\)](#) (const char \*function, const char \*file, int line, const char \*check)
- void [setErrorStream](#) (std::ostream &stream)
- std::ostream & [print](#) (std::ostream &o) const

## Protected Attributes

- `std::ostream * _errorStream`

### 16.80.1 Detailed Description

A precondition failed.

The `throw` mechanism is usually used here as in

```
if (!check)
    failure(__func__, __LINE__, "this check just failed");
```

The parameters of the constructor help debugging.

### 16.80.2 Constructor & Destructor Documentation

#### 16.80.2.1 Failure()

```
Failure ( ) [inline]
```

### 16.80.3 Member Function Documentation

#### 16.80.3.1 operator>() [1/2]

```
void operator() (
    const char * function,
    int line,
    const char * check ) [inline]
```

A precondition failed.

##### Parameters

|                 |   |
|-----------------|---|
| <i>function</i> | usually <code>__func__</code> , the function that threw the error |
| <i>line</i>     | usually <code>__LINE__</code> , the line where it happened        |
| <i>check</i>    | a string telling what failed.                                     |

#### 16.80.3.2 operator>() [2/2]

```
void operator() (
    const char * function,
    const char * file,
    int line,
    const char * check ) [inline]
```

A precondition failed. The parameter help debugging. This is not much different from the previous except we can digg faster in the file where the exception was triggered.

##### Parameters

|                 |   |
|-----------------|---|
| <i>function</i> | usually <code>__func__</code> , the function that threw the error |
| <i>file</i>     | usually <code>__FILE__</code> , the file where this function is   |
| <i>line</i>     | usually <code>__LINE__</code> , the line where it happened        |
| <i>check</i>    | a string telling what failed.                                     |

**16.80.3.3 setErrorMessage()**

```
void setErrorMessage (
    std::ostream & stream )
```

**16.80.3.4 print()**

```
std::ostream & print (
    std::ostream & o ) const [inline]
```

overload the virtual print of LinboxError.

**Parameters**

|          |               |
|----------|---------------|
| <i>o</i> | output stream |
|----------|---------------|

**16.80.4 Field Documentation****16.80.4.1 \_errorMessage**

```
std::ostream* _errorMessage [protected]
```

The documentation for this class was generated from the following file:

- [debug.h](#)

**16.81 FailureCharpolyCheck Class Reference**

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

**16.82 FailureDetCheck Class Reference**

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

**16.83 FailureFgemmCheck Class Reference**

```
#include <checkers_fflas.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_fflas.h](#)

**16.84 FailureInvertCheck Class Reference**

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

## 16.85 FailurePLUQCheck Class Reference

```
#include <checkers_ffpack.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_ffpack.h](#)

## 16.86 FailureTrsmCheck Class Reference

```
#include <checkers_fflas.h>
```

The documentation for this class was generated from the following file:

- [checkers\\_fflas.h](#)

## 16.87 FieldSimd< \_Field > Class Template Reference

### Public Types

- using [Field](#) = [\\_Field](#)
- using [Element](#) = typename [Field::Element](#)
- using [simd](#) = [Simd](#)< typename [\\_Field::Element](#) >
- using [vect\\_t](#) = typename [simd::vect\\_t](#)
- using [scalar\\_t](#) = typename [simd::scalar\\_t](#)

### Public Member Functions

- [FieldSimd](#) (const [Field](#) &f)
- [FieldSimd](#) (const [Self](#) &)=default
- [FieldSimd](#) ([Self](#) &&)=default
- [Self](#) & operator= (const [Self](#) &)=default
- [Self](#) & operator= ([Self](#) &&)=default
- [INLINE vect\\_t init](#) ([vect\\_t](#) &x, const [vect\\_t](#) a) const
- [INLINE vect\\_t init](#) (const [vect\\_t](#) a) const
- [INLINE vect\\_t add](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t add](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t addin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t add\\_r](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t add\\_r](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t addin\\_r](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t sub](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t sub](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- [INLINE vect\\_t subin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t sub\\_r](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t sub\\_r](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t subin\\_r](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t zero](#) ([vect\\_t](#) &x) const
- [INLINE vect\\_t zero](#) () const
- [INLINE vect\\_t mod](#) ([vect\\_t](#) &c) const
- [INLINE vect\\_t mul](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mul](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mulin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mul\\_r](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t mul\\_r](#) (const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t axpy](#) ([vect\\_t](#) &r, const [vect\\_t](#) a, const [vect\\_t](#) b, const [vect\\_t](#) c) const
- [INLINE vect\\_t axpy](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b) const
- [INLINE vect\\_t axpyin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b) const

- `INLINE vect_t axpy_r (vect_t &r, const vect_t a, const vect_t b, const vect_t c) const`
- `INLINE vect_t axpy_r (const vect_t c, const vect_t a, const vect_t b) const`
- `INLINE vect_t axpyin_r (vect_t &c, const vect_t a, const vect_t b) const`
- `INLINE vect_t maxpy (vect_t &r, const vect_t a, const vect_t b, const vect_t c) const`
- `INLINE vect_t maxpy (const vect_t c, const vect_t a, const vect_t b) const`
- `INLINE vect_t maxpyin (vect_t &c, const vect_t a, const vect_t b) const`

## Static Public Attributes

- static const constexpr size\_t `vect_size` = `simd::vect_size`
- static const constexpr size\_t `alignment` = `simd::alignment`

## 16.87.1 Member Typedef Documentation

### 16.87.1.1 Field

using `Field` = `_Field`

### 16.87.1.2 Element

using `Element` = `typename Field::Element`

### 16.87.1.3 simd

using `simd` = `Simd<typename _Field::Element>`

### 16.87.1.4 vect\_t

using `vect_t` = `typename simd::vect_t`

### 16.87.1.5 scalar\_t

using `scalar_t` = `typename simd::scalar_t`

## 16.87.2 Constructor & Destructor Documentation

### 16.87.2.1 FieldSimd() [1/3]

```
FieldSimd (
    const Field & f ) [inline]
```

### 16.87.2.2 FieldSimd() [2/3]

```
FieldSimd (
    const Self & ) [default]
```

### 16.87.2.3 FieldSimd() [3/3]

```
FieldSimd (
    Self && ) [default]
```

## 16.87.3 Member Function Documentation

### 16.87.3.1 operator=() [1/2]

```
Self & operator= (
    const Self & ) [default]
```

### 16.87.3.2 operator=() [2/2]

```
Self & operator= (
    Self && ) [default]
```

### 16.87.3.3 init() [1/2]

```
INLINE vect_t init (
    vect_t & x,
    const vect_t a ) const [inline]
```

### 16.87.3.4 init() [2/2]

```
INLINE vect_t init (
    const vect_t a ) const [inline]
```

### 16.87.3.5 add() [1/2]

```
INLINE vect_t add (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline]
```

### 16.87.3.6 add() [2/2]

```
INLINE vect_t add (
    const vect_t a,
    const vect_t b ) [inline]
```

### 16.87.3.7 addin()

```
INLINE vect_t addin (
    vect_t & a,
    const vect_t b ) const [inline]
```

### 16.87.3.8 add\_r() [1/2]

```
INLINE vect_t add_r (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.87.3.9 add\_r() [2/2]**

```
INLINE vect_t add_r (  
    const vect_t a,  
    const vect_t b ) const [inline]
```

**16.87.3.10 addin\_r()**

```
INLINE vect_t addin_r (  
    vect_t & a,  
    const vect_t b ) const [inline]
```

**16.87.3.11 sub() [1/2]**

```
INLINE vect_t sub (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline]
```

**16.87.3.12 sub() [2/2]**

```
INLINE vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline]
```

**16.87.3.13 subin()**

```
INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) const [inline]
```

**16.87.3.14 sub\_r() [1/2]**

```
INLINE vect_t sub_r (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

**16.87.3.15 sub\_r() [2/2]**

```
INLINE vect_t sub_r (  
    const vect_t a,  
    const vect_t b ) const [inline]
```

**16.87.3.16 subin\_r()**

```
INLINE vect_t subin_r (  
    vect_t & a,  
    const vect_t b ) const [inline]
```

**16.87.3.17 zero() [1/2]**

```
INLINE vect_t zero (
    vect_t & x ) const [inline]
```

**16.87.3.18 zero() [2/2]**

```
INLINE vect_t zero ( ) const [inline]
```

**16.87.3.19 mod()**

```
INLINE vect_t mod (
    vect_t & c ) const [inline]
```

**16.87.3.20 mul() [1/2]**

```
INLINE vect_t mul (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.87.3.21 mul() [2/2]**

```
INLINE vect_t mul (
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.87.3.22 mulin()**

```
INLINE vect_t mulin (
    vect_t & a,
    const vect_t b ) const [inline]
```

**16.87.3.23 mul\_r() [1/2]**

```
INLINE vect_t mul_r (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.87.3.24 mul\_r() [2/2]**

```
INLINE vect_t mul_r (
    const vect_t a,
    const vect_t b ) const [inline]
```

**16.87.3.25 axpy() [1/2]**

```
INLINE vect_t axpy (
    vect_t & r,
    const vect_t a,
```



```
const vect_t b,  
const vect_t c ) const [inline]
```

#### 16.87.3.26 axpy() [2/2]

```
INLINE vect_t axpy (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

#### 16.87.3.27 axpyin()

```
INLINE vect_t axpyin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

#### 16.87.3.28 axpy\_r() [1/2]

```
INLINE vect_t axpy_r (  
    vect_t & r,  
    const vect_t a,  
    const vect_t b,  
    const vect_t c ) const [inline]
```

#### 16.87.3.29 axpy\_r() [2/2]

```
INLINE vect_t axpy_r (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

#### 16.87.3.30 axpyin\_r()

```
INLINE vect_t axpyin_r (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) const [inline]
```

#### 16.87.3.31 maxpy() [1/2]

```
INLINE vect_t maxpy (  
    vect_t & r,  
    const vect_t a,  
    const vect_t b,  
    const vect_t c ) const [inline]
```

#### 16.87.3.32 maxpy() [2/2]

```
INLINE vect_t maxpy (  
    const vect_t c,
```

```
const vect_t a,
const vect_t b ) const [inline]
```

### 16.87.3.33 maxpyin()

```
INLINE vect_t maxpyin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) const [inline]
```

## 16.87.4 Field Documentation

### 16.87.4.1 vect\_size

```
const constexpr size_t vect_size = simd::vect_size [static], [constexpr]
```

### 16.87.4.2 alignment

```
const constexpr size_t alignment = simd::alignment [static], [constexpr]
```

The documentation for this class was generated from the following file:

- [simd\\_modular.inl](#)

## 16.88 FieldTraits< Field > Struct Template Reference

FieldTrait.

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::GenericTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.88.1 Detailed Description

```
template<class Field>
struct FFLAS::FieldTraits< Field >
```

FieldTrait.

### 16.88.2 Member Typedef Documentation

#### 16.88.2.1 category

```
typedef FieldCategories::GenericTag category
```

### 16.88.3 Field Documentation

### 16.88.3.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.89 FieldTraits< FFPACK::RNSInteger< T > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.89.1 Member Typedef Documentation

#### 16.89.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.89.2 Field Documentation

#### 16.89.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.90 FieldTraits< FFPACK::RNSIntegerMod< T > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::ModularTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.90.1 Member Typedef Documentation

**16.90.1.1 category**

```
typedef FieldCategories::ModularTag category
```

**16.90.2 Field Documentation****16.90.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.91 FieldTraits< Givaro::Modular< Element > > Struct Template Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [FieldCategories::ModularTag](#) category

**Static Public Attributes**

- static const bool [balanced](#) = false

**16.91.1 Member Typedef Documentation****16.91.1.1 category**

```
typedef FieldCategories::ModularTag category
```

**16.91.2 Field Documentation****16.91.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.92 FieldTraits< Givaro::ModularBalanced< Element > > Struct Template Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [FieldCategories::ModularTag](#) category

## Static Public Attributes

- static const bool [balanced](#) = true

### 16.92.1 Member Typedef Documentation

#### 16.92.1.1 category

typedef [FieldCategories::ModularTag](#) category

### 16.92.2 Field Documentation

#### 16.92.2.1 balanced

```
const bool balanced = true [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.93 FieldTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [FieldCategories::UnparametricTag](#) category

## Static Public Attributes

- static const bool [balanced](#) = false

### 16.93.1 Member Typedef Documentation

#### 16.93.1.1 category

typedef [FieldCategories::UnparametricTag](#) category

### 16.93.2 Field Documentation

#### 16.93.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.94 FieldTraits< Givaro::ZRing< float > > Struct Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [FieldCategories::UnparametricTag](#) category

## Static Public Attributes

- static const bool [balanced](#) = false

### 16.94.1 Member Typedef Documentation

#### 16.94.1.1 category

typedef [FieldCategories::UnparametricTag](#) category

### 16.94.2 Field Documentation

#### 16.94.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.95 FieldTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [FieldCategories::UnparametricTag](#) category

## Static Public Attributes

- static const bool [balanced](#) = false

### 16.95.1 Member Typedef Documentation

#### 16.95.1.1 category

typedef [FieldCategories::UnparametricTag](#) category

### 16.95.2 Field Documentation

#### 16.95.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.96 FieldTraits< Givaro::ZRing< int16\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.96.1 Member Typedef Documentation

#### 16.96.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.96.2 Field Documentation

#### 16.96.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.97 FieldTraits< Givaro::ZRing< int32\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.97.1 Member Typedef Documentation

#### 16.97.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.97.2 Field Documentation

**16.97.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.98 FieldTraits< Givaro::ZRing< int64\_t > > Struct Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [FieldCategories::UnparametricTag](#) category

**Static Public Attributes**

- static const bool [balanced](#) = false

**16.98.1 Member Typedef Documentation****16.98.1.1 category**

```
typedef FieldCategories::UnparametricTag category
```

**16.98.2 Field Documentation****16.98.2.1 balanced**

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

**16.99 FieldTraits< Givaro::ZRing< Reclnt::ruint< K > > > Struct Template Reference**

```
#include <field-traits.h>
```

**Public Types**

- typedef [FieldCategories::UnparametricTag](#) category

**Static Public Attributes**

- static const bool [balanced](#) = false

**16.99.1 Member Typedef Documentation****16.99.1.1 category**

```
typedef FieldCategories::UnparametricTag category
```



## 16.99.2 Field Documentation

### 16.99.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.100 FieldTraits< Givaro::ZRing< uint16\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

## 16.100.1 Member Typedef Documentation

### 16.100.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

## 16.100.2 Field Documentation

### 16.100.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.101 FieldTraits< Givaro::ZRing< uint32\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

## 16.101.1 Member Typedef Documentation

#### 16.101.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.101.2 Field Documentation

#### 16.101.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.102 FieldTraits< Givaro::ZRing< uint64\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [FieldCategories::UnparametricTag](#) category

### Static Public Attributes

- static const bool [balanced](#) = false

### 16.102.1 Member Typedef Documentation

#### 16.102.1.1 category

```
typedef FieldCategories::UnparametricTag category
```

### 16.102.2 Field Documentation

#### 16.102.2.1 balanced

```
const bool balanced = false [static]
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.103 Fixed Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.104 FixedPrecIntTag Struct Reference

Fixed precision integers above machine precision: Givaro::reclnt.

```
#include <field-traits.h>
```

### 16.104.1 Detailed Description

Fixed precision integers above machine precision: Givaro::reclnt.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.105 ForStrategy1D< blocksize\_t, Cut, Param > Struct Template Reference

### Public Member Functions

- [ForStrategy1D](#) (const blocksize\_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- [ForStrategy1D](#) (const blocksize\_t b, const blocksize\_t e, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- void [build](#) (const blocksize\_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- blocksize\_t [initialize](#) ()
- bool [isTerminated](#) () const
- blocksize\_t [begin](#) () const
- blocksize\_t [end](#) () const
- blocksize\_t [numblocks](#) () const
- blocksize\_t [blockindex](#) () const
- blocksize\_t [operator++](#) ()

### Protected Attributes

- blocksize\_t [ibeg](#)
- blocksize\_t [iend](#)
- blocksize\_t [current](#)
- blocksize\_t [firstBlockSize](#)
- blocksize\_t [lastBlockSize](#)
- blocksize\_t [changeBS](#)
- blocksize\_t [numBlock](#)

### 16.105.1 Constructor & Destructor Documentation

#### 16.105.1.1 ForStrategy1D() [1/2]

```
ForStrategy1D (
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

#### 16.105.1.2 ForStrategy1D() [2/2]

```
ForStrategy1D (
    const blocksize_t b,
    const blocksize_t e,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

### 16.105.2 Member Function Documentation

#### 16.105.2.1 build()

```
void build (
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

#### 16.105.2.2 initialize()

```
blocksize_t initialize ( ) [inline]
```

#### 16.105.2.3 isTerminated()

```
bool isTerminated ( ) const [inline]
```

#### 16.105.2.4 begin()

```
blocksize_t begin ( ) const [inline]
```

#### 16.105.2.5 end()

```
blocksize_t end ( ) const [inline]
```

#### 16.105.2.6 numblocks()

```
blocksize_t numblocks ( ) const [inline]
```

#### 16.105.2.7 blockindex()

```
blocksize_t blockindex ( ) const [inline]
```

#### 16.105.2.8 operator++()

```
blocksize_t operator++ ( ) [inline]
```

### 16.105.3 Field Documentation

#### 16.105.3.1 ibeg

```
blocksize_t ibeg [protected]
```

#### 16.105.3.2 iend

```
blocksize_t iend [protected]
```

#### 16.105.3.3 current

```
blocksize_t current [protected]
```

**16.105.3.4 firstBlockSize**

```
blocksize_t firstBlockSize [protected]
```

**16.105.3.5 lastBlockSize**

```
blocksize_t lastBlockSize [protected]
```

**16.105.3.6 changeBS**

```
blocksize_t changeBS [protected]
```

**16.105.3.7 numBlock**

```
blocksize_t numBlock [protected]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.106 ForStrategy2D< blocksize\_t, Cut, Param > Struct Template Reference

### Public Member Functions

- [ForStrategy2D](#) (const blocksize\_t m, const blocksize\_t n, const [ParSeqHelper::Parallel](#)< Cut, Param > H)
- blocksize\_t [initialize](#) ()
- bool [isTerminated](#) () const
- blocksize\_t [ibegin](#) () const
- blocksize\_t [jbegin](#) () const
- blocksize\_t [iend](#) () const
- blocksize\_t [jend](#) () const
- blocksize\_t [operator++](#) ()
- blocksize\_t [rownumblocks](#) () const
- blocksize\_t [colnumblocks](#) () const
- blocksize\_t [blockindex](#) () const
- blocksize\_t [rowblockindex](#) () const
- blocksize\_t [colblockindex](#) () const

### Protected Attributes

- blocksize\_t [\\_ibeg](#)
- blocksize\_t [\\_iend](#)
- blocksize\_t [\\_jbeg](#)
- blocksize\_t [\\_jend](#)
- blocksize\_t [rowBlockSize](#)
- blocksize\_t [colBlockSize](#)
- blocksize\_t [current](#)
- blocksize\_t [lastRBS](#)
- blocksize\_t [lastCBS](#)
- blocksize\_t [changeRBS](#)
- blocksize\_t [changeCBS](#)
- blocksize\_t [numRowBlock](#)
- blocksize\_t [numColBlock](#)
- blocksize\_t [BLOCKS](#)

## Friends

- `std::ostream & operator<< (std::ostream &out, const ForStrategy2D &FS2D)`

## 16.106.1 Constructor & Destructor Documentation

### 16.106.1.1 ForStrategy2D()

```
ForStrategy2D (
    const blocksize_t m,
    const blocksize_t n,
    const ParSeqHelper::Parallel< Cut, Param > H ) [inline]
```

## 16.106.2 Member Function Documentation

### 16.106.2.1 initialize()

```
blocksize_t initialize ( ) [inline]
```

### 16.106.2.2 isTerminated()

```
bool isTerminated ( ) const [inline]
```

### 16.106.2.3 ibegin()

```
blocksize_t ibegin ( ) const [inline]
```

### 16.106.2.4 jbegin()

```
blocksize_t jbegin ( ) const [inline]
```

### 16.106.2.5 iend()

```
blocksize_t iend ( ) const [inline]
```

### 16.106.2.6 jend()

```
blocksize_t jend ( ) const [inline]
```

### 16.106.2.7 operator++()

```
blocksize_t operator++ ( ) [inline]
```

### 16.106.2.8 rownumblocks()

```
blocksize_t rownumblocks ( ) const [inline]
```

#### 16.106.2.9 colnumblocks()

```
blocksize_t colnumblocks ( ) const [inline]
```

#### 16.106.2.10 blockindex()

```
blocksize_t blockindex ( ) const [inline]
```

#### 16.106.2.11 rowblockindex()

```
blocksize_t rowblockindex ( ) const [inline]
```

#### 16.106.2.12 colblockindex()

```
blocksize_t colblockindex ( ) const [inline]
```

### 16.106.3 Friends And Related Function Documentation

#### 16.106.3.1 operator<<

```
std::ostream & operator<< (
    std::ostream & out,
    const ForStrategy2D< blocksize_t, Cut, Param > & FS2D ) [friend]
```

### 16.106.4 Field Documentation

#### 16.106.4.1 \_ibeg

```
blocksize_t _ibeg [protected]
```

#### 16.106.4.2 \_iend

```
blocksize_t _iend [protected]
```

#### 16.106.4.3 \_jbeg

```
blocksize_t _jbeg [protected]
```

#### 16.106.4.4 \_jend

```
blocksize_t _jend [protected]
```

#### 16.106.4.5 rowBlockSize

```
blocksize_t rowBlockSize [protected]
```

**16.106.4.6 colBlockSize**

`blocksize_t colBlockSize [protected]`

**16.106.4.7 current**

`blocksize_t current [protected]`

**16.106.4.8 lastRBS**

`blocksize_t lastRBS [protected]`

**16.106.4.9 lastCBS**

`blocksize_t lastCBS [protected]`

**16.106.4.10 changeRBS**

`blocksize_t changeRBS [protected]`

**16.106.4.11 changeCBS**

`blocksize_t changeCBS [protected]`

**16.106.4.12 numRowsBlock**

`blocksize_t numRowsBlock [protected]`

**16.106.4.13 numColBlock**

`blocksize_t numColBlock [protected]`

**16.106.4.14 BLOCKS**

`blocksize_t BLOCKS [protected]`

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

**16.107 ftrmmLeftLowerNoTransNonUnit< Element > Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

**16.108 ftrmmLeftLowerNoTransUnit< Element > Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)



## 16.109 ftrmmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.110 ftrmmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.111 ftrmmLeftUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.112 ftrmmLeftUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.113 ftrmmLeftUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.114 ftrmmLeftUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.115 ftrmmRightLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.116 ftrmmRightLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.117 **ftrmmRightLowerTransNonUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.118 **ftrmmRightLowerTransUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.119 **ftrmmRightUpperNoTransNonUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.120 **ftrmmRightUpperNoTransUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.121 **ftrmmRightUpperTransNonUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.122 **ftrmmRightUpperTransUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.123 **ftrsmLeftLowerNoTransNonUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.124 **ftrsmLeftLowerNoTransUnit**< **Element** > **Class Template Reference**

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.125 ftrsmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.126 ftrsmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.127 ftrsmLeftUpperNoTransNonUnit< Element > Class Template Reference

Computes the maximal size for delaying the modular reduction in a triangular system resolution.

### 16.127.1 Detailed Description

```
template<class Element>
class FFLAS::Protected::ftrsmLeftUpperNoTransNonUnit< Element >
```

Computes the maximal size for delaying the modular reduction in a triangular system resolution.  
Compute the maximal dimension  $k$ , such that a unit diagonal triangular system of dimension  $k$  can be solved over  $\mathbb{Z}$  without overflow of the underlying floating point representation.

**Bibliography** • Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.

#### Parameters

|     |                                      |
|-----|--------------------------------------|
| $F$ | Finite Field/Ring of the computation |
|-----|--------------------------------------|

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.128 ftrsmLeftUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.129 ftrsmLeftUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.130 ftrsmLeftUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.131 **ftsmRightLowerNoTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.132 **ftsmRightLowerNoTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.133 **ftsmRightLowerTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.134 **ftsmRightLowerTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.135 **ftsmRightUpperNoTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.136 **ftsmRightUpperNoTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.137 **ftsmRightUpperTransNonUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

### 16.138 **ftsmRightUpperTransUnit**< Element > Class Template Reference

The documentation for this class was generated from the following file:

- [fflas\\_level3.inl](#)

## 16.139 GenericTag Struct Reference

default is generic

```
#include <field-traits.h>
```

### 16.139.1 Detailed Description

default is generic

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.140 GenericTag Struct Reference

generic ring.

```
#include <field-traits.h>
```

### 16.140.1 Detailed Description

generic ring.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.141 Grain Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.142 has\_minus\_eq\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

### 16.142.1 Field Documentation

#### 16.142.1.1 value

```
constexpr bool value = type::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.143 has\_minus\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

### 16.143.1 Field Documentation

#### 16.143.1.1 value

`constexpr bool value = type::value [static], [constexpr]`

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.144 has\_mul\_eq\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

### 16.144.1 Field Documentation

#### 16.144.1.1 value

`constexpr bool value = type::value [static], [constexpr]`

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.145 has\_mul\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

### 16.145.1 Field Documentation

#### 16.145.1.1 value

`constexpr bool value = type::value [static], [constexpr]`

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.146 has\_operation< T > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#)

## 16.146.1 Field Documentation

### 16.146.1.1 value

constexpr bool value [static], [constexpr]

**Initial value:**

```
= (has_plus<T>::value && has_minus<T>::value && has_equal<T>::value &&
    has_plus_eq<T>::value && has_minus_eq<T>::value && has_mul<T>::value
    && has_mul_eq<T>::value)
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.147 has\_plus\_eq\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

## 16.147.1 Field Documentation

### 16.147.1.1 value

constexpr bool value = type::value [static], [constexpr]

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.148 has\_plus\_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

### Static Public Attributes

- static constexpr bool [value](#) = type::value

## 16.148.1 Field Documentation

### 16.148.1.1 value

constexpr bool value = type::value [static], [constexpr]

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.149 HelperFlag Struct Reference

```
#include <fflas_sparse.h>
```

## Static Public Attributes

- static constexpr [uint64\\_t none](#) = 0\_ui64
- static constexpr [uint64\\_t coo](#) = 1\_ui64
- static constexpr [uint64\\_t csr](#) = 1\_ui64 << 1
- static constexpr [uint64\\_t ell](#) = 1\_ui64 << 2
- static constexpr [uint64\\_t aut](#) = 1\_ui64 << 32
- static constexpr [uint64\\_t pm1](#) = 1\_ui64 << 33

### 16.149.1 Field Documentation

#### 16.149.1.1 none

constexpr [uint64\\_t none](#) = 0\_ui64 [static], [constexpr]

#### 16.149.1.2 coo

constexpr [uint64\\_t coo](#) = 1\_ui64 [static], [constexpr]

#### 16.149.1.3 csr

constexpr [uint64\\_t csr](#) = 1\_ui64 << 1 [static], [constexpr]

#### 16.149.1.4 ell

constexpr [uint64\\_t ell](#) = 1\_ui64 << 2 [static], [constexpr]

#### 16.149.1.5 aut

constexpr [uint64\\_t aut](#) = 1\_ui64 << 32 [static], [constexpr]

#### 16.149.1.6 pm1

constexpr [uint64\\_t pm1](#) = 1\_ui64 << 33 [static], [constexpr]

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

### 16.150 HelperMod< Field, ElementTraits > Struct Template Reference

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

### 16.151 HelperMod< Field, ElementCategories::MachineIntTag > Struct Template Reference

#### Public Member Functions

- [HelperMod](#) ()
- [HelperMod](#) (const [Field](#) &F)



## Data Fields

- [Field::Element p](#)
- double [invp](#)
- double [min](#)
- double [max](#)
- [int64\\_t pow50rem](#)

## 16.151.1 Constructor & Destructor Documentation

### 16.151.1.1 HelperMod() [1/2]

```
HelperMod ( ) [inline]
```

### 16.151.1.2 HelperMod() [2/2]

```
HelperMod (
    const Field & F ) [inline]
```

## 16.151.2 Field Documentation

### 16.151.2.1 p

```
Field::Element p
```

### 16.151.2.2 invp

```
double invp
```

### 16.151.2.3 min

```
double min
```

### 16.151.2.4 max

```
double max
```

### 16.151.2.5 pow50rem

```
int64\_t pow50rem
```

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

## 16.152 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > Struct Template Reference

### Public Member Functions

- [HelperMod\(\)](#)
- [HelperMod\(const Field &F\)](#)

### Data Fields

- [Field::Element p](#)

### 16.152.1 Constructor & Destructor Documentation

#### 16.152.1.1 HelperMod() [1/2]

[HelperMod\(\)](#) [inline]

#### 16.152.1.2 HelperMod() [2/2]

[HelperMod\(const Field &F\)](#) [inline]

### 16.152.2 Field Documentation

#### 16.152.2.1 p

[Field::Element p](#)

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

## 16.153 HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > Struct Template Reference

### Public Member Functions

- [HelperMod\(\)](#)
- [HelperMod\(const Field &F\)](#)

### Data Fields

- [Field::Element p](#)

### 16.153.1 Constructor & Destructor Documentation

#### 16.153.1.1 HelperMod() [1/2]

[HelperMod\(\)](#) [inline]

**16.153.1.2 HelperMod()** [2/2]

```
HelperMod (
    const Field & F ) [inline]
```

**16.153.2 Field Documentation****16.153.2.1 p**

Field::Element p

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

## 16.154 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag > Struct Template Reference

**Public Member Functions**

- [HelperMod\(\)](#)
- [HelperMod](#) (const [Field](#) &F)

**Data Fields**

- [Field::Element](#) p
- [Field::Element](#) invp
- [Field::Element](#) min
- [Field::Element](#) max

**16.154.1 Constructor & Destructor Documentation****16.154.1.1 HelperMod()** [1/2]

```
HelperMod ( ) [inline]
```

**16.154.1.2 HelperMod()** [2/2]

```
HelperMod (
    const Field & F ) [inline]
```

**16.154.2 Field Documentation****16.154.2.1 p**

Field::Element p

**16.154.2.2 invp**

Field::Element invp

**16.154.2.3 min**

`Field::Element min`

**16.154.2.4 max**

`Field::Element max`

The documentation for this struct was generated from the following file:

- [fflas\\_freduce.inl](#)

**16.155 Hybrid Struct Reference**

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

**16.156 Info Struct Reference****Public Member Functions**

- [Info](#) ([uint64\\_t](#) it, [uint64\\_t](#) s, [uint64\\_t](#) p)
- [Info](#) ()=default
- [Info](#) (const [Info](#) &)=default
- [Info](#) ([Info](#) &&)=default
- [Info](#) & [operator=](#) (const [Info](#) &)=default
- [Info](#) & [operator=](#) ([Info](#) &&)=default

**Data Fields**

- [uint64\\_t](#) size = 0
- [uint64\\_t](#) perm = 0
- [uint64\\_t](#) begin = 0

**16.156.1 Constructor & Destructor Documentation****16.156.1.1 Info() [1/4]**

```
Info (
    uint64_t it,
    uint64_t s,
    uint64_t p ) [inline]
```

**16.156.1.2 Info() [2/4]**

```
Info ( ) [default]
```

**16.156.1.3 Info() [3/4]**

```
Info (
    const Info & ) [default]
```

#### 16.156.1.4 Info() [4/4]

```
Info (
    Info && ) [default]
```

### 16.156.2 Member Function Documentation

#### 16.156.2.1 operator=() [1/2]

```
Info & operator= (
    const Info & ) [default]
```

#### 16.156.2.2 operator=() [2/2]

```
Info & operator= (
    Info && ) [default]
```

### 16.156.3 Field Documentation

#### 16.156.3.1 size

```
uint64_t size = 0
```

#### 16.156.3.2 perm

```
uint64_t perm = 0
```

#### 16.156.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- [csr\\_hyb\\_utils.inl](#)

## 16.157 Info Struct Reference

### Public Member Functions

- [Info](#) (uint64\_t it, uint64\_t s, uint64\_t p)
- [Info](#) ()=default
- [Info](#) (const [Info](#) &)=default
- [Info](#) ([Info](#) &&)=default
- [Info](#) & [operator=](#) (const [Info](#) &)=default
- [Info](#) & [operator=](#) ([Info](#) &&)=default

### Data Fields

- [uint64\\_t](#) size = 0
- [uint64\\_t](#) perm = 0
- [uint64\\_t](#) begin = 0

## 16.157.1 Constructor & Destructor Documentation

### 16.157.1.1 Info() [1/4]

```
Info (
    uint64_t it,
    uint64_t s,
    uint64_t p ) [inline]
```

### 16.157.1.2 Info() [2/4]

```
Info ( ) [default]
```

### 16.157.1.3 Info() [3/4]

```
Info (
    const Info & ) [default]
```

### 16.157.1.4 Info() [4/4]

```
Info (
    Info && ) [default]
```

## 16.157.2 Member Function Documentation

### 16.157.2.1 operator=() [1/2]

```
Info & operator= (
    const Info & ) [default]
```

### 16.157.2.2 operator=() [2/2]

```
Info & operator= (
    Info && ) [default]
```

## 16.157.3 Field Documentation

### 16.157.3.1 size

```
uint64_t size = 0
```

### 16.157.3.2 perm

```
uint64_t perm = 0
```

### 16.157.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- [sell\\_utils.inl](#)

## 16.158 is\_simd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

### Public Types

- using [type](#) = std::integral\_constant< bool, false >

### Static Public Attributes

- static const constexpr bool [value](#) = false

### 16.158.1 Member Typedef Documentation

#### 16.158.1.1 type

```
using type = std::integral_constant<bool, false>
```

### 16.158.2 Field Documentation

#### 16.158.2.1 value

```
const constexpr bool value = false [static], [constexpr]
```

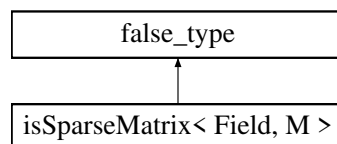
The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

## 16.159 isSparseMatrix< Field, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, M >:



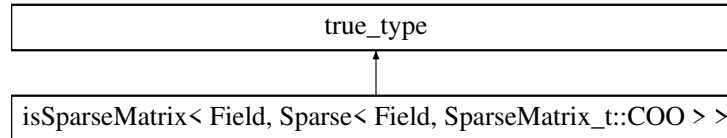
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.160 `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >` Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >`:



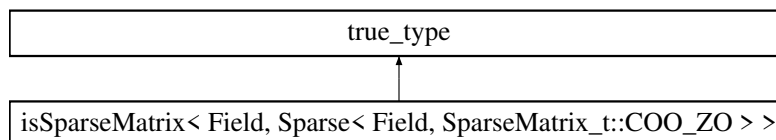
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.161 `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >` Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >`:



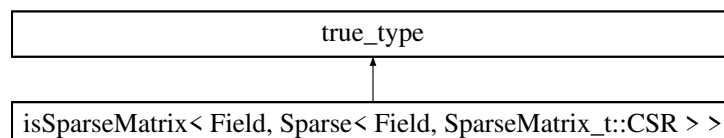
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.162 `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >` Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >`:



The documentation for this struct was generated from the following file:

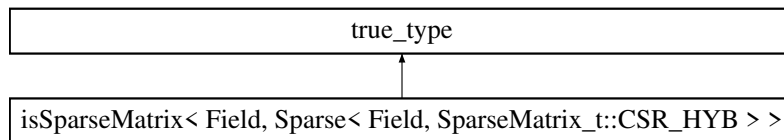
- [sparse\\_matrix\\_traits.h](#)

## 16.163 `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >` Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for `isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >`:





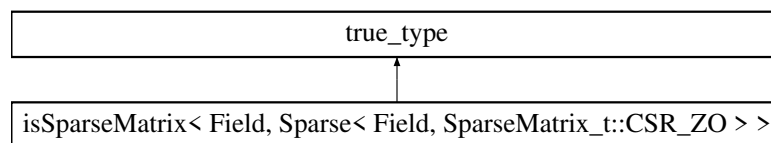
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.164 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >:



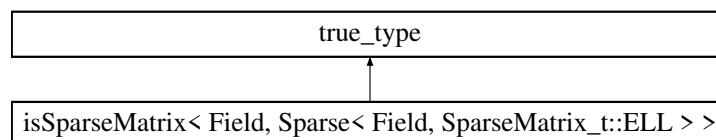
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.165 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL > >:



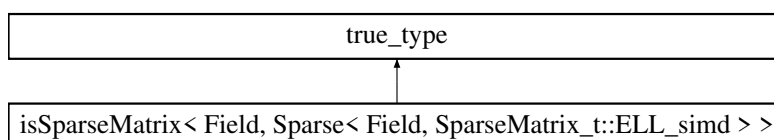
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.166 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > >:



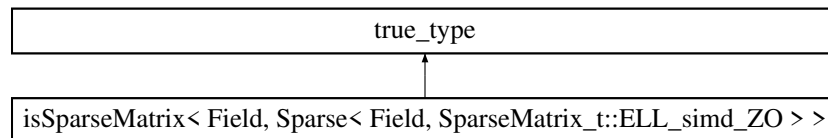
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.167 **isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >:



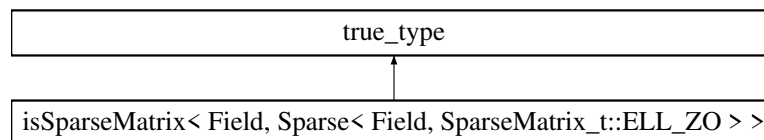
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.168 **isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >:



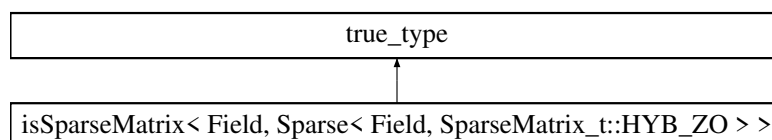
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.169 **isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > > Struct Template Reference**

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > >:



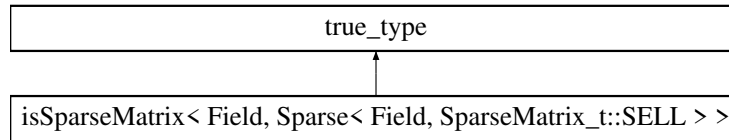
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.170 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL > >:



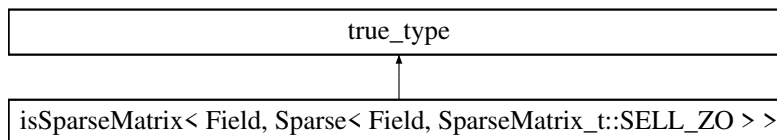
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.171 isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >:



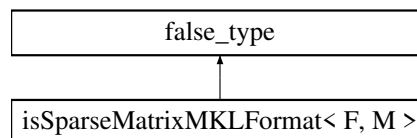
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.172 isSparseMatrixMKLFormat< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrixMKLFormat< F, M >:



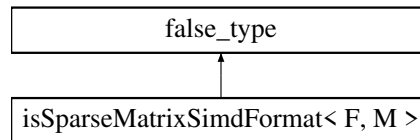
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.173 isSparseMatrixSimdFormat< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrixSimdFormat< F, M >:



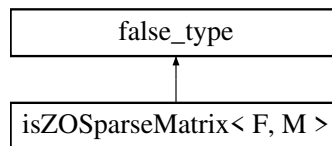
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.174 isZOSparseMatrix< F, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< F, M >:



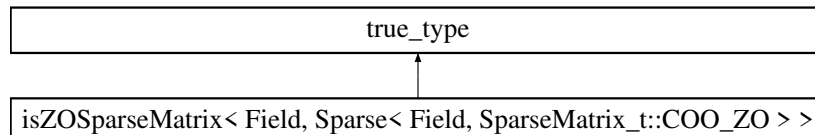
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.175 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >:



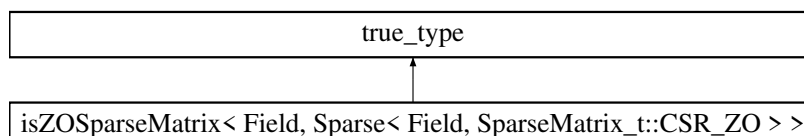
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.176 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >:



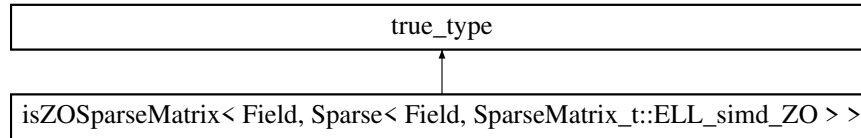
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.177 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >:



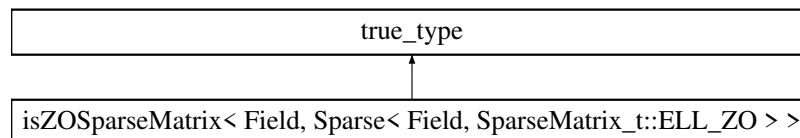
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.178 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >:



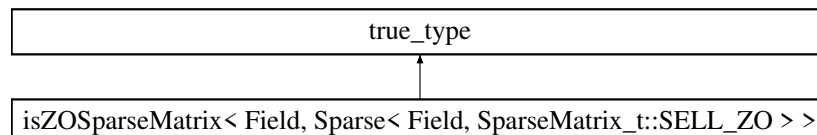
The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.179 isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >:



The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.180 Iterative Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.181 LazyTag Struct Reference

Performs field operations with delayed mod only when necessary. Result may not be reduced.

```
#include <field-traits.h>
```

### 16.181.1 Detailed Description

Performs field operations with delayed mod only when necessary. Result may not be reduced.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.182 limits< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.183 limits< char > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef char [T](#)

### Static Public Member Functions

- static constexpr char [max](#) () noexcept
- static constexpr char [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

### 16.183.1 Member Typedef Documentation

#### 16.183.1.1 T

```
typedef char T
```

### 16.183.2 Member Function Documentation

#### 16.183.2.1 max()

```
static constexpr char max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.183.2.2 min()

```
static constexpr char min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.183.2.3 digits()

```
static constexpr int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.184 limits< double > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef double [T](#)

### Static Public Member Functions

- static constexpr [int64\\_t](#) [max](#) () noexcept
- static constexpr [int64\\_t](#) [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

### 16.184.1 Member Typedef Documentation

#### 16.184.1.1 T

```
typedef double T
```

### 16.184.2 Member Function Documentation

#### 16.184.2.1 max()

```
static constexpr int64\_t max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.184.2.2 min()

```
static constexpr int64\_t min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.184.2.3 digits()

```
static constexpr int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.185 limits< float > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef float [T](#)

### Static Public Member Functions

- static constexpr [int32\\_t](#) [max](#) () noexcept
- static constexpr [int32\\_t](#) [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

## 16.185.1 Member Typedef Documentation

### 16.185.1.1 T

```
typedef float T
```

## 16.185.2 Member Function Documentation

### 16.185.2.1 max()

```
static constexpr int32_t max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.185.2.2 min()

```
static constexpr int32_t min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.185.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.186 limits< Givaro::Integer > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef Givaro::Integer T

### Static Public Member Functions

- static constexpr int [max](#) () noexcept
- static constexpr int [min](#) () noexcept

## 16.186.1 Member Typedef Documentation

### 16.186.1.1 T

```
typedef Givaro::Integer T
```

## 16.186.2 Member Function Documentation

### 16.186.2.1 max()

```
static constexpr int max ( ) [inline], [static], [constexpr], [noexcept]
```



### 16.186.2.2 min()

```
static constexpr int min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.187 limits< int > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef int [T](#)

### Static Public Member Functions

- static constexpr int [max](#) () noexcept
- static constexpr int [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

## 16.187.1 Member Typedef Documentation

### 16.187.1.1 T

```
typedef int T
```

## 16.187.2 Member Function Documentation

### 16.187.2.1 max()

```
static constexpr int max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.187.2.2 min()

```
static constexpr int min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.187.2.3 digits()

```
static constexpr int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.188 limits< long > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef long [T](#)

## Static Public Member Functions

- static constexpr long [max](#) () noexcept
- static constexpr long [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

### 16.188.1 Member Typedef Documentation

#### 16.188.1.1 T

```
typedef long T
```

### 16.188.2 Member Function Documentation

#### 16.188.2.1 max()

```
static constexpr long max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.188.2.2 min()

```
static constexpr long min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.188.2.3 digits()

```
static constexpr int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.189 limits< long long > Struct Reference

```
#include <flimits.h>
```

## Public Types

- typedef long long [T](#)

## Static Public Member Functions

- static constexpr long long [max](#) () noexcept
- static constexpr long long [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

### 16.189.1 Member Typedef Documentation

#### 16.189.1.1 T

```
typedef long long T
```

## 16.189.2 Member Function Documentation

### 16.189.2.1 max()

```
static constexpr long long max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.189.2.2 min()

```
static constexpr long long min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.189.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.190 limits< RecInt::rint< K > > Struct Template Reference

```
#include <flimits.h>
```

### Public Types

- typedef [RecInt::ruint](#)< K > [T](#)

### Static Public Member Functions

- static constexpr [RecInt::rint](#)< K > [max](#) () noexcept
- static constexpr [RecInt::rint](#)< K > [min](#) () noexcept

## 16.190.1 Member Typedef Documentation

### 16.190.1.1 T

```
typedef RecInt::ruint<K> T
```

## 16.190.2 Member Function Documentation

### 16.190.2.1 max()

```
static constexpr RecInt::rint< K > max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.190.2.2 min()

```
static constexpr RecInt::rint< K > min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.191 limits< RecInt::ruint< K > > Struct Template Reference

```
#include <flimits.h>
```

### Public Types

- typedef [RecInt::ruint](#)< K > [T](#)

### Static Public Member Functions

- static constexpr [RecInt::ruint](#)< K > [max](#) () noexcept
- static constexpr [RecInt::ruint](#)< K > [min](#) () noexcept

### 16.191.1 Member Typedef Documentation

#### 16.191.1.1 T

```
typedef RecInt::ruint<K> T
```

### 16.191.2 Member Function Documentation

#### 16.191.2.1 max()

```
static constexpr RecInt::ruint< K > max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.191.2.2 min()

```
static constexpr RecInt::ruint< K > min ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.192 limits< short int > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef short int [T](#)

### Static Public Member Functions

- static constexpr short int [max](#) () noexcept
- static constexpr short int [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

### 16.192.1 Member Typedef Documentation

#### 16.192.1.1 T

```
typedef short int T
```

## 16.192.2 Member Function Documentation

### 16.192.2.1 max()

```
static constexpr short int max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.192.2.2 min()

```
static constexpr short int min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.192.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.193 limits< signed char > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef signed char [T](#)

### Static Public Member Functions

- static constexpr signed char [max](#) () noexcept
- static constexpr signed char [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

## 16.193.1 Member Typedef Documentation

### 16.193.1.1 T

```
typedef signed char T
```

## 16.193.2 Member Function Documentation

### 16.193.2.1 max()

```
static constexpr signed char max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.193.2.2 min()

```
static constexpr signed char min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.193.2.3 digits()

```
static constexpr int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.194 limits< unsigned char > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef unsigned char [T](#)

### Static Public Member Functions

- static constexpr unsigned char [max](#) () noexcept
- static constexpr unsigned char [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

### 16.194.1 Member Typedef Documentation

#### 16.194.1.1 T

```
typedef unsigned char T
```

### 16.194.2 Member Function Documentation

#### 16.194.2.1 max()

```
static constexpr unsigned char max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.194.2.2 min()

```
static constexpr unsigned char min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.194.2.3 digits()

```
static constexpr int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.195 limits< unsigned int > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef unsigned int [T](#)

## Static Public Member Functions

- static constexpr unsigned int [max](#) () noexcept
- static constexpr unsigned int [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

### 16.195.1 Member Typedef Documentation

#### 16.195.1.1 T

```
typedef unsigned int T
```

### 16.195.2 Member Function Documentation

#### 16.195.2.1 max()

```
static constexpr unsigned int max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.195.2.2 min()

```
static constexpr unsigned int min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.195.2.3 digits()

```
static constexpr int32\_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.196 limits< unsigned long > Struct Reference

```
#include <flimits.h>
```

## Public Types

- typedef unsigned long [T](#)

## Static Public Member Functions

- static constexpr unsigned long [max](#) () noexcept
- static constexpr unsigned long [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

### 16.196.1 Member Typedef Documentation

#### 16.196.1.1 T

```
typedef unsigned long T
```

## 16.196.2 Member Function Documentation

### 16.196.2.1 max()

```
static constexpr unsigned long max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.196.2.2 min()

```
static constexpr unsigned long min ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.196.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.197 limits< unsigned long long > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef unsigned long long [T](#)

### Static Public Member Functions

- static constexpr unsigned long long [max](#) () noexcept
- static constexpr unsigned long long [min](#) () noexcept
- static constexpr [int32\\_t](#) [digits](#) () noexcept

## 16.197.1 Member Typedef Documentation

### 16.197.1.1 T

```
typedef unsigned long long T
```

## 16.197.2 Member Function Documentation

### 16.197.2.1 max()

```
static constexpr unsigned long long max ( ) [inline], [static], [constexpr], [noexcept]
```

### 16.197.2.2 min()

```
static constexpr unsigned long long min ( ) [inline], [static], [constexpr], [noexcept]
```



### 16.197.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.198 limits< unsigned short int > Struct Reference

```
#include <flimits.h>
```

### Public Types

- typedef unsigned short int [T](#)

### Static Public Member Functions

- static constexpr unsigned short int [max](#) () noexcept
- static constexpr unsigned short int [min](#) () noexcept
- static constexpr [int32\\_t digits](#) () noexcept

### 16.198.1 Member Typedef Documentation

#### 16.198.1.1 T

```
typedef unsigned short int T
```

### 16.198.2 Member Function Documentation

#### 16.198.2.1 max()

```
static constexpr unsigned short int max ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.198.2.2 min()

```
static constexpr unsigned short int min ( ) [inline], [static], [constexpr], [noexcept]
```

#### 16.198.2.3 digits()

```
static constexpr int32_t digits ( ) [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- [flimits.h](#)

## 16.199 MachineFloatTag Struct Reference

float or double

```
#include <field-traits.h>
```

### 16.199.1 Detailed Description

float or double

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.200 MachineIntTag Struct Reference

short, int, long, long long, and unsigned variants

```
#include <field-traits.h>
```

### 16.200.1 Detailed Description

short, int, long, long long, and unsigned variants

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.201 MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeTrait](#), [ParSeqTrait](#) > [Self\\_t](#)
- typedef [associatedDelayedField](#)< constField >::type [DelayedField\\_t](#)
- typedef [associatedDelayedField](#)< constField >::field [DelayedField](#)
- typedef [DelayedField::Element](#) [DFElt](#)

### Public Member Functions

- void [initC](#) ()
- void [initA](#) ()
- void [initB](#) ()
- void [initOut](#) ()
- size\_t [MaxDelayedDim](#) ([DFElt](#) beta)
- bool [Aunfit](#) ()
- bool [Bunfit](#) ()
- void [setOutBounds](#) (const size\_t k, const [DFElt](#) alpha, const [DFElt](#) beta)
- bool [checkA](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)
- bool [checkB](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_TRANSPOSE](#) tb, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)
- bool [checkOut](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)
- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, [ParSeqTrait](#) \_PS)
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait](#) \_PS=[ParSeqTrait](#)())
- template<class F2, typename AlgoT2, typename FT2, typename PS2 >  
[MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)
- [MMHelper](#) (const [Field](#) &F, int w, [DFElt](#) \_Amin, [DFElt](#) \_Amax, [DFElt](#) \_Bmin, [DFElt](#) \_Bmax, [DFElt](#) \_Cmin, [DFElt](#) \_Cmax, [ParSeqTrait](#) \_PS=[ParSeqTrait](#)())

## Data Fields

- int [recLevel](#)
- [DFElt](#) FieldMin
- [DFElt](#) FieldMax
- [DFElt](#) Amin
- [DFElt](#) Amax
- [DFElt](#) Bmin
- [DFElt](#) Bmax
- [DFElt](#) Cmin
- [DFElt](#) Cmax
- [DFElt](#) Outmin
- [DFElt](#) Outmax
- [DFElt](#) MaxStorableValue
- const [DelayedField\\_t](#) [delayedField](#)
- [ParSeqTrait](#) [parseq](#)

## Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

## 16.201.1 Member Typedef Documentation

### 16.201.1.1 Self\_t

```
typedef MMHelper<Field,AlgoTrait,ModeTrait,ParSeqTrait> Self_t
```

### 16.201.1.2 DelayedField\_t

```
typedef associatedDelayedField<constField>::type DelayedField_t
```

### 16.201.1.3 DelayedField

```
typedef associatedDelayedField<constField>::field DelayedField
```

### 16.201.1.4 DFElt

```
typedef DelayedField::Element DFElt
```

## 16.201.2 Constructor & Destructor Documentation

### 16.201.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

**16.201.2.2 MMHelper() [2/5]**

```
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

**16.201.2.3 MMHelper() [3/5]**

```
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

**16.201.2.4 MMHelper() [4/5]**

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

**16.201.2.5 MMHelper() [5/5]**

```
MMHelper (
    const Field & F,
    int w,
    DFelt _Amin,
    DFelt _Amax,
    DFelt _Bmin,
    DFelt _Bmax,
    DFelt _Cmin,
    DFelt _Cmax,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

**16.201.3 Member Function Documentation****16.201.3.1 initC()**

```
void initC ( ) [inline]
```

**16.201.3.2 initA()**

```
void initA ( ) [inline]
```

**16.201.3.3 initB()**

```
void initB ( ) [inline]
```

**16.201.3.4 initOut()**

```
void initOut ( ) [inline]
```

**16.201.3.5 MaxDelayedDim()**

```
size_t MaxDelayedDim (
    DFElt beta ) [inline]
```

**16.201.3.6 Aunfit()**

```
bool Aunfit ( ) [inline]
```

**16.201.3.7 Bunfit()**

```
bool Bunfit ( ) [inline]
```

**16.201.3.8 setOutBounds()**

```
void setOutBounds (
    const size_t k,
    const DFElt alpha,
    const DFElt beta ) [inline]
```

**16.201.3.9 checkA()**

```
bool checkA (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE ta,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda ) [inline]
```

**16.201.3.10 checkB()**

```
bool checkB (
    const Field & F,
    const FFLAS::FFLAS_TRANSPOSE tb,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr B,
    const size_t ldb ) [inline]
```

**16.201.3.11 checkOut()**

```
bool checkOut (
    const Field & F,
    const size_t M,
    const size_t N,
    typename Field::ConstElement_ptr A,
    const size_t lda ) [inline]
```

**16.201.4 Friends And Related Function Documentation**

**16.201.4.1 operator<<**

```
std::ostream & operator<< (  
    std::ostream & out,  
    const Self_t & M ) [friend]
```

**16.201.5 Field Documentation****16.201.5.1 recLevel**

```
int recLevel
```

**16.201.5.2 FieldMin**

```
DFElt FieldMin
```

**16.201.5.3 FieldMax**

```
DFElt FieldMax
```

**16.201.5.4 Amin**

```
DFElt Amin
```

**16.201.5.5 Amax**

```
DFElt Amax
```

**16.201.5.6 Bmin**

```
DFElt Bmin
```

**16.201.5.7 Bmax**

```
DFElt Bmax
```

**16.201.5.8 Cmin**

```
DFElt Cmin
```

**16.201.5.9 Cmax**

```
DFElt Cmax
```

**16.201.5.10 Outmin**

```
DFElt Outmin
```

### 16.201.5.11 Outmax

[DFelt](#) Outmax

### 16.201.5.12 MaxStorableValue

[DFelt](#) MaxStorableValue

### 16.201.5.13 delayedField

const [DelayedField\\_t](#) delayedField

### 16.201.5.14 parseq

[ParSeqTrait](#) parseq

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.202 MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [FFPACK::RNSInteger](#)< E >, [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (Givaro::Integer Amax, Givaro::Integer Bmax)
- [MMHelper](#) (const [FFPACK::RNSInteger](#)< E > &F, size\_t m, size\_t n, size\_t k, [ParSeqTrait](#) PS=[ParSeqTrait](#)())
- [MMHelper](#) (const [FFPACK::RNSInteger](#)< E > &F, int wino, [ParSeqTrait](#) PS=[ParSeqTrait](#)())
- template<class F2 , class A2 , class M2 , class PS2 > [MMHelper](#) ([MMHelper](#)< F2, A2, M2, PS2 > H2)
- void [setNorm](#) (Givaro::Integer p)

### Data Fields

- Givaro::Integer [normA](#)
- Givaro::Integer [normB](#)
- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

### 16.202.1 Member Typedef Documentation

### 16.202.1.1 Self\_t

```
typedef MMHelper<FFPACK::RNSInteger<E>, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait>
Self_t
```

## 16.202.2 Constructor & Destructor Documentation

### 16.202.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

### 16.202.2.2 MMHelper() [2/5]

```
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

### 16.202.2.3 MMHelper() [3/5]

```
MMHelper (
    const FFPACK::RNSInteger< E > & F,
    size_t m,
    size_t n,
    size_t k,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

### 16.202.2.4 MMHelper() [4/5]

```
MMHelper (
    const FFPACK::RNSInteger< E > & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

### 16.202.2.5 MMHelper() [5/5]

```
MMHelper (
    MMHelper< F2, A2, M2, PS2 > H2 ) [inline]
```

## 16.202.3 Member Function Documentation

### 16.202.3.1 setNorm()

```
void setNorm (
    Givaro::Integer p ) [inline]
```

## 16.202.4 Friends And Related Function Documentation



#### 16.202.4.1 operator<<

```
std::ostream & operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

### 16.202.5 Field Documentation

#### 16.202.5.1 normA

Givaro::Integer normA

#### 16.202.5.2 normB

Givaro::Integer normB

#### 16.202.5.3 recLevel

int recLevel

#### 16.202.5.4 parseq

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fgemm\\_classical\\_mp.inl](#)

## 16.203 MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

### Public Types

- typedef MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, [ModeCategories::DefaultTag](#), ParSeqTrait > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (Givaro::Integer Amax, Givaro::Integer Bmax)
- [MMHelper](#) (const [FFPACK::RNSIntegerMod](#)< E > &F, size\_t m, size\_t n, size\_t k, ParSeqTrait PS=[ParSeqTrait](#)())
- [MMHelper](#) (const [FFPACK::RNSIntegerMod](#)< E > &F, int wino, ParSeqTrait PS=[ParSeqTrait](#)())
- template<class F2, typename AlgoT2, typename FT2, typename PS2 > [MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)
- void [setNorm](#) (Givaro::Integer p)

### Data Fields

- Givaro::Integer [normA](#)
- Givaro::Integer [normB](#)
- int [recLevel](#)
- ParSeqTrait [parseq](#)

## Friends

- `std::ostream & operator<< (std::ostream &out, const Self_t &M)`

### 16.203.1 Member Typedef Documentation

#### 16.203.1.1 Self\_t

```
typedef MMHelper<FFPACK::RNSIntegerMod<E>, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait>
Self_t
```

### 16.203.2 Constructor & Destructor Documentation

#### 16.203.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

#### 16.203.2.2 MMHelper() [2/5]

```
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

#### 16.203.2.3 MMHelper() [3/5]

```
MMHelper (
    const FFPACK::RNSIntegerMod< E > & F,
    size_t m,
    size_t n,
    size_t k,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

#### 16.203.2.4 MMHelper() [4/5]

```
MMHelper (
    const FFPACK::RNSIntegerMod< E > & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

#### 16.203.2.5 MMHelper() [5/5]

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

### 16.203.3 Member Function Documentation

#### 16.203.3.1 setNorm()

```
void setNorm (
    Givaro::Integer p ) [inline]
```

## 16.203.4 Friends And Related Function Documentation

### 16.203.4.1 operator<<

```
std::ostream & operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

## 16.203.5 Field Documentation

### 16.203.5.1 normA

Givaro::Integer normA

### 16.203.5.2 normB

Givaro::Integer normB

### 16.203.5.3 recLevel

int recLevel

### 16.203.5.4 parseq

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fgemm\\_classical\\_mp.inl](#)

## 16.204 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::ConvertTo](#)< [Dest](#) >, [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, [ParSeqTrait](#) \_PS)
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait](#) \_PS=[ParSeqTrait](#)())
- template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 >  
[MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)

### Data Fields

- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

## 16.204.1 Member Typedef Documentation

### 16.204.1.1 Self\_t

```
typedef MMHelper<Field,AlgoTrait, ModeCategories::ConvertTo<Dest>,ParSeqTrait> Self_t
```

## 16.204.2 Constructor & Destructor Documentation

### 16.204.2.1 MMHelper() [1/4]

```
MMHelper ( ) [inline]
```

### 16.204.2.2 MMHelper() [2/4]

```
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

### 16.204.2.3 MMHelper() [3/4]

```
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

### 16.204.2.4 MMHelper() [4/4]

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

## 16.204.3 Friends And Related Function Documentation

### 16.204.3.1 operator<<

```
std::ostream & operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

## 16.204.4 Field Documentation

### 16.204.4.1 recLevel

```
int recLevel
```

#### 16.204.4.2 parseq

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

## 16.205 MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait > Struct Template Reference

### Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) >, [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- template<class F2 , class A2 , class M2 , class PS2 >  
[MMHelper](#) ([MMHelper](#)< F2, A2, M2, PS2 > H2)
- [MMHelper](#) (Givaro::Integer Amax, Givaro::Integer Bmax)
- [MMHelper](#) (const [Field](#) &F, size\_t m, size\_t n, size\_t k, ParSeqTrait PS=ParSeqTrait())
- [MMHelper](#) (const [Field](#) &F, int wino, ParSeqTrait PS=ParSeqTrait())
- void [setNorm](#) (Givaro::Integer p)

### Data Fields

- Givaro::Integer [normA](#)
- Givaro::Integer [normB](#)
- int [recLevel](#)
- ParSeqTrait [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

## 16.205.1 Member Typedef Documentation

### 16.205.1.1 Self\_t

```
typedef MMHelper<Field, AlgoTrait,ModeCategories::ConvertTo<ElementCategories::RNSElementTag>,  
ParSeqTrait> Self\_t
```

## 16.205.2 Constructor & Destructor Documentation

### 16.205.2.1 MMHelper() [1/5]

```
MMHelper ( ) [inline]
```

**16.205.2.2 MMHelper() [2/5]**

```
MMHelper (
    MMHelper< F2, A2, M2, PS2 > H2 ) [inline]
```

**16.205.2.3 MMHelper() [3/5]**

```
MMHelper (
    Givaro::Integer Amax,
    Givaro::Integer Bmax ) [inline]
```

**16.205.2.4 MMHelper() [4/5]**

```
MMHelper (
    const Field & F,
    size_t m,
    size_t n,
    size_t k,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

**16.205.2.5 MMHelper() [5/5]**

```
MMHelper (
    const Field & F,
    int wino,
    ParSeqTrait PS = ParSeqTrait() ) [inline]
```

**16.205.3 Member Function Documentation****16.205.3.1 setNorm()**

```
void setNorm (
    Givaro::Integer p ) [inline]
```

**16.205.4 Friends And Related Function Documentation****16.205.4.1 operator<<**

```
std::ostream & operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

**16.205.5 Field Documentation****16.205.5.1 normA**

```
Givaro::Integer normA
```

### 16.205.5.2 normB

Givaro::Integer normB

### 16.205.5.3 recLevel

int recLevel

### 16.205.5.4 parseq

ParSeqTrait parseq

The documentation for this struct was generated from the following file:

- [fgemm\\_classical\\_mp.inl](#)

## 16.206 MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait > Struct Template Reference

FGEMM Helper for Default and ConvertTo modes of operation.

### Public Types

- typedef [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) > [Self\\_t](#)

### Public Member Functions

- [MMHelper](#) ()
- [MMHelper](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, [ParSeqTrait](#) \_PS)
- [MMHelper](#) (const [Field](#) &F, int w, [ParSeqTrait](#) \_PS=[ParSeqTrait](#)())
- template<class F2 , typename AlgoT2 , typename FT2 , typename PS2 >  
[MMHelper](#) ([MMHelper](#)< F2, AlgoT2, FT2, PS2 > &WH)

### Data Fields

- int [recLevel](#)
- [ParSeqTrait](#) [parseq](#)

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Self\\_t](#) &M)

### 16.206.1 Detailed Description

```
template<class Field, typename AlgoTrait, typename ParSeqTrait>
struct FFLAS::MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >
```

FGEMM Helper for Default and ConvertTo modes of operation.

### 16.206.2 Member Typedef Documentation

#### 16.206.2.1 Self\_t

```
typedef MMHelper<Field,AlgoTrait, ModeCategories::DefaultTag,ParSeqTrait> Self\_t
```

### 16.206.3 Constructor & Destructor Documentation

#### 16.206.3.1 MMHelper() [1/4]

```
MMHelper ( ) [inline]
```

#### 16.206.3.2 MMHelper() [2/4]

```
MMHelper (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait _PS ) [inline]
```

#### 16.206.3.3 MMHelper() [3/4]

```
MMHelper (
    const Field & F,
    int w,
    ParSeqTrait _PS = ParSeqTrait() ) [inline]
```

#### 16.206.3.4 MMHelper() [4/4]

```
MMHelper (
    MMHelper< F2, AlgoT2, FT2, PS2 > & WH ) [inline]
```

### 16.206.4 Friends And Related Function Documentation

#### 16.206.4.1 operator<<

```
std::ostream & operator<< (
    std::ostream & out,
    const Self_t & M ) [friend]
```

### 16.206.5 Field Documentation

#### 16.206.5.1 recLevel

```
int recLevel
```

#### 16.206.5.2 parseq

```
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)



## 16.207 ModeTraits< Field > Struct Template Reference

[ModeTraits.](#)

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DefaultTag](#) value

#### 16.207.1 Detailed Description

```
template<class Field>
```

```
struct FFLAS::ModeTraits< Field >
```

[ModeTraits.](#)

#### 16.207.2 Member Typedef Documentation

##### 16.207.2.1 value

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.208 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DelayedTag](#) value

#### 16.208.1 Member Typedef Documentation

##### 16.208.1.1 value

```
typedef ModeCategories::DelayedTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.209 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) > value

## 16.209.1 Member Typedef Documentation

### 16.209.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.210 ModeTraits< Givaro::Modular< int16\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

### 16.210.1 Member Typedef Documentation

#### 16.210.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.211 ModeTraits< Givaro::Modular< int32\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

### 16.211.1 Member Typedef Documentation

#### 16.211.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.212 ModeTraits< Givaro::Modular< int8\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

### 16.212.1 Member Typedef Documentation

#### 16.212.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.213 ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag >](#) value

### 16.213.1 Member Typedef Documentation

#### 16.213.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.214 ModeTraits< Givaro::Modular< uint16\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

## Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

### 16.214.1 Member Typedef Documentation

#### 16.214.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.215 ModeTraits< Givaro::Modular< uint32\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

### 16.215.1 Member Typedef Documentation

#### 16.215.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.216 ModeTraits< Givaro::Modular< uint8\_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

### 16.216.1 Member Typedef Documentation

#### 16.216.1.1 value

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.217 ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DelayedTag value](#)

### 16.217.1 Member Typedef Documentation

**16.217.1.1 value**

```
typedef ModeCategories::DelayedTag value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.218 ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

**Public Types**

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value](#)

**16.218.1 Member Typedef Documentation****16.218.1.1 value**

```
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.219 ModeTraits< Givaro::ModularBalanced< int16\_t > > Struct Reference

```
#include <field-traits.h>
```

**Public Types**

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

**16.219.1 Member Typedef Documentation****16.219.1.1 value**

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.220 ModeTraits< Givaro::ModularBalanced< int32\_t > > Struct Reference

```
#include <field-traits.h>
```

**Public Types**

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value](#)

## 16.220.1 Member Typedef Documentation

### 16.220.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.221 ModeTraits< Givaro::ModularBalanced< int8\_t > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >](#) value

## 16.221.1 Member Typedef Documentation

### 16.221.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::MachineFloatTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.222 ModeTraits< Givaro::Montgomery< T > > Struct Template Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DefaultBoundedTag](#) value

## 16.222.1 Member Typedef Documentation

### 16.222.1.1 value

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.223 ModeTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DefaultBoundedTag](#) value

### 16.223.1 Member Typedef Documentation

#### 16.223.1.1 value

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.224 ModeTraits< Givaro::ZRing< float > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::DefaultBoundedTag](#) value

### 16.224.1 Member Typedef Documentation

#### 16.224.1.1 value

typedef [ModeCategories::DefaultBoundedTag](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.225 ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

```
#include <field-traits.h>
```

### Public Types

- typedef [ModeCategories::ConvertTo< ElementCategories::RNSElementTag >](#) value

### 16.225.1 Member Typedef Documentation

#### 16.225.1.1 value

typedef [ModeCategories::ConvertTo<ElementCategories::RNSElementTag>](#) value

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.226 ModularBalanced< T > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

## 16.227 ModularTag Struct Reference

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`  
`#include <field-traits.h>`

### 16.227.1 Detailed Description

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`  
 The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.228 Montgomery< T > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

## 16.229 need\_field\_characteristic< Field > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = false

### 16.229.1 Field Documentation

#### 16.229.1.1 value

```
constexpr bool value = false [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.230 need\_field\_characteristic< Givaro::Modular< Field > > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = true

### 16.230.1 Field Documentation

#### 16.230.1.1 value

```
constexpr bool value = true [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.231 need\_field\_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference

### Static Public Attributes

- static constexpr bool [value](#) = true



### 16.231.1 Field Documentation

#### 16.231.1.1 value

`constexpr bool value = true [static], [constexpr]`

The documentation for this struct was generated from the following file:

- [benchmark-fgemv.C](#)

## 16.232 NoSimd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

### Public Types

- using [vect\\_t](#) = T \*
- using [scalar\\_t](#) = T

### Static Public Member Functions

- static const std::string [type\\_string](#) ()
- template<class TT >  
static constexpr bool [valid](#) (TT p)
- template<class TT >  
static constexpr bool [compliant](#) (TT n)

### Static Public Attributes

- static const constexpr size\_t [vect\\_size](#) = 1

### 16.232.1 Member Typedef Documentation

#### 16.232.1.1 vect\_t

```
using vect\_t = T*
```

#### 16.232.1.2 scalar\_t

```
using scalar\_t = T
```

### 16.232.2 Member Function Documentation

#### 16.232.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

#### 16.232.2.2 valid()

```
static constexpr bool valid (
    TT p ) [inline], [static], [constexpr]
```

**16.232.2.3 compliant()**

```
static constexpr bool compliant (
    TT n ) [inline], [static], [constexpr]
```

**16.232.3 Field Documentation****16.232.3.1 vect\_size**

```
const constexpr size_t vect_size = 1 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

**16.233 Parallel< C, P > Struct Template Reference****Public Types**

- typedef C [Cut](#)
- typedef P [Param](#)

**Public Member Functions**

- [Parallel](#) (size\_t n=[NUM\\_THREADS](#))
- size\_t [numthreads](#) () const
- size\_t & [set\\_numthreads](#) (size\_t n)

**Friends**

- std::ostream & [operator<<](#) (std::ostream &out, const [Parallel](#) &p)

**16.233.1 Member Typedef Documentation****16.233.1.1 Cut**

```
typedef C Cut
```

**16.233.1.2 Param**

```
typedef P Param
```

**16.233.2 Constructor & Destructor Documentation****16.233.2.1 Parallel()**

```
Parallel (
    size_t n = NUM\_THREADS ) [inline]
```

**16.233.3 Member Function Documentation**

### 16.233.3.1 numthreads()

```
size_t numthreads ( ) const [inline]
```

### 16.233.3.2 set\_numthreads()

```
size_t & set_numthreads (
    size_t n ) [inline]
```

## 16.233.4 Friends And Related Function Documentation

### 16.233.4.1 operator<<

```
std::ostream & operator<< (
    std::ostream & out,
    const Parallel< C, P > & p ) [friend]
```

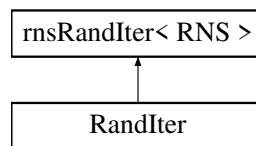
The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.234 RNSInteger< RNS >::RandIter Class Reference

```
#include <rns-integer.h>
```

Inheritance diagram for RNSInteger< RNS >::RandIter:



## Public Member Functions

- [RandIter](#) (const [RNSInteger< RNS >](#) &F, size\_t size=0, uint64\_t seed=0)
- [RNS::Element & random](#) (typename [RNS::Element](#) &elt) const  
*RNS ring Element random assignement.*
- [RNS::Element random](#) () const
- [RNS::Element & operator\(\)](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element operator\(\)](#) () const
- const [RNS](#) & [ring](#) () const

### 16.234.1 Constructor & Destructor Documentation

#### 16.234.1.1 RandIter()

```
RandIter (
    const RNSInteger< RNS > & F,
    size_t size = 0,
    uint64_t seed = 0 ) [inline]
```

### 16.234.2 Member Function Documentation

**16.234.2.1 random() [1/2]**

```
RNS::Element & random (
    typename RNS::Element & elt ) const [inline], [inherited]
```

RNS ring Element random assignment.

Element is supposed to be initialized

Returns

random ring Element

**16.234.2.2 random() [2/2]**

```
RNS::Element random ( ) const [inline], [inherited]
```

**16.234.2.3 operator>() [1/2]**

```
RNS::Element & operator() (
    typename RNS::Element & elt ) const [inline], [inherited]
```

**16.234.2.4 operator>() [2/2]**

```
RNS::Element operator() ( ) const [inline], [inherited]
```

**16.234.2.5 ring()**

```
const RNS & ring ( ) const [inline], [inherited]
```

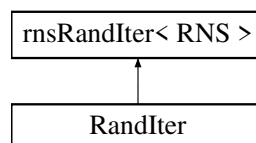
The documentation for this class was generated from the following file:

- [rns-integer.h](#)

**16.235 RNSIntegerMod< RNS >::RandIter Class Reference**

```
#include <rns-integer-mod.h>
```

Inheritance diagram for RNSIntegerMod< RNS >::RandIter:

**Public Member Functions**

- [RandIter](#) (const [RNSIntegerMod< RNS >](#) &F, [size\\_t](#) size=0, [uint64\\_t](#) seed=0)
- [RNS::Element & random](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element random](#) () const
- [RNS::Element & operator\(\)](#) (typename [RNS::Element](#) &elt) const
- [RNS::Element operator\(\)](#) () const
- const [RNS](#) & [ring](#) () const

**16.235.1 Constructor & Destructor Documentation**

**16.235.1.1 RandIter()**

```
RandIter (
    const RNSIntegerMod< RNS > & F,
    size_t size = 0,
    uint64_t seed = 0 ) [inline]
```

**16.235.2 Member Function Documentation****16.235.2.1 random() [1/2]**

```
RNS::Element & random (
    typename RNS::Element & elt ) const [inline]
```

**16.235.2.2 random() [2/2]**

```
RNS::Element random ( ) const [inline], [inherited]
```

**16.235.2.3 operator>() [1/2]**

```
RNS::Element & operator() (
    typename RNS::Element & elt ) const [inline], [inherited]
```

**16.235.2.4 operator>() [2/2]**

```
RNS::Element operator() ( ) const [inline], [inherited]
```

**16.235.2.5 ring()**

```
const RNS & ring ( ) const [inline], [inherited]
```

The documentation for this class was generated from the following file:

- [rns-integer-mod.h](#)

**16.236 readMyMachineType< Field, T > Struct Template Reference**

```
#include <read_sparse.h>
```

**Public Types**

- typedef [Field::Element](#) [Element](#)
- typedef [Field::Element\\_ptr](#) [Element\\_ptr](#)

**Public Member Functions**

- void [operator\(\)](#) (const [Field](#) &F, [Element](#) &modulo, [Element\\_ptr](#) val, std::ifstream &file, const [uint64\\_t](#) dims, const [mask\\_t](#) data\_type, const [mask\\_t](#) field\_desc)

**16.236.1 Member Typedef Documentation**

### 16.236.1.1 Element

```
typedef Field::Element Element
```

### 16.236.1.2 Element\_ptr

```
typedef Field::Element_ptr Element_ptr
```

## 16.236.2 Member Function Documentation

### 16.236.2.1 operator>()

```
void operator() (
    const Field & F,
    Element & modulo,
    Element_ptr val,
    std::ifstream & file,
    const uint64_t dims,
    const mask_t data_type,
    const mask_t field_desc )
```

The documentation for this struct was generated from the following file:

- [read\\_sparse.h](#)

## 16.237 readMyMachineType< Field, mpz\_t > Struct Template Reference

```
#include <read_sparse.h>
```

### Public Types

- typedef [Field::Element](#) [Element](#)
- typedef [Field::Element\\_ptr](#) [Element\\_ptr](#)

### Public Member Functions

- void [operator\(\)](#) (const [Field](#) &F, [Element](#) &modulo, [Element\\_ptr](#) val, std::ifstream &file, const [uint64\\_t](#) dims, const [mask\\_t](#) data\_type, const [mask\\_t](#) field\_desc)

## 16.237.1 Member Typedef Documentation

### 16.237.1.1 Element

```
typedef Field::Element Element
```

### 16.237.1.2 Element\_ptr

```
typedef Field::Element_ptr Element_ptr
```

## 16.237.2 Member Function Documentation

**16.237.2.1 operator()()**

```
void operator() (
    const Field & F,
    typename Field::Element & modulo,
    typename Field::Element_ptr val,
    std::ifstream & file,
    const uint64_t dims,
    const mask_t data_type,
    const mask_t field_desc )
```

The documentation for this struct was generated from the following file:

- [read\\_sparse.h](#)

**16.238 Recursive Struct Reference**

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

**16.239 Recursive Struct Reference**

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

**16.240 rint< K > Class Template Reference**

The documentation for this class was generated from the following file:

- [field-traits.h](#)

**16.241 rns\_double Struct Reference**

```
#include <rns-double.h>
```

**Public Types**

- typedef Givaro::Integer [integer](#)
- typedef Givaro::Modular< double > [ModField](#)
- typedef double [BasisElement](#)
- typedef [rns\\_double\\_elt](#) [Element](#)
- typedef [rns\\_double\\_elt\\_ptr](#) [Element\\_ptr](#)
- typedef [rns\\_double\\_elt\\_cstptr](#) [ConstElement\\_ptr](#)

**Public Member Functions**

- [rns\\_double](#) (const [integer](#) &bound, size\_t pbits, bool rnsmod=false, long seed=time(NULL))
- [rns\\_double](#) (size\_t pbits, size\_t size, long seed=time(NULL))
- template<typename Vect >  
  [rns\\_double](#) (const Vect &basis, bool rnsmod=false, long seed=time(NULL))
- [rns\\_double](#) (const [RNSIntegerMod](#)< [rns\\_double](#) > &basis, bool rnsmod=false, long seed=time(NULL))
- void [precompute\\_cst](#) (size\_t K=0)
- template<typename T >  
  void [init](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const T \*A, size\_t lda, const [integer](#) &maxA, bool RNS\_MAJOR=false) const

- void `init` (size\_t m, size\_t n, double \*Arns, size\_t rda, const integer \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false) const
- void `init_transpose` (size\_t m, size\_t n, double \*Arns, size\_t rda, const integer \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false) const
- void `convert` (size\_t m, size\_t n, integer gamma, integer \*A, size\_t lda, const double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const
- void `convert_transpose` (size\_t m, size\_t n, integer gamma, integer \*A, size\_t lda, const double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const
- void `reduce` (size\_t n, double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const
- template<size\_t K>  
void `init` (size\_t m, size\_t n, double \*Arns, size\_t rda, const Reclnt::ruint< K > \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false) const
- template<size\_t K>  
void `convert` (size\_t m, size\_t n, integer gamma, Reclnt::ruint< K > \*A, size\_t lda, const double \*Arns, size\_t rda, integer p=0, bool RNS\_MAJOR=false) const

## Data Fields

- std::vector< double, AlignedAllocator< double, Alignment::CACHE\_LINE > > `_basis`
- std::vector< double, AlignedAllocator< double, Alignment::CACHE\_LINE > > `_basisMax`
- std::vector< double, AlignedAllocator< double, Alignment::CACHE\_LINE > > `_negbasis`
- std::vector< double, AlignedAllocator< double, Alignment::CACHE\_LINE > > `_invbasis`
- std::vector< ModField > `_field_rns`
- integer `_M`
- std::vector< integer > `_Mi`
- std::vector< double > `_MMi`
- std::vector< double > `_crt_in`
- std::vector< double > `_crt_out`
- size\_t `_size`
- size\_t `_pbits`
- size\_t `_ldm`
- integer `_mi_sum`

## 16.241.1 Member Typedef Documentation

### 16.241.1.1 integer

```
typedef Givaro::Integer integer
```

### 16.241.1.2 ModField

```
typedef Givaro::Modular<double> ModField
```

### 16.241.1.3 BasisElement

```
typedef double BasisElement
```

### 16.241.1.4 Element

```
typedef rns_double_elt Element
```



### 16.241.1.5 Element\_ptr

```
typedef rns_double_elt_ptr Element_ptr
```

### 16.241.1.6 ConstElement\_ptr

```
typedef rns_double_elt_cstptr ConstElement_ptr
```

## 16.241.2 Constructor & Destructor Documentation

### 16.241.2.1 rns\_double() [1/4]

```
rns_double (
    const integer & bound,
    size_t pbits,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

### 16.241.2.2 rns\_double() [2/4]

```
rns_double (
    size_t pbits,
    size_t size,
    long seed = time(NULL) ) [inline]
```

### 16.241.2.3 rns\_double() [3/4]

```
rns_double (
    const Vect & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

### 16.241.2.4 rns\_double() [4/4]

```
rns_double (
    const RNSIntegerMod< rns_double > & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

## 16.241.3 Member Function Documentation

### 16.241.3.1 precompute\_cst()

```
void precompute_cst (
    size_t K = 0 ) [inline]
```

### 16.241.3.2 init() [1/3]

```
void init (
    size_t m,
    size_t n,
```

```
double * Arns,  
size_t rda,  
const T * A,  
size_t lda,  
const integer & maxA,  
bool RNS_MAJOR = false ) const [inline]
```

#### 16.241.3.3 init() [2/3]

```
void init (  
    size_t m,  
    size_t n,  
    double * Arns,  
    size_t rda,  
    const integer * A,  
    size_t lda,  
    size_t k,  
    bool RNS_MAJOR = false ) const [inline]
```

#### 16.241.3.4 init\_transpose()

```
void init_transpose (  
    size_t m,  
    size_t n,  
    double * Arns,  
    size_t rda,  
    const integer * A,  
    size_t lda,  
    size_t k,  
    bool RNS_MAJOR = false ) const [inline]
```

#### 16.241.3.5 convert() [1/2]

```
void convert (  
    size_t m,  
    size_t n,  
    integer gamma,  
    integer * A,  
    size_t lda,  
    const double * Arns,  
    size_t rda,  
    bool RNS_MAJOR = false ) const [inline]
```

#### 16.241.3.6 convert\_transpose()

```
void convert_transpose (  
    size_t m,  
    size_t n,  
    integer gamma,  
    integer * A,  
    size_t lda,  
    const double * Arns,  
    size_t rda,  
    bool RNS_MAJOR = false ) const [inline]
```

**16.241.3.7 reduce()**

```
void reduce (
    size_t n,
    double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]
```

**16.241.3.8 init() [3/3]**

```
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const RecInt::ruint< K > * A,
    size_t lda,
    size_t k,
    bool RNS_MAJOR = false ) const [inline]
```

**16.241.3.9 convert() [2/2]**

```
void convert (
    size_t m,
    size_t n,
    integer gamma,
    RecInt::ruint< K > * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    integer p = 0,
    bool RNS_MAJOR = false ) const [inline]
```

**16.241.4 Field Documentation****16.241.4.1 \_basis**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

**16.241.4.2 \_basisMax**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

**16.241.4.3 \_negbasis**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

**16.241.4.4 \_invbasis**

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

#### 16.241.4.5 `_field_rns`

```
std::vector<ModField> _field_rns
```

#### 16.241.4.6 `_M`

```
integer _M
```

#### 16.241.4.7 `_Mi`

```
std::vector<integer> _Mi
```

#### 16.241.4.8 `_MMi`

```
std::vector<double> _MMi
```

#### 16.241.4.9 `_crt_in`

```
std::vector<double> _crt_in
```

#### 16.241.4.10 `_crt_out`

```
std::vector<double> _crt_out
```

#### 16.241.4.11 `_size`

```
size_t _size
```

#### 16.241.4.12 `_pbits`

```
size_t _pbits
```

#### 16.241.4.13 `_ldm`

```
size_t _ldm
```

#### 16.241.4.14 `_mi_sum`

```
integer _mi_sum
```

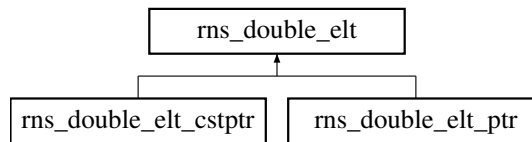
The documentation for this struct was generated from the following files:

- [rns-double.h](#)
- [rns-double-recint.inl](#)
- [rns-double.inl](#)

## 16.242 `rns_double_elt` Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for `rns_double_elt`:



## Public Member Functions

- [rns\\_double\\_elt](#) ()
- [~rns\\_double\\_elt](#) ()
- [rns\\_double\\_elt](#) (double \*p, size\_t r, size\_t a=false)
- [rns\\_double\\_elt\\_ptr](#) operator& ()
- [rns\\_double\\_elt\\_cstptr](#) operator& () const
- [rns\\_double\\_elt](#) (const [rns\\_double\\_elt](#) &x)

## Data Fields

- double \* [\\_ptr](#)
- size\_t [\\_stride](#)
- bool [\\_alloc](#)

## 16.242.1 Constructor & Destructor Documentation

### 16.242.1.1 [rns\\_double\\_elt](#)() [1/3]

```
rns\_double\_elt ( ) [inline]
```

### 16.242.1.2 [~rns\\_double\\_elt](#)()

```
~rns\_double\_elt ( ) [inline]
```

### 16.242.1.3 [rns\\_double\\_elt](#)() [2/3]

```
rns\_double\_elt (
    double * p,
    size_t r,
    size_t a = false ) [inline]
```

### 16.242.1.4 [rns\\_double\\_elt](#)() [3/3]

```
rns\_double\_elt (
    const rns\_double\_elt & x ) [inline]
```

## 16.242.2 Member Function Documentation

### 16.242.2.1 [operator&\(\)](#) [1/2]

```
rns\_double\_elt\_ptr operator& ( ) [inline]
```

### 16.242.2.2 operator&() [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline]
```

## 16.242.3 Field Documentation

### 16.242.3.1 \_ptr

```
double* _ptr
```

### 16.242.3.2 \_stride

```
size_t _stride
```

### 16.242.3.3 \_alloc

```
bool _alloc
```

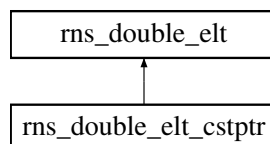
The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

## 16.243 rns\_double\_elt\_cstptr Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for rns\_double\_elt\_cstptr:



## Public Member Functions

- [rns\\_double\\_elt\\_cstptr](#) ()
- [rns\\_double\\_elt\\_cstptr](#) (double \*p, size\_t r)
- [rns\\_double\\_elt\\_cstptr](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- [rns\\_double\\_elt\\_cstptr](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- [rns\\_double\\_elt\\_cstptr](#) ([rns\\_double\\_elt\\_cstptr](#) &&)=default
- [rns\\_double\\_elt\\_cstptr](#) \* [operator&](#) ()
- [rns\\_double\\_elt](#) & [operator\\*](#) () const
- [rns\\_double\\_elt](#) [operator\[\]](#) (size\_t i) const
- [rns\\_double\\_elt](#) & [operator\[\]](#) (size\_t i)
- [rns\\_double\\_elt\\_cstptr](#) [operator++](#) ()
- [rns\\_double\\_elt\\_cstptr](#) [operator--](#) ()
- [rns\\_double\\_elt\\_cstptr](#) [operator+](#) (size\_t inc) const
- [rns\\_double\\_elt\\_cstptr](#) [operator-](#) (size\_t inc) const
- [rns\\_double\\_elt\\_cstptr](#) & [operator+=](#) (size\_t inc)
- [rns\\_double\\_elt\\_cstptr](#) & [operator-=](#) (size\_t inc)
- [rns\\_double\\_elt\\_cstptr](#) & [operator=](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- bool [operator<](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- bool [operator!=](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- [rns\\_double\\_elt\\_cstptr](#) [operator&](#) () const

## Data Fields

- [rns\\_double\\_elt](#) other
- `double * \_ptr`
- `size_t \_stride`
- `bool \_alloc`

## 16.243.1 Constructor & Destructor Documentation

### 16.243.1.1 `rns_double_elt_cstptr()` [1/5]

```
rns_double_elt_cstptr ( ) [inline]
```

### 16.243.1.2 `rns_double_elt_cstptr()` [2/5]

```
rns_double_elt_cstptr (
    double * p,
    size_t r ) [inline]
```

### 16.243.1.3 `rns_double_elt_cstptr()` [3/5]

```
rns_double_elt_cstptr (
    const rns\_double\_elt\_ptr & x ) [inline]
```

### 16.243.1.4 `rns_double_elt_cstptr()` [4/5]

```
rns_double_elt_cstptr (
    const rns\_double\_elt\_cstptr & x ) [inline]
```

### 16.243.1.5 `rns_double_elt_cstptr()` [5/5]

```
rns_double_elt_cstptr (
    rns\_double\_elt\_cstptr && ) [default]
```

## 16.243.2 Member Function Documentation

### 16.243.2.1 `operator&()` [1/2]

```
rns\_double\_elt\_cstptr * operator& ( ) [inline]
```

### 16.243.2.2 `operator*()`

```
rns\_double\_elt & operator* ( ) const [inline]
```

### 16.243.2.3 `operator[]()` [1/2]

```
rns\_double\_elt operator[] (
    size_t i ) const [inline]
```

**16.243.2.4 operator[]()** [2/2]

```
rns_double_elt & operator[] (
    size_t i ) [inline]
```

**16.243.2.5 operator++()**

```
rns_double_elt_cstptr operator++ ( ) [inline]
```

**16.243.2.6 operator--()**

```
rns_double_elt_cstptr operator-- ( ) [inline]
```

**16.243.2.7 operator+()**

```
rns_double_elt_cstptr operator+ (
    size_t inc ) const [inline]
```

**16.243.2.8 operator-()**

```
rns_double_elt_cstptr operator- (
    size_t inc ) const [inline]
```

**16.243.2.9 operator+=()**

```
rns_double_elt_cstptr & operator+= (
    size_t inc ) [inline]
```

**16.243.2.10 operator-=()**

```
rns_double_elt_cstptr & operator-= (
    size_t inc ) [inline]
```

**16.243.2.11 operator=()**

```
rns_double_elt_cstptr & operator= (
    const rns_double_elt_cstptr & x ) [inline]
```

**16.243.2.12 operator<()**

```
bool operator< (
    const rns_double_elt_cstptr & x ) [inline]
```

**16.243.2.13 operator"!="()**

```
bool operator!= (
    const rns_double_elt_cstptr & x ) [inline]
```

**16.243.2.14 operator&()** [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline], [inherited]
```



### 16.243.3 Field Documentation

#### 16.243.3.1 other

`rns_double_elt` other

#### 16.243.3.2 \_ptr

`double* _ptr` [inherited]

#### 16.243.3.3 \_stride

`size_t _stride` [inherited]

#### 16.243.3.4 \_alloc

`bool _alloc` [inherited]

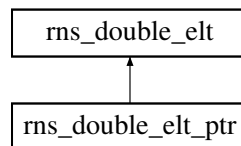
The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

## 16.244 rns\_double\_elt\_ptr Struct Reference

```
#include <rns-double-elt.h>
```

Inheritance diagram for `rns_double_elt_ptr`:



### Public Member Functions

- [rns\\_double\\_elt\\_ptr](#) ()
- [rns\\_double\\_elt\\_ptr](#) (double \*p, size\_t r)
- [rns\\_double\\_elt\\_ptr](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- [rns\\_double\\_elt\\_ptr](#) (const [rns\\_double\\_elt\\_cstptr](#) &x)
- [rns\\_double\\_elt\\_ptr](#) ([rns\\_double\\_elt\\_ptr](#) &&)=default
- [rns\\_double\\_elt\\_ptr](#) \* [operator&](#) ()
- [rns\\_double\\_elt](#) & [operator\\*](#) ()
- [rns\\_double\\_elt](#) [operator\[\]](#) (size\_t i) const
- [rns\\_double\\_elt](#) & [operator\[\]](#) (size\_t i)
- [rns\\_double\\_elt\\_ptr](#) [operator++](#) ()
- [rns\\_double\\_elt\\_ptr](#) [operator--](#) ()
- [rns\\_double\\_elt\\_ptr](#) [operator+](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) [operator-](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) & [operator+=](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) & [operator-=](#) (size\_t inc)
- [rns\\_double\\_elt\\_ptr](#) & [operator=](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- bool [operator<](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- bool [operator!=](#) (const [rns\\_double\\_elt\\_ptr](#) &x)
- [rns\\_double\\_elt\\_cstptr](#) [operator&](#) () const

## Data Fields

- [rns\\_double\\_elt](#) other
- `double * _ptr`
- `size_t _stride`
- `bool _alloc`

## 16.244.1 Constructor & Destructor Documentation

### 16.244.1.1 `rns_double_elt_ptr()` [1/5]

```
rns_double_elt_ptr ( ) [inline]
```

### 16.244.1.2 `rns_double_elt_ptr()` [2/5]

```
rns_double_elt_ptr (
    double * p,
    size_t r ) [inline]
```

### 16.244.1.3 `rns_double_elt_ptr()` [3/5]

```
rns_double_elt_ptr (
    const rns\_double\_elt\_ptr & x ) [inline]
```

### 16.244.1.4 `rns_double_elt_ptr()` [4/5]

```
rns_double_elt_ptr (
    const rns\_double\_elt\_cstptr & x ) [inline]
```

### 16.244.1.5 `rns_double_elt_ptr()` [5/5]

```
rns_double_elt_ptr (
    rns\_double\_elt\_ptr && ) [default]
```

## 16.244.2 Member Function Documentation

### 16.244.2.1 `operator&()` [1/2]

```
rns\_double\_elt\_ptr * operator& ( ) [inline]
```

### 16.244.2.2 `operator*()`

```
rns\_double\_elt & operator* ( ) [inline]
```

### 16.244.2.3 `operator[]()` [1/2]

```
rns\_double\_elt operator[] (
    size_t i ) const [inline]
```

**16.244.2.4 operator[]()** [2/2]

```
rns_double_elt & operator[] (
    size_t i ) [inline]
```

**16.244.2.5 operator++()**

```
rns_double_elt_ptr operator++ ( ) [inline]
```

**16.244.2.6 operator--()**

```
rns_double_elt_ptr operator-- ( ) [inline]
```

**16.244.2.7 operator+()**

```
rns_double_elt_ptr operator+ (
    size_t inc ) [inline]
```

**16.244.2.8 operator-()**

```
rns_double_elt_ptr operator- (
    size_t inc ) [inline]
```

**16.244.2.9 operator+=()**

```
rns_double_elt_ptr & operator+= (
    size_t inc ) [inline]
```

**16.244.2.10 operator-=()**

```
rns_double_elt_ptr & operator-= (
    size_t inc ) [inline]
```

**16.244.2.11 operator=()**

```
rns_double_elt_ptr & operator= (
    const rns_double_elt_ptr & x ) [inline]
```

**16.244.2.12 operator<()**

```
bool operator< (
    const rns_double_elt_ptr & x ) [inline]
```

**16.244.2.13 operator"!="()**

```
bool operator!= (
    const rns_double_elt_ptr & x ) [inline]
```

**16.244.2.14 operator&()** [2/2]

```
rns_double_elt_cstptr operator& ( ) const [inline], [inherited]
```

### 16.244.3 Field Documentation

#### 16.244.3.1 other

`rns_double_elt` other

#### 16.244.3.2 \_ptr

`double* _ptr` [inherited]

#### 16.244.3.3 \_stride

`size_t _stride` [inherited]

#### 16.244.3.4 \_alloc

`bool _alloc` [inherited]

The documentation for this struct was generated from the following file:

- [rns-double-elt.h](#)

## 16.245 `rns_double_extended` Struct Reference

```
#include <rns-double.h>
```

### Public Types

- typedef Givaro::Integer [integer](#)
- typedef Givaro::ModularExtended< double > [ModField](#)
- typedef double [BasisElement](#)
- typedef [rns\\_double\\_elt](#) [Element](#)
- typedef [rns\\_double\\_elt\\_ptr](#) [Element\\_ptr](#)
- typedef [rns\\_double\\_elt\\_cstptr](#) [ConstElement\\_ptr](#)

### Public Member Functions

- [rns\\_double\\_extended](#) (const [integer](#) &bound, size\_t pbits, bool rnsmod=false, long seed=time(NULL))
- [rns\\_double\\_extended](#) (size\_t pbits, size\_t size, long seed=time(NULL))
- template<typename Vect >  
  [rns\\_double\\_extended](#) (const Vect &basis, bool rnsmod=false, long seed=time(NULL))
- void [precompute\\_cst](#) ()
- void [init](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const [integer](#) \*A, size\_t lda, const [integer](#) &maxA, bool RNS\_MAJOR=false) const
- void [init](#) (size\_t m, size\_t n, double \*Arns, size\_t rda, const [integer](#) \*A, size\_t lda, size\_t k, bool RNS\_MAJOR=false)
- void [convert](#) (size\_t m, size\_t n, [integer](#) gamma, [integer](#) \*A, size\_t lda, const double \*Arns, size\_t rda, bool RNS\_MAJOR=false)
- void [init](#) (size\_t m, double \*Arns, const [integer](#) \*A, size\_t lda) const
- void [convert](#) (size\_t m, [integer](#) \*A, const double \*Arns) const
- void [reduce](#) (size\_t n, double \*Arns, size\_t rda, bool RNS\_MAJOR=false) const

## Data Fields

- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basis`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _basisMax`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _negbasis`
- `std::vector< double, AlignedAllocator< double, Alignment::CACHE_LINE > > _invbasis`
- `std::vector< ModField > _field_rns`
- `integer _M`
- `std::vector< integer > _Mi`
- `std::vector< double > _MMi`
- `std::vector< double > _crt_in`
- `std::vector< double > _crt_out`
- `size_t _size`
- `size_t _pbits`
- `size_t _ldm`

## 16.245.1 Member Typedef Documentation

### 16.245.1.1 integer

```
typedef Givaro::Integer integer
```

### 16.245.1.2 ModField

```
typedef Givaro::ModularExtended<double> ModField
```

### 16.245.1.3 BasisElement

```
typedef double BasisElement
```

### 16.245.1.4 Element

```
typedef rns\_double\_elt Element
```

### 16.245.1.5 Element\_ptr

```
typedef rns\_double\_elt\_ptr Element\_ptr
```

### 16.245.1.6 ConstElement\_ptr

```
typedef rns\_double\_elt\_cstptr ConstElement\_ptr
```

## 16.245.2 Constructor & Destructor Documentation

**16.245.2.1 rns\_double\_extended() [1/3]**

```
rns_double_extended (
    const integer & bound,
    size_t pbits,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

**16.245.2.2 rns\_double\_extended() [2/3]**

```
rns_double_extended (
    size_t pbits,
    size_t size,
    long seed = time(NULL) ) [inline]
```

**16.245.2.3 rns\_double\_extended() [3/3]**

```
rns_double_extended (
    const Vect & basis,
    bool rnsmod = false,
    long seed = time(NULL) ) [inline]
```

**16.245.3 Member Function Documentation****16.245.3.1 precompute\_cst()**

```
void precompute_cst ( ) [inline]
```

**16.245.3.2 init() [1/3]**

```
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const integer * A,
    size_t lda,
    const integer & maxA,
    bool RNS_MAJOR = false ) const [inline]
```

**16.245.3.3 init() [2/3]**

```
void init (
    size_t m,
    size_t n,
    double * Arns,
    size_t rda,
    const integer * A,
    size_t lda,
    size_t k,
    bool RNS_MAJOR = false ) [inline]
```

#### 16.245.3.4 convert() [1/2]

```
void convert (
    size_t m,
    size_t n,
    integer gamma,
    integer * A,
    size_t lda,
    const double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) [inline]
```

#### 16.245.3.5 init() [3/3]

```
void init (
    size_t m,
    double * Arns,
    const integer * A,
    size_t lda ) const [inline]
```

#### 16.245.3.6 convert() [2/2]

```
void convert (
    size_t m,
    integer * A,
    const double * Arns ) const [inline]
```

#### 16.245.3.7 reduce()

```
void reduce (
    size_t n,
    double * Arns,
    size_t rda,
    bool RNS_MAJOR = false ) const [inline]
```

### 16.245.4 Field Documentation

#### 16.245.4.1 \_basis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

#### 16.245.4.2 \_basisMax

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

#### 16.245.4.3 \_negbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

#### 16.245.4.4 `_invbasis`

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

#### 16.245.4.5 `_field_rns`

```
std::vector<ModField> _field_rns
```

#### 16.245.4.6 `_M`

```
integer _M
```

#### 16.245.4.7 `_Mi`

```
std::vector<integer> _Mi
```

#### 16.245.4.8 `_MMi`

```
std::vector<double> _MMi
```

#### 16.245.4.9 `_crt_in`

```
std::vector<double> _crt_in
```

#### 16.245.4.10 `_crt_out`

```
std::vector<double> _crt_out
```

#### 16.245.4.11 `_size`

```
size_t _size
```

#### 16.245.4.12 `_pbits`

```
size_t _pbits
```

#### 16.245.4.13 `_ldm`

```
size_t _ldm
```

The documentation for this struct was generated from the following files:

- [rns-double.h](#)
- [rns-double.inl](#)

## 16.246 RNSElementTag Struct Reference

Representation in a Residue Number System.

```
#include <field-traits.h>
```



### 16.246.1 Detailed Description

Representation in a Residue Number System.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

## 16.247 RNSInteger< RNS > Class Template Reference

```
#include <rns-integer.h>
```

### Data Structures

- class [RandIter](#)

### Public Types

- typedef [RNS::Element](#) [Element](#)
- typedef [RNS::Element\\_ptr](#) [Element\\_ptr](#)
- typedef [RNS::ConstElement\\_ptr](#) [ConstElement\\_ptr](#)

### Public Member Functions

- [RNSInteger](#) (const [RNS](#) &myrns)
- template<typename T >  
  [RNSInteger](#) (const T &F)
- const [RNS](#) & [rns](#) () const
- size\_t [size](#) () const
- bool [isOne](#) (const [Element](#) &x) const
- bool [isMOne](#) (const [Element](#) &x) const
- bool [isZero](#) (const [Element](#) &x) const
- integer [characteristic](#) ([integer](#) &p) const
- integer [cardinality](#) ([integer](#) &p) const
- [Element](#) & [init](#) ([Element](#) &x) const
- [Element](#) & [init](#) ([Element](#) &x, const Givaro::Integer &y) const
- [Element](#) & [reduce](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) & [reduce](#) ([Element](#) &x) const
- Givaro::Integer [convert](#) (Givaro::Integer &x, const [Element](#) &y) const
- [Element](#) & [assign](#) ([Element](#) &x, const [Element](#) &y) const
- std::ostream & [write](#) (std::ostream &os, const [Element](#) &y) const
- std::ostream & [write](#) (std::ostream &os) const

### Data Fields

- [Element](#) [one](#)
- [Element](#) [mOne](#)
- [Element](#) [zero](#)

### Protected Types

- typedef [RNS::BasisElement](#) [BasisElement](#)
- typedef Givaro::Integer [integer](#)

### Protected Attributes

- const [RNS](#) \* [\\_rns](#)

## 16.247.1 Member Typedef Documentation

### 16.247.1.1 BasisElement

```
typedef RNS::BasisElement BasisElement [protected]
```

### 16.247.1.2 integer

```
typedef Givaro::Integer integer [protected]
```

### 16.247.1.3 Element

```
typedef RNS::Element Element
```

### 16.247.1.4 Element\_ptr

```
typedef RNS::Element_ptr Element_ptr
```

### 16.247.1.5 ConstElement\_ptr

```
typedef RNS::ConstElement_ptr ConstElement_ptr
```

## 16.247.2 Constructor & Destructor Documentation

### 16.247.2.1 RNSInteger() [1/2]

```
RNSInteger (
    const RNS & myrns ) [inline]
```

### 16.247.2.2 RNSInteger() [2/2]

```
RNSInteger (
    const T & F ) [inline]
```

## 16.247.3 Member Function Documentation

### 16.247.3.1 rns()

```
const RNS & rns ( ) const [inline]
```

### 16.247.3.2 size()

```
size_t size ( ) const [inline]
```

### 16.247.3.3 isOne()

```
bool isOne (
    const Element & x ) const [inline]
```

### 16.247.3.4 isMOne()

```
bool isMOne (
    const Element & x ) const [inline]
```

### 16.247.3.5 isZero()

```
bool isZero (
    const Element & x ) const [inline]
```

### 16.247.3.6 characteristic()

```
integer characteristic (
    integer & p ) const [inline]
```

### 16.247.3.7 cardinality()

```
integer cardinality (
    integer & p ) const [inline]
```

### 16.247.3.8 init() [1/2]

```
Element & init (
    Element & x ) const [inline]
```

### 16.247.3.9 init() [2/2]

```
Element & init (
    Element & x,
    const Givaro::Integer & y ) const [inline]
```

### 16.247.3.10 reduce() [1/2]

```
Element & reduce (
    Element & x,
    const Element & y ) const [inline]
```

### 16.247.3.11 reduce() [2/2]

```
Element & reduce (
    Element & x ) const [inline]
```

**16.247.3.12 convert()**

```
Givaro::Integer convert (
    Givaro::Integer & x,
    const Element & y ) const [inline]
```

**16.247.3.13 assign()**

```
Element & assign (
    Element & x,
    const Element & y ) const [inline]
```

**16.247.3.14 write() [1/2]**

```
std::ostream & write (
    std::ostream & os,
    const Element & y ) const [inline]
```

**16.247.3.15 write() [2/2]**

```
std::ostream & write (
    std::ostream & os ) const [inline]
```

**16.247.4 Field Documentation****16.247.4.1 \_rns**

```
const RNS* _rns [protected]
```

**16.247.4.2 one**

```
Element one
```

**16.247.4.3 mOne**

```
Element mOne
```

**16.247.4.4 zero**

```
Element zero
```

The documentation for this class was generated from the following files:

- [field-traits.h](#)
- [rns-integer.h](#)

**16.248 RNSIntegerMod< RNS > Class Template Reference**

```
#include <rns-integer-mod.h>
```

**Data Structures**

- class [RandIter](#)

## Public Types

- typedef [RNS::Element](#) [Element](#)
- typedef [RNS::Element\\_ptr](#) [Element\\_ptr](#)
- typedef [RNS::ConstElement\\_ptr](#) [ConstElement\\_ptr](#)

## Public Member Functions

- [RNSIntegerMod](#) (const [integer](#) &p, const [RNS](#) &myrns)
- const [rns\\_double](#) &[rns](#) () const
- const [RNSInteger](#)< [RNS](#) > &[delayed](#) () const
- [size\\_t](#) [size](#) () const
- bool [isOne](#) (const [Element](#) &x) const
- bool [isMOne](#) (const [Element](#) &x) const
- bool [isZero](#) (const [Element](#) &x) const
- [integer](#) &[characteristic](#) ([integer](#) &p) const
- [integer](#) [characteristic](#) () const
- [integer](#) &[cardinality](#) ([integer](#) &p) const
- [integer](#) [cardinality](#) () const
- [integer](#) [minElement](#) () const
- [integer](#) [maxElement](#) () const
- [Element](#) &[init](#) ([Element](#) &x) const
- [Element](#) &[init](#) ([Element](#) &x, const Givaro::Integer &y) const
- [Element](#) &[reduce](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) &[reduce](#) ([Element](#) &x) const
- [Element](#) &[init](#) ([Element](#) &x, const [Element](#) &y) const
- Givaro::Integer [convert](#) (Givaro::Integer &x, const [Element](#) &y) const
- [Element](#) &[assign](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) &[add](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[sub](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[neg](#) ([Element](#) &x, const [Element](#) &y) const
- [Element](#) &[mul](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[axpyin](#) ([Element](#) &x, const [Element](#) &y, const [Element](#) &z) const
- [Element](#) &[inv](#) ([Element](#) &x, const [Element](#) &y) const
- bool [areEqual](#) (const [Element](#) &x, const [Element](#) &y) const
- std::ostream &[write](#) (std::ostream &os, const [Element](#) &y) const
- std::ostream &[write](#) (std::ostream &os) const
- void [reduce\\_modp](#) ([size\\_t](#) n, [Element\\_ptr](#) B) const
- std::ostream &[write\\_matrix](#) (std::ostream &c, const double \*E, int n, int m, int lda) const
- std::ostream &[write\\_matrix\\_long](#) (std::ostream &c, const double \*E, int n, int m, int lda) const
- void [reduce\\_modp](#) ([size\\_t](#) m, [size\\_t](#) n, [Element\\_ptr](#) B, [size\\_t](#) lda) const
- void [reduce\\_modp\\_rnsmajor](#) ([size\\_t](#) n, [Element\\_ptr](#) B) const

## Data Fields

- [Element](#) [one](#)
- [Element](#) [mOne](#)
- [Element](#) [zero](#)

## Protected Types

- typedef [RNS::BasisElement](#) [BasisElement](#)
- typedef Givaro::Modular< [BasisElement](#) > [ModField](#)
- typedef Givaro::Integer [integer](#)

## Protected Attributes

- `integer _p`
- `std::vector< BasisElement, AlignedAllocator< BasisElement, Alignment::CACHE_LINE > > _Mi_modp_rns`
- `std::vector< BasisElement, AlignedAllocator< BasisElement, Alignment::CACHE_LINE > > _iM_modp_rns`
- `const RNS * _rns`
- `Givaro::Modular< Givaro::Integer > _F`
- `RNSInteger< RNS > _RNSdelayed`

## 16.248.1 Member Typedef Documentation

### 16.248.1.1 Element

```
typedef RNS::Element Element
```

### 16.248.1.2 Element\_ptr

```
typedef RNS::Element\_ptr Element_ptr
```

### 16.248.1.3 ConstElement\_ptr

```
typedef RNS::ConstElement\_ptr ConstElement_ptr
```

### 16.248.1.4 BasisElement

```
typedef RNS::BasisElement BasisElement [protected]
```

### 16.248.1.5 ModField

```
typedef Givaro::Modular<BasisElement> ModField [protected]
```

### 16.248.1.6 integer

```
typedef Givaro::Integer integer [protected]
```

## 16.248.2 Constructor & Destructor Documentation

### 16.248.2.1 RNSIntegerMod()

```
RNSIntegerMod (
    const integer & p,
    const RNS & myrns ) [inline]
```

## 16.248.3 Member Function Documentation

### 16.248.3.1 rns()

```
const rns\_double & rns ( ) const [inline]
```

**16.248.3.2 delayed()**

```
const RNSInteger< RNS > & delayed ( ) const [inline]
```

**16.248.3.3 size()**

```
size_t size ( ) const [inline]
```

**16.248.3.4 isOne()**

```
bool isOne (
    const Element & x ) const [inline]
```

**16.248.3.5 isMOne()**

```
bool isMOne (
    const Element & x ) const [inline]
```

**16.248.3.6 isZero()**

```
bool isZero (
    const Element & x ) const [inline]
```

**16.248.3.7 characteristic() [1/2]**

```
integer & characteristic (
    integer & p ) const [inline]
```

**16.248.3.8 characteristic() [2/2]**

```
integer characteristic ( ) const [inline]
```

**16.248.3.9 cardinality() [1/2]**

```
integer & cardinality (
    integer & p ) const [inline]
```

**16.248.3.10 cardinality() [2/2]**

```
integer cardinality ( ) const [inline]
```

**16.248.3.11 minElement()**

```
integer minElement ( ) const [inline]
```

**16.248.3.12 maxElement()**

```
integer maxElement ( ) const [inline]
```

**16.248.3.13 init() [1/3]**

```
Element & init (  
    Element & x ) const [inline]
```

**16.248.3.14 init() [2/3]**

```
Element & init (  
    Element & x,  
    const Givaro::Integer & y ) const [inline]
```

**16.248.3.15 reduce() [1/2]**

```
Element & reduce (  
    Element & x,  
    const Element & y ) const [inline]
```

**16.248.3.16 reduce() [2/2]**

```
Element & reduce (  
    Element & x ) const [inline]
```

**16.248.3.17 init() [3/3]**

```
Element & init (  
    Element & x,  
    const Element & y ) const [inline]
```

**16.248.3.18 convert()**

```
Givaro::Integer convert (  
    Givaro::Integer & x,  
    const Element & y ) const [inline]
```

**16.248.3.19 assign()**

```
Element & assign (  
    Element & x,  
    const Element & y ) const [inline]
```

**16.248.3.20 add()**

```
Element & add (  
    Element & x,  
    const Element & y,  
    const Element & z ) const [inline]
```

**16.248.3.21 sub()**

```
Element & sub (  
    Element & x,
```



```
const Element & y,
const Element & z ) const [inline]
```

### 16.248.3.22 neg()

```
Element & neg (
    Element & x,
    const Element & y ) const [inline]
```

### 16.248.3.23 mul()

```
Element & mul (
    Element & x,
    const Element & y,
    const Element & z ) const [inline]
```

### 16.248.3.24 axpyin()

```
Element & axpyin (
    Element & x,
    const Element & y,
    const Element & z ) const [inline]
```

### 16.248.3.25 inv()

```
Element & inv (
    Element & x,
    const Element & y ) const [inline]
```

### 16.248.3.26 areEqual()

```
bool areEqual (
    const Element & x,
    const Element & y ) const [inline]
```

### 16.248.3.27 write() [1/2]

```
std::ostream & write (
    std::ostream & os,
    const Element & y ) const [inline]
```

### 16.248.3.28 write() [2/2]

```
std::ostream & write (
    std::ostream & os ) const [inline]
```

### 16.248.3.29 reduce\_modp() [1/2]

```
void reduce_modp (
    size_t n,
    Element_ptr B ) const [inline]
```

**16.248.3.30 write\_matrix()**

```
std::ostream & write_matrix (
    std::ostream & c,
    const double * E,
    int n,
    int m,
    int lda ) const [inline]
```

**16.248.3.31 write\_matrix\_long()**

```
std::ostream & write_matrix_long (
    std::ostream & c,
    const double * E,
    int n,
    int m,
    int lda ) const [inline]
```

**16.248.3.32 reduce\_modp() [2/2]**

```
void reduce_modp (
    size_t m,
    size_t n,
    Element_ptr B,
    size_t lda ) const [inline]
```

**16.248.3.33 reduce\_modp\_rnsmajor()**

```
void reduce_modp_rnsmajor (
    size_t n,
    Element_ptr B ) const [inline]
```

**16.248.4 Field Documentation****16.248.4.1 \_p**

```
integer _p [protected]
```

**16.248.4.2 \_Mi\_modp\_rns**

```
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _Mi_modp_↵
rns [protected]
```

**16.248.4.3 \_iM\_modp\_rns**

```
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _iM_modp_↵
rns [protected]
```

**16.248.4.4 \_rns**

```
const RNS* _rns [protected]
```

16.248.4.5 `_F`

```
Givaro::Modular<Givaro::Integer> _F [protected]
```

16.248.4.6 `_RNSdelayed`

```
RNSInteger<RNS> _RNSdelayed [protected]
```

16.248.4.7 `one`

```
Element one
```

16.248.4.8 `mOne`

```
Element mOne
```

16.248.4.9 `zero`

```
Element zero
```

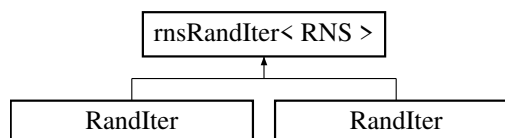
The documentation for this class was generated from the following files:

- [field-traits.h](#)
- [rns-integer-mod.h](#)

16.249 `rnsRandIter< RNS >` Class Template Reference

```
#include <rns-double.h>
```

Inheritance diagram for `rnsRandIter< RNS >`:



## Public Member Functions

- `rnsRandIter` (const `RNS` &`R`, `size_t` `size`=0, `uint64_t` `seed`=0)
- `RNS::Element` & `random` (typename `RNS::Element` &`elt`) const  
*RNS ring Element random assignement.*
- `RNS::Element` & `operator()` (typename `RNS::Element` &`elt`) const
- `RNS::Element` `operator()` () const
- `RNS::Element` `random` () const
- const `RNS` & `ring` () const

## 16.249.1 Constructor &amp; Destructor Documentation

16.249.1.1 `rnsRandIter()`

```

rnsRandIter (
    const RNS & R,
    size_t size = 0,
    uint64_t seed = 0 ) [inline]

```

## 16.249.2 Member Function Documentation

### 16.249.2.1 random() [1/2]

```
RNS::Element & random (
    typename RNS::Element & elt ) const [inline]
```

RNS ring Element random assignment.

Element is supposed to be initialized

Returns

random ring Element

### 16.249.2.2 operator>() [1/2]

```
RNS::Element & operator() (
    typename RNS::Element & elt ) const [inline]
```

### 16.249.2.3 operator>() [2/2]

```
RNS::Element operator() ( ) const [inline]
```

### 16.249.2.4 random() [2/2]

```
RNS::Element random ( ) const [inline]
```

### 16.249.2.5 ring()

```
const RNS & ring ( ) const [inline]
```

The documentation for this class was generated from the following file:

- [rns-double.h](#)

## 16.250 Row Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

## 16.251 ruint< K > Class Template Reference

The documentation for this class was generated from the following file:

- [field-traits.h](#)

## 16.252 ScalFunctions< Element, Enable > Struct Template Reference

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.253 ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type > Struct Template Reference

### Static Public Member Functions

- static Element [zero](#) ()
- static Element [vand](#) (Element x1, Element x2)
- static Element [vor](#) (Element x1, Element x2)
- static Element [vxor](#) (Element x1, Element x2)
- static Element [vandnot](#) (Element x1, Element x2)
- static Element [ceil](#) (Element x)
- static Element [floor](#) (Element x)
- static Element [round](#) (Element x)
- static Element [add](#) (Element x1, Element x2)
- static Element [addin](#) (Element &x1, Element x2)
- static Element [sub](#) (Element x1, Element x2)
- static Element [subin](#) (Element &x1, Element x2)
- static Element [mul](#) (Element x1, Element x2)
- static Element [mulin](#) (Element &x1, Element x2)
- static Element [div](#) (Element x1, Element x2)
- static Element [fmadd](#) (Element x1, Element x2, Element x3)
- static Element [fmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [fmsub](#) (Element x1, Element x2, Element x3)
- static Element [fmsubin](#) (Element &x1, Element x2, Element x3)
- static Element [fnmadd](#) (Element x1, Element x2, Element x3)
- static Element [fnmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [lesser](#) (Element x1, Element x2)
- static Element [lesser\\_eq](#) (Element x1, Element x2)
- static Element [greater](#) (Element x1, Element x2)
- static Element [greater\\_eq](#) (Element x1, Element x2)
- static Element [eq](#) (Element x1, Element x2)

### 16.253.1 Member Function Documentation

#### 16.253.1.1 zero()

```
static Element zero ( ) [inline], [static]
```

#### 16.253.1.2 vand()

```
static Element vand (
    Element x1,
    Element x2 ) [inline], [static]
```

#### 16.253.1.3 vor()

```
static Element vor (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.253.1.4 vxor()**

```
static Element vxor (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.5 vandnot()**

```
static Element vandnot (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.6 ceil()**

```
static Element ceil (  
    Element x ) [inline], [static]
```

**16.253.1.7 floor()**

```
static Element floor (  
    Element x ) [inline], [static]
```

**16.253.1.8 round()**

```
static Element round (  
    Element x ) [inline], [static]
```

**16.253.1.9 add()**

```
static Element add (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.10 addin()**

```
static Element addin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.11 sub()**

```
static Element sub (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.12 subin()**

```
static Element subin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.13 mul()**

```
static Element mul (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.14 mulin()**

```
static Element mulin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.15 div()**

```
static Element div (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.253.1.16 fmadd()**

```
static Element fmadd (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.253.1.17 fmaddin()**

```
static Element fmaddin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.253.1.18 fmsub()**

```
static Element fmsub (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.253.1.19 fmsubin()**

```
static Element fmsubin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.253.1.20 fnmadd()**

```
static Element fnmadd (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.253.1.21 fnmaddin()**

```
static Element fnmaddin (
    Element & x1,
    Element x2,
    Element x3 ) [inline], [static]
```

**16.253.1.22 lesser()**

```
static Element lesser (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.253.1.23 lesser\_eq()**

```
static Element lesser_eq (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.253.1.24 greater()**

```
static Element greater (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.253.1.25 greater\_eq()**

```
static Element greater_eq (
    Element x1,
    Element x2 ) [inline], [static]
```

**16.253.1.26 eq()**

```
static Element eq (
    Element x1,
    Element x2 ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.254 ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type > Struct Template Reference

### Static Public Member Functions

- static Element [zero](#) ()
- static Element [round](#) (Element x)
- static Element [vand](#) (Element x1, Element x2)
- static Element [vor](#) (Element x1, Element x2)
- static Element [vxor](#) (Element x1, Element x2)
- static Element [vandnot](#) (Element x1, Element x2)
- static Element [add](#) (Element x1, Element x2)
- static Element [addin](#) (Element &x1, Element x2)



- static Element [sub](#) (Element x1, Element x2)
- static Element [subin](#) (Element &x1, Element x2)
- static Element [mul](#) (Element x1, Element x2)
- static Element [mullo](#) (Element x1, Element x2)
- static Element [mulhi](#) (Element x1, Element x2)
- static Element [mulx](#) (Element x1, Element x2)
- static Element [fmadd](#) (Element x1, Element x2, Element x3)
- static Element [fmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [fmaddx](#) (Element x1, Element x2, Element x3)
- static Element [fmaddxin](#) (Element &x1, Element x2, Element x3)
- static Element [fmsub](#) (Element x1, Element x2, Element x3)
- static Element [fmsubin](#) (Element &x1, Element x2, Element x3)
- static Element [fmsubx](#) (Element x1, Element x2, Element x3)
- static Element [fmsubxin](#) (Element &x1, Element x2, Element x3)
- static Element [fnmadd](#) (Element x1, Element x2, Element x3)
- static Element [fnmaddin](#) (Element &x1, Element x2, Element x3)
- static Element [fnmaddx](#) (Element x1, Element x2, Element x3)
- static Element [fnmaddxin](#) (Element &x1, Element x2, Element x3)
- template<int s, bool EnableTrue = true>  
static enable\_if<is\_signed< Element >::value &&EnableTrue, Element >::type [sra](#) (Element x1)
- template<int s, bool EnableTrue = true>  
static enable\_if< is\_signed< Element >::value &&EnableTrue, Element >::type [sra](#) (Element x1)
- template<int s>  
static Element [srl](#) (Element x1)
- template<int s>  
static Element [sll](#) (Element x1)
- static Element [lesser](#) (Element x1, Element x2)
- static Element [lesser\\_eq](#) (Element x1, Element x2)
- static Element [greater](#) (Element x1, Element x2)
- static Element [greater\\_eq](#) (Element x1, Element x2)
- static Element [eq](#) (Element x1, Element x2)

## 16.254.1 Member Function Documentation

### 16.254.1.1 zero()

```
static Element zero ( ) [inline], [static]
```

### 16.254.1.2 round()

```
static Element round (  
    Element x ) [inline], [static]
```

### 16.254.1.3 vand()

```
static Element vand (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.4 vor()**

```
static Element vor (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.5 vxor()**

```
static Element vxor (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.6 vandnot()**

```
static Element vandnot (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.7 add()**

```
static Element add (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.8 addin()**

```
static Element addin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.9 sub()**

```
static Element sub (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.10 subin()**

```
static Element subin (  
    Element & x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.11 mul()**

```
static Element mul (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.12 mullo()**

```
static Element mullo (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.13 mulhi()**

```
static Element mulhi (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.14 mulx()**

```
static Element mulx (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.15 fmadd()**

```
static Element fmadd (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.16 fmaddin()**

```
static Element fmaddin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.17 fmaddx()**

```
static Element fmaddx (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.18 fmaddxin()**

```
static Element fmaddxin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.19 fmsub()**

```
static Element fmsub (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.20 fmsubin()**

```
static Element fmsubin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.21 fmsubx()**

```
static Element fmsubx (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.22 fmsubxin()**

```
static Element fmsubxin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.23 fnmadd()**

```
static Element fnmadd (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.24 fnmaddin()**

```
static Element fnmaddin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.25 fnmaddx()**

```
static Element fnmaddx (  
    Element x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.26 fnmaddxin()**

```
static Element fnmaddxin (  
    Element & x1,  
    Element x2,  
    Element x3 ) [inline], [static]
```

**16.254.1.27 sra() [1/2]**

```
static enable_if<!is_signed< Element >::value &&EnableTrue, Element >::type sra (  
    Element x1 ) [inline], [static]
```

**16.254.1.28 sra()** [2/2]

```
static enable_if< is_signed< Element >::value &&EnableTrue, Element >::type sra (  
    Element x1 ) [inline], [static]
```

**16.254.1.29 srl()**

```
static Element srl (  
    Element x1 ) [inline], [static]
```

**16.254.1.30 sll()**

```
static Element sll (  
    Element x1 ) [inline], [static]
```

**16.254.1.31 lesser()**

```
static Element lesser (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.32 lesser\_eq()**

```
static Element lesser_eq (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.33 greater()**

```
static Element greater (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.34 greater\_eq()**

```
static Element greater_eq (  
    Element x1,  
    Element x2 ) [inline], [static]
```

**16.254.1.35 eq()**

```
static Element eq (  
    Element x1,  
    Element x2 ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [test-simd.C](#)

## 16.255 Sequential Struct Reference

### Public Member Functions

- [Sequential](#) ()
- [Sequential](#) (size\_t nth)
- template<class Cut, class Param >  
[Sequential](#) ([Parallel](#)< Cut, Param > &)
- size\_t [numthreads](#) () const

### Friends

- std::ostream & [operator<<](#) (std::ostream &out, const [Sequential](#) &)

## 16.255.1 Constructor & Destructor Documentation

### 16.255.1.1 Sequential() [1/3]

```
Sequential ( ) [inline]
```

### 16.255.1.2 Sequential() [2/3]

```
Sequential (
    size_t nth ) [inline]
```

### 16.255.1.3 Sequential() [3/3]

```
Sequential (
    Parallel< Cut, Param > & ) [inline]
```

## 16.255.2 Member Function Documentation

### 16.255.2.1 numthreads()

```
size_t numthreads ( ) const [inline]
```

## 16.255.3 Friends And Related Function Documentation

### 16.255.3.1 operator<<

```
std::ostream & operator<< (
    std::ostream & out,
    const Sequential & ) [friend]
```

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

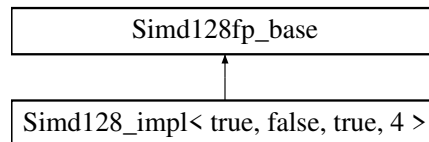
## 16.256 Simd128\_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- [simd128.inl](#)

## 16.257 Simd128\_impl< true, false, true, 4 > Struct Reference

Inheritance diagram for Simd128\_impl< true, false, true, 4 >:



### Static Public Member Functions

- static const std::string [type\\_string](#) ()

### 16.257.1 Member Function Documentation

#### 16.257.1.1 type\_string()

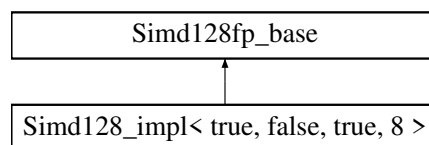
```
static const std::string type_string ( ) [inline], [static], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_float.inl](#)

## 16.258 Simd128\_impl< true, false, true, 8 > Struct Reference

Inheritance diagram for Simd128\_impl< true, false, true, 8 >:



### Static Public Member Functions

- static const std::string [type\\_string](#) ()

### 16.258.1 Member Function Documentation

#### 16.258.1.1 type\_string()

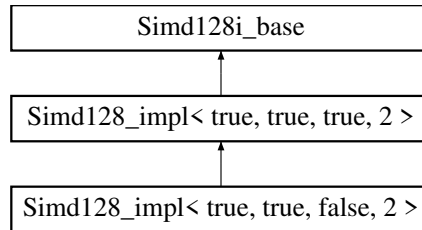
```
static const std::string type_string ( ) [inline], [static], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_double.inl](#)

## 16.259 Simd128\_impl< true, true, false, 2 > Struct Reference

Inheritance diagram for Simd128\_impl< true, true, false, 2 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [scalar\\_t](#) = [uint16\\_t](#)
- using [vect\\_t](#) = [\\_\\_m128i](#)

### Static Public Member Functions

- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmsubxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST scalar\\_t hadd\\_to\\_scal](#) (const [vect\\_t](#) a)
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)



- `template<class T >`  
`static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)`
- `static INLINE PURE vect_t load (const scalar_t *const p)`
- `static INLINE PURE vect_t loadu (const scalar_t *const p)`
- `static INLINE void store (scalar_t *p, vect_t v)`
- `static INLINE void storeu (scalar_t *p, vect_t v)`
- `static INLINE void stream (scalar_t *p, const vect_t v)`
- `template<int s>`  
`static INLINE CONST vect_t sll (const vect_t a)`
- `template<int s>`  
`static INLINE CONST vect_t srl (const vect_t a)`
- `template<uint32_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmadddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t round (const vect_t a)`
- `static INLINE vect_t mod (vect_t &C, const vect_t &P, const __m64 &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`
- `static const std::string type_string ()`
- `static INLINE CONST vect_t zero ()`
- `template<uint8_t s>`  
`static INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t srl128 (const vect_t a)`
- `static INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- `static const constexpr size_t vect_size = 8`
- `static const constexpr size_t alignment = 16`

## 16.259.1 Member Typedef Documentation

### 16.259.1.1 scalar\_t

using `scalar_t` = `uint16_t`

### 16.259.1.2 vect\_t

```
using vect_t = __m128i [inherited]
```

## 16.259.2 Member Function Documentation

### 16.259.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.259.2.2 set() [1/2]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

### 16.259.2.3 gather() [1/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.259.2.4 load() [1/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

### 16.259.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

### 16.259.2.6 store() [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

### 16.259.2.7 storeu() [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.259.2.8 stream()** [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.259.2.9 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.259.2.10 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.259.2.11 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.259.2.12 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.259.2.13 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.259.2.14 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.259.2.15 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.259.2.16 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.259.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.259.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.259.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.259.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.259.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.259.2.22 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.259.2.23 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.259.2.24 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.259.2.25 set1() [2/2]**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```

**16.259.2.26 set() [2/2]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static], [inherited]
```

**16.259.2.27 gather() [2/2]**

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

**16.259.2.28 load() [2/2]**

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.259.2.29 loadu() [2/2]**

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.259.2.30 store() [2/2]**

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

**16.259.2.31 storeu() [2/2]**

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

**16.259.2.32 stream()** [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static], [inherited]
```

**16.259.2.33 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.259.2.34 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.259.2.35 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.259.2.36 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.37 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.38 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.39 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.40 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.41 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.42 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.43 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.44 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.45 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.46 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.47 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.48 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.49 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.50 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.51 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.52 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.259.2.53 mod()**

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const __m64 & INV_P,  
    const vect_t & NEG_P,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static], [inherited]
```

**16.259.2.54 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.259.2.55 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.259.2.56 sll128()**

```
static INLINE CONST vect_t sll128 (  
    const vect_t a ) [inline], [static], [inherited]
```



**16.259.2.57 srl128()**

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.259.2.58 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.59 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.60 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.2.61 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.259.3 Field Documentation****16.259.3.1 vect\_size**

```
const constexpr size_t vect_size = 8 [static], [constexpr], [inherited]
```

**16.259.3.2 alignment**

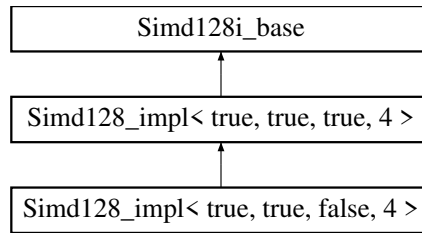
```
const constexpr size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int16.inl](#)

**16.260 Simd128\_impl< true, true, false, 4 > Struct Reference**

Inheritance diagram for Simd128\_impl< true, true, false, 4 >:



## Data Structures

- union [Converter](#)

## Public Types

- using [scalar\\_t](#) = [uint32\\_t](#)
- using [vect\\_t](#) = [\\_\\_m128i](#)

## Static Public Member Functions

- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmsubxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST scalar\\_t hadd\\_to\\_scal](#) (const [vect\\_t](#) a)
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)

- static `INLINE` void `stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>  
static `INLINE` `CONST` `vect_t` `sll` (const `vect_t` a)
- template<int s>  
static `INLINE` `CONST` `vect_t` `srl` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE` `CONST` `vect_t` `shuffle` (const `vect_t` a)
- static `INLINE` `CONST` `vect_t` `unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `unpackhi` (const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>  
static `INLINE` `CONST` `vect_t` `blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `add` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `round` (const `vect_t` a)
- static `INLINE` `vect_t` `mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE` `CONST` `vect_t` `zero` ()
- template<uint8\_t s>  
static `INLINE` `CONST` `vect_t` `sll128` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE` `CONST` `vect_t` `srl128` (const `vect_t` a)
- static `INLINE` `CONST` `vect_t` `vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 4
- static const constexpr size\_t `alignment` = 16

## 16.260.1 Member Typedef Documentation

### 16.260.1.1 scalar\_t

```
using scalar_t = uint32_t
```

### 16.260.1.2 vect\_t

```
using vect_t = __m128i [inherited]
```

## 16.260.2 Member Function Documentation

### 16.260.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.260.2.2 set() [1/2]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static]
```

### 16.260.2.3 gather() [1/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.260.2.4 load() [1/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

### 16.260.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

### 16.260.2.6 store() [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

### 16.260.2.7 storeu() [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

### 16.260.2.8 stream() [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.260.2.9 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.260.2.10 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.260.2.11 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.260.2.12 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.13 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.14 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.15 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.16 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.260.2.17 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.260.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.260.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.260.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.260.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.260.2.22 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.260.2.23 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

#### 16.260.2.24 compliant()

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

#### 16.260.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static], [inherited]
```

**16.260.2.26 set()** [2/2]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static], [inherited]
```

**16.260.2.27 gather()** [2/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static], [inherited]
```

**16.260.2.28 load()** [2/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.260.2.29 loadu()** [2/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.260.2.30 store()** [2/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static], [inherited]
```

**16.260.2.31 storeu()** [2/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static], [inherited]
```

**16.260.2.32 stream()** [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static], [inherited]
```

**16.260.2.33 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.260.2.34 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.260.2.35 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.260.2.36 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.37 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.38 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.39 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.40 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.41 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.42 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.43 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```



**16.260.2.44 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.45 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.46 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.47 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.48 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.49 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.50 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.51 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.52 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.260.2.53 mod()**

```
static INLINE vect_t mod (  
    vect_t & C,  
    const vect_t & P,  
    const vect_t & INVP,  
    const vect_t & NEGP,  
    const vect_t & MIN,  
    const vect_t & MAX,  
    vect_t & Q,  
    vect_t & T ) [inline], [static], [inherited]
```

**16.260.2.54 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.260.2.55 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.260.2.56 sll128()**

```
static INLINE CONST vect_t sll128 (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.260.2.57 srl128()**

```
static INLINE CONST vect_t srl128 (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.260.2.58 vand()**

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.59 vor()**

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.60 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.2.61 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.260.3 Field Documentation****16.260.3.1 vect\_size**

```
const constexpr size_t vect_size = 4 [static], [constexpr], [inherited]
```

**16.260.3.2 alignment**

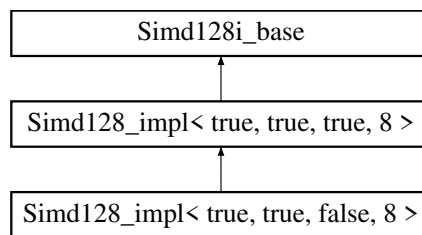
```
const constexpr size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int32.inl](#)

**16.261 Simd128\_impl< true, true, false, 8 > Struct Reference**

Inheritance diagram for Simd128\_impl< true, true, false, 8 >:

**Data Structures**

- union [Converter](#)

**Public Types**

- using [scalar\\_t](#) = [uint64\\_t](#)
- using [vect\\_t](#) = [\\_\\_m128i](#)

**Static Public Member Functions**

- static **INLINE** **CONST** [vect\\_t](#) set1 (const [scalar\\_t](#) x)
- static **INLINE** **CONST** [vect\\_t](#) set (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1)

- `template<class T >`  
`static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)`
- `static INLINE PURE vect_t load (const scalar_t *const p)`
- `static INLINE PURE vect_t loadu (const scalar_t *const p)`
- `static INLINE void store (scalar_t *p, vect_t v)`
- `static INLINE void storeu (scalar_t *p, vect_t v)`
- `static INLINE void stream (scalar_t *p, const vect_t v)`
- `template<int s>`  
`static INLINE CONST vect_t sra (const vect_t a)`
- `static INLINE CONST vect_t greater (vect_t a, vect_t b)`
- `static INLINE CONST vect_t lesser (vect_t a, vect_t b)`
- `static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mullo (const vect_t x0, const vect_t x1)`
- `static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubxin (vect_t c, const vect_t a, const vect_t b)`
- `static INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- `template<class T >`  
`static constexpr bool valid (T *p)`
- `template<class T >`  
`static constexpr bool compliant (T n)`
- `static INLINE CONST vect_t set1 (const scalar_t x)`
- `static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1)`
- `template<class T >`  
`static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)`
- `template<int idx>`  
`static INLINE CONST scalar_t get (vect_t v)`
- `static INLINE PURE vect_t load (const scalar_t *const p)`
- `static INLINE PURE vect_t loadu (const scalar_t *const p)`
- `static INLINE void store (scalar_t *p, vect_t v)`
- `static INLINE void storeu (scalar_t *p, vect_t v)`
- `static INLINE void stream (scalar_t *p, const vect_t v)`
- `template<int s>`  
`static INLINE CONST vect_t sll (const vect_t a)`
- `template<int s>`  
`static INLINE CONST vect_t srl (const vect_t a)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`

- static `INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t round (const vect_t a)`
- static `INLINE CONST vect_t mask_high ()`
- static `INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)`
- static `INLINE vect_t mod (vect_t &C, const __m128d &P, const __m128d &INVP, const __m128d &NEGP, const vect_t &POW50REM, const __m128d &MIN, const __m128d &MAX, __m128d &Q, __m128d &T)`
- static const std::string `type_string ()`
- static `INLINE CONST vect_t zero ()`
- `template<uint8_t s>`  
static `INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`  
static `INLINE CONST vect_t srl128 (const vect_t a)`
- static `INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 2
- static const constexpr size\_t `alignment` = 16

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits (const vect_t x)`

## 16.261.1 Member Typedef Documentation

### 16.261.1.1 scalar\_t

```
using scalar_t = uint64_t
```

### 16.261.1.2 vect\_t

```
using vect_t = __m128i [inherited]
```

## 16.261.2 Member Function Documentation

### 16.261.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.261.2.2 set() [1/2]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1 ) [inline], [static]
```

**16.261.2.3 gather()** [1/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

**16.261.2.4 load()** [1/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.261.2.5 loadu()** [1/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.261.2.6 store()** [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.261.2.7 storeu()** [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.261.2.8 stream()** [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.261.2.9 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.261.2.10 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.261.2.11 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.261.2.12 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.261.2.13 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.261.2.14 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t x0,  
    const vect_t x1 ) [inline], [static]
```

**16.261.2.15 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.261.2.16 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.261.2.17 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.261.2.18 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.261.2.19 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.261.2.20 fmsubx()**

```
static INLINE CONST vect_t fmsubx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.261.2.21 fmsubxin() [1/2]**

```
static INLINE CONST vect_t fmsubxin (
    vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.261.2.22 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

**16.261.2.23 valid()**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.261.2.24 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.261.2.25 set1() [2/2]**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```

**16.261.2.26 set() [2/2]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1 ) [inline], [static], [inherited]
```

**16.261.2.27 gather() [2/2]**

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

**16.261.2.28 get()**

```
static INLINE CONST scalar_t get (
    vect_t v ) [inline], [static], [inherited]
```



**16.261.2.29 load()** [2/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.261.2.30 loadu()** [2/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.261.2.31 store()** [2/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static], [inherited]
```

**16.261.2.32 storeu()** [2/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static], [inherited]
```

**16.261.2.33 stream()** [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static], [inherited]
```

**16.261.2.34 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.261.2.35 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.261.2.36 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.261.2.37 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.38 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.39 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.40 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.41 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.42 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.43 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.44 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.45 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.46 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.47 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.48 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.49 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.50 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.51 fmsubxin() [2/2]**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.52 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.53 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.261.2.54 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

**16.261.2.55 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static], [inherited]
```

**16.261.2.56 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m128d & P,
    const __m128d & INVP,
    const __m128d & NEGP,
    const vect_t & POW5OREM,
    const __m128d & MIN,
    const __m128d & MAX,
    __m128d & Q,
    __m128d & T ) [static], [inherited]
```

**16.261.2.57 signbits()**

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected], [inherited]
```

**16.261.2.58 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.261.2.59 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.261.2.60 sll128()**

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.261.2.61 srl128()**

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.261.2.62 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.63 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.64 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.2.65 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.261.3 Field Documentation****16.261.3.1 vect\_size**

```
const constexpr size_t vect_size = 2 [static], [constexpr], [inherited]
```

**16.261.3.2 alignment**

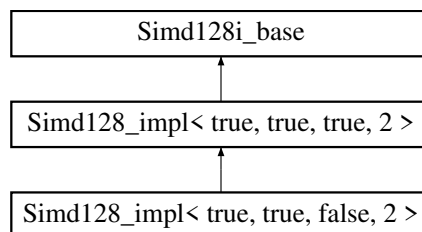
```
const constexpr size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int64.inl](#)

**16.262 Simd128\_impl< true, true, true, 2 > Struct Reference**

Inheritance diagram for Simd128\_impl< true, true, true, 2 >:

**Data Structures**

- union [Converter](#)

**Public Types**

- using [vect\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int16\_t

## Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- template<uint32\_t s>  
static [INLINE CONST vect\\_t shuffle](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t blend](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t add](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t addin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t sub](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t subin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mul](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmadd](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmadd](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsub](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmsubin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmsubxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST scalar\\_t hadd\\_to\\_scal](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t round](#) (const [vect\\_t](#) a)

- static `INLINE vect_t mod (vect_t &C, const vect_t &P, const __m64 &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`
- static const std::string `type_string ()`
- static `INLINE CONST vect_t zero ()`
- `template<uint8_t s>`  
static `INLINE CONST vect_t sll128 (const vect_t a)`
- `template<uint8_t s>`  
static `INLINE CONST vect_t srl128 (const vect_t a)`
- static `INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- static `INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 8
- static const constexpr size\_t `alignment` = 16

## 16.262.1 Member Typedef Documentation

### 16.262.1.1 vect\_t

```
using vect_t = __m128i
```

### 16.262.1.2 scalar\_t

```
using scalar_t = int16_t
```

## 16.262.2 Member Function Documentation

### 16.262.2.1 valid()

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

### 16.262.2.2 compliant()

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

### 16.262.2.3 set1()

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

#### 16.262.2.4 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

#### 16.262.2.5 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.262.2.6 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.262.2.7 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.262.2.8 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.262.2.9 storeu()

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.262.2.10 stream()

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.262.2.11 sll()

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```



**16.262.2.12 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.262.2.13 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.262.2.14 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.262.2.15 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.16 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.17 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.18 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.19 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.20 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.21 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.22 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.23 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.24 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.25 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.26 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.27 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.28 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.29 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.30 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.31 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.32 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.33 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.34 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.35 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.262.2.36 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.262.2.37 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.262.2.38 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.262.2.39 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.262.2.40 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.262.2.41 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.262.2.42 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.262.2.43 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.262.2.44 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.262.2.45 mod()**

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const __m64 & INV_P,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

**16.262.2.46 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.262.2.47 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.262.2.48 sll128()**

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.262.2.49 srl128()**

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.262.2.50 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.262.2.51 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.262.2.52 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

### 16.262.2.53 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

## 16.262.3 Field Documentation

### 16.262.3.1 vect\_size

```
const constexpr size_t vect_size = 8 [static], [constexpr]
```

### 16.262.3.2 alignment

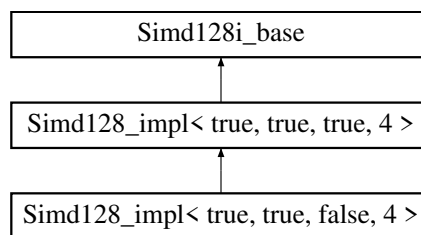
```
const constexpr size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int16.inl](#)

## 16.263 Simd128\_impl< true, true, true, 4 > Struct Reference

Inheritance diagram for Simd128\_impl< true, true, true, 4 >:



## Data Structures

- union [Converter](#)

## Public Types

- using [vect\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int32\_t

## Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static **INLINE** **CONST** [vect\\_t](#) [set1](#) (const [scalar\\_t](#) x)
- static **INLINE** **CONST** [vect\\_t](#) [set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class T >  
static **INLINE** **PURE** [vect\\_t](#) [gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static **INLINE** **PURE** [vect\\_t](#) [load](#) (const [scalar\\_t](#) \*const p)
- static **INLINE** **PURE** [vect\\_t](#) [loadu](#) (const [scalar\\_t](#) \*const p)
- static **INLINE** void [store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static **INLINE** void [storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)

- static `INLINE` void `stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>
  - static `INLINE` `CONST` `vect_t` `sll` (const `vect_t` a)
- template<int s>
  - static `INLINE` `CONST` `vect_t` `srl` (const `vect_t` a)
- template<int s>
  - static `INLINE` `CONST` `vect_t` `sra` (const `vect_t` a)
- template<uint8\_t s>
  - static `INLINE` `CONST` `vect_t` `shuffle` (const `vect_t` a)
- static `INLINE` `CONST` `vect_t` `unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `unpackhi` (const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>
  - static `INLINE` `CONST` `vect_t` `blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `add` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `vect_t` `fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `scalar_t` `hadd_to_scal` (const `vect_t` a)
- static `INLINE` `CONST` `vect_t` `round` (const `vect_t` a)
- static `INLINE` `vect_t` `mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE` `CONST` `vect_t` `zero` ()
- template<uint8\_t s>
  - static `INLINE` `CONST` `vect_t` `sll128` (const `vect_t` a)
- template<uint8\_t s>
  - static `INLINE` `CONST` `vect_t` `srl128` (const `vect_t` a)
- static `INLINE` `CONST` `vect_t` `vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE` `CONST` `vect_t` `vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 4
- static const constexpr size\_t `alignment` = 16

## 16.263.1 Member Typedef Documentation

### 16.263.1.1 `vect_t`

```
using vect_t = __m128i
```

### 16.263.1.2 `scalar_t`

```
using scalar_t = int32_t
```

## 16.263.2 Member Function Documentation

### 16.263.2.1 `valid()`

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr]
```

### 16.263.2.2 `compliant()`

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr]
```

### 16.263.2.3 `set1()`

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.263.2.4 `set()`

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static]
```

### 16.263.2.5 `gather()`

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```



**16.263.2.6 load()**

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.263.2.7 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.263.2.8 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.263.2.9 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.263.2.10 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.263.2.11 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.263.2.12 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.263.2.13 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.263.2.14 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.263.2.15 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.16 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.17 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.18 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.19 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.20 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.21 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.22 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.23 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.24 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.25 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.26 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.27 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.28 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.29 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.30 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.31 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.32 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.33 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.34 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.35 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.36 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.37 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.263.2.38 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,
```

```
const vect_t b ) [inline], [static]
```

#### 16.263.2.39 greater()

```
static INLINE CONST vect_t greater (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.263.2.40 lesser()

```
static INLINE CONST vect_t lesser (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.263.2.41 greater\_eq()

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.263.2.42 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.263.2.43 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

#### 16.263.2.44 round()

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

#### 16.263.2.45 mod()

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

#### 16.263.2.46 type\_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.263.2.47 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.263.2.48 sll128()**

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.263.2.49 srl128()**

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static], [inherited]
```

**16.263.2.50 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.263.2.51 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.263.2.52 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.263.2.53 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.263.3 Field Documentation****16.263.3.1 vect\_size**

```
const constexpr size_t vect_size = 4 [static], [constexpr]
```

**16.263.3.2 alignment**

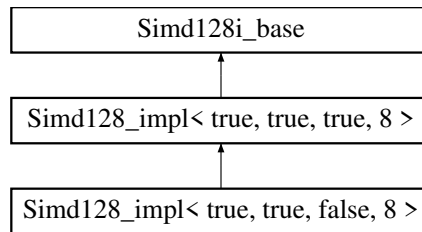
```
const constexpr size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int32.inl](#)

## 16.264 Simd128\_impl< true, true, true, 8 > Struct Reference

Inheritance diagram for Simd128\_impl< true, true, true, 8 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [vect\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int64\_t

### Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- template<int idx>  
static [INLINE CONST scalar\\_t get](#) ([vect\\_t](#) v)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t shuffle](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t blend](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t add](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t addin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t sub](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t subin](#) ([vect\\_t](#) &a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) (const [vect\\_t](#) x0, const [vect\\_t](#) x1)

- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const `__m128d` &P, const `__m128d` &INVP, const `__m128d` &NEGP, const `vect_t` &POW50REM, const `__m128d` &MIN, const `__m128d` &MAX, `__m128d` &Q, `__m128d` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- template<uint8\_t s>  
static `INLINE CONST vect_t sll128` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t srl128` (const `vect_t` a)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 2
- static const constexpr size\_t `alignment` = 16

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

## 16.264.1 Member Typedef Documentation

### 16.264.1.1 vect\_t

```
using vect_t = __m128i
```

### 16.264.1.2 scalar\_t

```
using scalar_t = int64_t
```



## 16.264.2 Member Function Documentation

### 16.264.2.1 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr]
```

### 16.264.2.2 compliant()

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr]
```

### 16.264.2.3 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.264.2.4 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1 ) [inline], [static]
```

### 16.264.2.5 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.264.2.6 get()

```
static INLINE CONST scalar_t get (  
    vect_t v ) [inline], [static]
```

### 16.264.2.7 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

### 16.264.2.8 loadu()

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

### 16.264.2.9 store()

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.264.2.10 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.264.2.11 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.264.2.12 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.264.2.13 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.264.2.14 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.264.2.15 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.264.2.16 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.17 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.18 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.19 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.20 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.21 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.22 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.23 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t x0,  
    const vect_t x1 ) [inline], [static]
```

**16.264.2.24 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.25 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.26 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.27 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.28 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.29 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.30 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.31 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.32 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.33 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.34 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.264.2.35 fmsubin()

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.264.2.36 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.264.2.37 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.264.2.38 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.264.2.39 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.264.2.40 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.264.2.41 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.264.2.42 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.264.2.43 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

**16.264.2.44 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

**16.264.2.45 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

**16.264.2.46 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]
```

**16.264.2.47 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m128d & P,
    const __m128d & INVP,
    const __m128d & NEGP,
    const vect_t & POW5OREM,
    const __m128d & MIN,
    const __m128d & MAX,
    __m128d & Q,
    __m128d & T ) [static]
```

**16.264.2.48 signbits()**

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]
```

**16.264.2.49 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.264.2.50 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

#### 16.264.2.51 sll128()

```
static INLINE CONST vect_t sll128 (  
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.264.2.52 srl128()

```
static INLINE CONST vect_t srl128 (  
    const vect_t a ) [inline], [static], [inherited]
```

#### 16.264.2.53 vand()

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.264.2.54 vor()

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.264.2.55 vxor()

```
static INLINE CONST vect_t vxor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.264.2.56 vandnot()

```
static INLINE CONST vect_t vandnot (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

### 16.264.3 Field Documentation

#### 16.264.3.1 vect\_size

```
const constexpr size_t vect_size = 2 [static], [constexpr]
```

#### 16.264.3.2 alignment

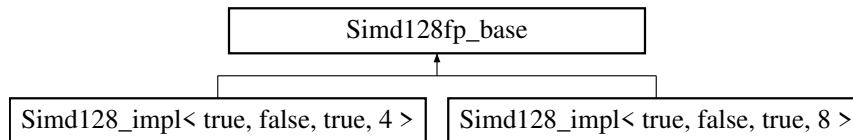
```
const constexpr size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd128\\_int64.inl](#)

## 16.265 Simd128fp\_base Struct Reference

Inheritance diagram for Simd128fp\_base:



## Static Public Member Functions

- static const std::string [type\\_string](#) ()

### 16.265.1 Member Function Documentation

#### 16.265.1.1 [type\\_string\(\)](#)

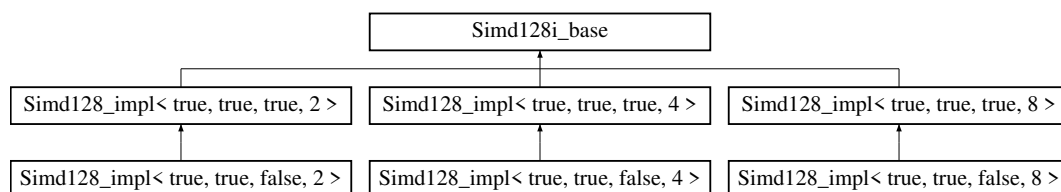
static const std::string [type\\_string](#) ( ) [inline], [static]

The documentation for this struct was generated from the following file:

- [simd128.inl](#)

## 16.266 Simd128i\_base Struct Reference

Inheritance diagram for Simd128i\_base:



## Public Types

- using [vect\\_t](#) = \_\_m128i

## Static Public Member Functions

- static const std::string [type\\_string](#) ()
- static [INLINE CONST vect\\_t zero](#) ()
- template<uint8\_t s>  
static [INLINE CONST vect\\_t sll128](#) (const [vect\\_t](#) a)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t srl128](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t vand](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t vor](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t vxor](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t vandnot](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

### 16.266.1 Member Typedef Documentation

#### 16.266.1.1 [vect\\_t](#)

using [vect\\_t](#) = \_\_m128i



## 16.266.2 Member Function Documentation

### 16.266.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.266.2.2 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

### 16.266.2.3 sll128()

```
static INLINE CONST vect_t sll128 (
    const vect_t a ) [inline], [static]
```

### 16.266.2.4 srl128()

```
static INLINE CONST vect_t srl128 (
    const vect_t a ) [inline], [static]
```

### 16.266.2.5 vand()

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

### 16.266.2.6 vor()

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

### 16.266.2.7 vxor()

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

### 16.266.2.8 vandnot()

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd128.inl](#)

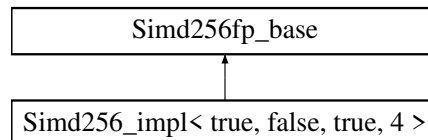
## 16.267 Simd256\_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- [simd256.inl](#)

## 16.268 Simd256\_impl< true, false, true, 4 > Struct Reference

Inheritance diagram for Simd256\_impl< true, false, true, 4 >:

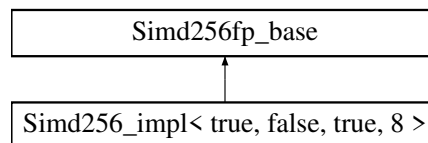


The documentation for this struct was generated from the following file:

- [simd256\\_float.inl](#)

## 16.269 Simd256\_impl< true, false, true, 8 > Struct Reference

Inheritance diagram for Simd256\_impl< true, false, true, 8 >:



### Public Types

- using [vect\\_t](#) = \_\_m256d
- using [scalar\\_t](#) = double

### Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t zero](#) ()
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE](#) void [store](#) (const [scalar\\_t](#) \*p, const [vect\\_t](#) v)
- static [INLINE](#) void [storeu](#) (const [scalar\\_t](#) \*p, const [vect\\_t](#) v)
- static [INLINE](#) void [stream](#) (const [scalar\\_t](#) \*p, const [vect\\_t](#) v)
- static [INLINE CONST vect\\_t unpacklo\\_twice](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi\\_twice](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t blendv (const vect_t a, const vect_t b, const vect_t mask)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t div (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t floor (const vect_t a)`
- `static INLINE CONST vect_t ceil (const vect_t a)`
- `static INLINE CONST vect_t round (const vect_t a)`
- `static INLINE CONST vect_t hadd (const vect_t a, const vect_t b)`
- `static INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- `static INLINE vect_t mod (vect_t &C, const vect_t &P, const vect_t &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)`

## Static Public Attributes

- `static const constexpr size_t vect_size = 4`
- `static const constexpr size_t alignment = 32`

## 16.269.1 Member Typedef Documentation

### 16.269.1.1 vect\_t

`using vect_t = __m256d`

### 16.269.1.2 scalar\_t

`using scalar_t = double`

## 16.269.2 Member Function Documentation

**16.269.2.1 valid()**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

**16.269.2.2 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

**16.269.2.3 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

**16.269.2.4 set1()**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.269.2.5 set()**

```
static INLINE CONST vect_t set (
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4 ) [inline], [static]
```

**16.269.2.6 gather()**

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

**16.269.2.7 load()**

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static]
```

**16.269.2.8 loadu()**

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static]
```

**16.269.2.9 store()**

```
static INLINE void store (
    const scalar_t * p,
    const vect_t v ) [inline], [static]
```

**16.269.2.10 storeu()**

```
static INLINE void storeu (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.269.2.11 stream()**

```
static INLINE void stream (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.269.2.12 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.13 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.14 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.15 blendv()**

```
static INLINE CONST vect_t blendv (  
    const vect_t a,  
    const vect_t b,  
    const vect_t mask ) [inline], [static]
```

**16.269.2.16 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.17 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.18 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.19 subin()**

```
static INLINE CONST vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.20 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.21 mulin()**

```
static INLINE CONST vect_t mulin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.22 div()**

```
static INLINE CONST vect_t div (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.23 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.24 fmaddin()**

```
static INLINE CONST vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.25 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.26 fnmaddin()**

```
static INLINE CONST vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.27 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.28 fmsubin()**

```
static INLINE CONST vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.29 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.30 lesser()**

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.31 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.32 greater()**

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.33 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.34 vand()**

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.35 vor()**

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.36 vxor()**

```
static INLINE CONST vect_t vxor (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.37 vandnot()**

```
static INLINE CONST vect_t vandnot (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.38 floor()**

```
static INLINE CONST vect_t floor (  
    const vect_t a ) [inline], [static]
```

**16.269.2.39 ceil()**

```
static INLINE CONST vect_t ceil (  
    const vect_t a ) [inline], [static]
```

**16.269.2.40 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.269.2.41 hadd()**

```
static INLINE CONST vect_t hadd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.269.2.42 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```



### 16.269.2.43 mod()

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INV_P,
    const vect_t & NEG_P,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

## 16.269.3 Field Documentation

### 16.269.3.1 vect\_size

```
const constexpr size_t vect_size = 4 [static], [constexpr]
```

### 16.269.3.2 alignment

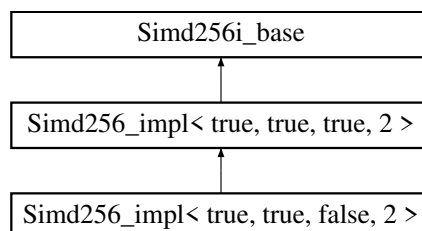
```
const constexpr size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd256\\_double.inl](#)

## 16.270 Simd256\_impl< true, true, false, 2 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, false, 2 >:



## Data Structures

- union [Converter](#)

## Public Types

- using [scalar\\_t](#) = [uint16\\_t](#)
- using [simdHalf](#) = [Simd128](#)< [scalar\\_t](#) >
- using [vect\\_t](#) = [\\_\\_m256i](#)
- using [half\\_t](#) = [\\_\\_m128i](#)

## Static Public Member Functions

- static **INLINE** **CONST** [vect\\_t](#) [set1](#) (const [scalar\\_t](#) x)
- static **INLINE** **CONST** [vect\\_t](#) [set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7, const [scalar\\_t](#) x8, const [scalar\\_t](#) x9, const [scalar\\_t](#) x10, const [scalar\\_t](#) x11, const [scalar\\_t](#) x12, const [scalar\\_t](#) x13, const [scalar\\_t](#) x14, const [scalar\\_t](#) x15)

- `template<class T >`  
`static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)`
- `static INLINE PURE vect_t load (const scalar_t *const p)`
- `static INLINE PURE vect_t loadu (const scalar_t *const p)`
- `static INLINE void store (scalar_t *p, vect_t v)`
- `static INLINE void storeu (scalar_t *p, vect_t v)`
- `static INLINE void stream (scalar_t *p, const vect_t v)`
- `template<int s>`  
`static INLINE CONST vect_t sra (const vect_t a)`
- `static INLINE CONST vect_t greater (vect_t a, vect_t b)`
- `static INLINE CONST vect_t lesser (vect_t a, vect_t b)`
- `static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulhi (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulx (vect_t a, vect_t b)`
- `static INLINE CONST vect_t fmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- `template<class T >`  
`static constexpr bool valid (T *p)`
- `template<class T >`  
`static constexpr bool compliant (T n)`
- `static INLINE CONST vect_t set1 (const scalar_t x)`
- `static INLINE CONST vect_t set (const scalar_t x0, const scalar_t x1, const scalar_t x2, const scalar_t x3, const scalar_t x4, const scalar_t x5, const scalar_t x6, const scalar_t x7, const scalar_t x8, const scalar_t x9, const scalar_t x10, const scalar_t x11, const scalar_t x12, const scalar_t x13, const scalar_t x14, const scalar_t x15)`
- `template<class T >`  
`static INLINE PURE vect_t gather (const scalar_t *const p, const T *const idx)`
- `static INLINE PURE vect_t load (const scalar_t *const p)`
- `static INLINE PURE vect_t loadu (const scalar_t *const p)`
- `static INLINE void store (scalar_t *p, vect_t v)`
- `static INLINE void storeu (scalar_t *p, vect_t v)`
- `static INLINE void stream (scalar_t *p, const vect_t v)`
- `template<int s>`  
`static INLINE CONST vect_t sll (const vect_t a)`
- `template<int s>`  
`static INLINE CONST vect_t srl (const vect_t a)`
- `template<uint64_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo_twice (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi_twice (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `static INLINE void unpacklohi (vect_t &s1, vect_t &s2, const vect_t a, const vect_t b)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t blend_twice (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`

- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 16
- static const constexpr size\_t `alignment` = 32

## 16.270.1 Member Typedef Documentation

### 16.270.1.1 scalar\_t

```
using scalar_t = uint16_t
```

### 16.270.1.2 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

### 16.270.1.3 vect\_t

```
using vect_t = __m256i [inherited]
```

### 16.270.1.4 half\_t

```
using half_t = __m128i [inherited]
```

## 16.270.2 Member Function Documentation

### 16.270.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.270.2.2 set()** [1/2]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7,  
    const scalar_t x8,  
    const scalar_t x9,  
    const scalar_t x10,  
    const scalar_t x11,  
    const scalar_t x12,  
    const scalar_t x13,  
    const scalar_t x14,  
    const scalar_t x15 ) [inline], [static]
```

**16.270.2.3 gather()** [1/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

**16.270.2.4 load()** [1/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.270.2.5 loadu()** [1/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.270.2.6 store()** [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.270.2.7 storeu()** [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.270.2.8 stream()** [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.270.2.9 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.270.2.10 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.270.2.11 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.270.2.12 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.13 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.14 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.15 mulx()**

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.270.2.16 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.270.2.17 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.270.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.270.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.270.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.270.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.270.2.22 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.270.2.23 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr], [inherited]
```

#### 16.270.2.24 compliant()

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr], [inherited]
```

#### 16.270.2.25 set1() [2/2]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static], [inherited]
```

**16.270.2.26 set()** [2/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7,
    const scalar_t x8,
    const scalar_t x9,
    const scalar_t x10,
    const scalar_t x11,
    const scalar_t x12,
    const scalar_t x13,
    const scalar_t x14,
    const scalar_t x15 ) [inline], [static], [inherited]
```

**16.270.2.27 gather()** [2/2]

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

**16.270.2.28 load()** [2/2]

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.270.2.29 loadu()** [2/2]

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.270.2.30 store()** [2/2]

```
static INLINE void store (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

**16.270.2.31 storeu()** [2/2]

```
static INLINE void storeu (
    scalar_t * p,
    vect_t v ) [inline], [static], [inherited]
```

**16.270.2.32 stream()** [2/2]

```
static INLINE void stream (
    scalar_t * p,
    const vect_t v ) [inline], [static], [inherited]
```

**16.270.2.33 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.270.2.34 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.270.2.35 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.270.2.36 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.37 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.38 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.39 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.40 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & s1,  
    vect_t & s2,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.41 blend\_twice()**

```
static INLINE CONST vect_t blend_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```



**16.270.2.42 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.43 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.44 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.45 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.46 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.47 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.48 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.49 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.50 fnmadd()**

```
static INLINE CONST vect_t fnmadd (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.51 fnmaddin()**

```
static INLINE vect_t fnmaddin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.52 fmsub()**

```
static INLINE CONST vect_t fmsub (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.53 fmsubin()**

```
static INLINE vect_t fmsubin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.54 eq()**

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.270.2.55 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

**16.270.2.56 mod()**

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static], [inherited]
```

### 16.270.2.57 type\_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

### 16.270.2.58 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

## 16.270.3 Field Documentation

### 16.270.3.1 vect\_size

```
const constexpr size_t vect_size = 16 [static], [constexpr], [inherited]
```

### 16.270.3.2 alignment

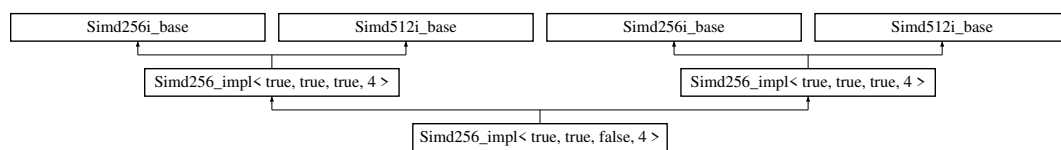
```
const constexpr size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd256\\_int16.inl](#)

## 16.271 Simd256\_impl< true, true, false, 4 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, false, 4 >:



## Data Structures

- union [Converter](#)

## Public Types

- using [scalar\\_t](#) = [uint32\\_t](#)
- using [simdHalf](#) = [Simd128](#)< [scalar\\_t](#) >
- using [scalar\\_t](#) = [uint32\\_t](#)
- using [simdHalf](#) = [Simd128](#)< [scalar\\_t](#) >
- using [vect\\_t](#) = [\\_\\_m256i](#)
- using [vect\\_t](#) = [\\_\\_m512i](#)
- using [half\\_t](#) = [\\_\\_m128i](#)
- using [half\\_t](#) = [\\_\\_m256i](#)

## Static Public Member Functions

- static **INLINE** **CONST** [vect\\_t](#) set1 (const [scalar\\_t](#) x)
- static **INLINE** **CONST** [vect\\_t](#) set (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)
- template<class T >  
static **INLINE** **PURE** [vect\\_t](#) gather (const [scalar\\_t](#) \*const p, const T \*const idx)
- static **INLINE** **PURE** [vect\\_t](#) load (const [scalar\\_t](#) \*const p)

- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>
  - static `INLINE CONST vect_t sra` (const `vect_t` a)
- static `INLINE CONST vect_t greater` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t lesser` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- template<class T >
  - static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>
  - static `INLINE CONST vect_t sra` (const `vect_t` a)
- static `INLINE CONST vect_t greater` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t lesser` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- template<class T >
  - static constexpr bool `valid` (T \*p)
- template<class T >
  - static constexpr bool `valid` (T \*p)
- template<class T >
  - static constexpr bool `compliant` (T n)
- template<class T >
  - static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)

- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7, const `scalar_t` x8, const `scalar_t` x9, const `scalar_t` x10, const `scalar_t` x11, const `scalar_t` x12, const `scalar_t` x13, const `scalar_t` x14, const `scalar_t` x15)
- template<class T >  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- template<uint32\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- template<uint64\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &s1, `vect_t` &s2, const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)

- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 8
- static const constexpr size\_t `alignment` = 32

## 16.271.1 Member Typedef Documentation

### 16.271.1.1 `scalar_t` [1/2]

```
using scalar_t = uint32_t
```

### 16.271.1.2 `simdHalf` [1/2]

```
using simdHalf = Simd128<scalar_t>
```

### 16.271.1.3 `scalar_t` [2/2]

```
using scalar_t = uint32_t
```

### 16.271.1.4 `simdHalf` [2/2]

```
using simdHalf = Simd128<scalar_t>
```

### 16.271.1.5 `vect_t` [1/2]

```
using vect_t = __m256i [inherited]
```

### 16.271.1.6 `vect_t` [2/2]

```
using vect_t = __m512i [inherited]
```

#### 16.271.1.7 half\_t [1/2]

```
using half_t = __m128i [inherited]
```

#### 16.271.1.8 half\_t [2/2]

```
using half_t = __m256i [inherited]
```

### 16.271.2 Member Function Documentation

#### 16.271.2.1 set1() [1/3]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

#### 16.271.2.2 set() [1/4]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

#### 16.271.2.3 gather() [1/3]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.271.2.4 load() [1/3]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.271.2.5 loadu() [1/3]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.271.2.6 store() [1/3]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.271.2.7 storeu()** [1/3]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.271.2.8 stream()** [1/3]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.271.2.9 sra()** [1/2]

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.271.2.10 greater()** [1/2]

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.271.2.11 lesser()** [1/2]

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.271.2.12 greater\_eq()** [1/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.13 lesser\_eq()** [1/2]

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.14 mulhi()** [1/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.15 mulx()** [1/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```



**16.271.2.16 fmaddx() [1/2]**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.17 fmaddxin() [1/2]**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.18 fnmaddx() [1/2]**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.19 fnmaddxin() [1/2]**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.20 fmsubx() [1/2]**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.21 fmsubxin() [1/2]**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.22 hadd\_to\_scal() [1/2]**

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

**16.271.2.23 set1()** [2/3]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

**16.271.2.24 set()** [2/4]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

**16.271.2.25 gather()** [2/3]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

**16.271.2.26 load()** [2/3]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.271.2.27 loadu()** [2/3]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.271.2.28 store()** [2/3]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.271.2.29 storeu()** [2/3]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.271.2.30 stream()** [2/3]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.271.2.31 sra()** [2/2]

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.271.2.32 greater()** [2/2]

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.271.2.33 lesser()** [2/2]

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.271.2.34 greater\_eq()** [2/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.35 lesser\_eq()** [2/2]

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.36 mulhi()** [2/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.37 mulx()** [2/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.271.2.38 fmaddx()** [2/2]

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.271.2.39 fmaddxin()** [2/2]

```
static INLINE vect_t fmaddxin (  
    vect_t & c,
```

```
const vect_t a,
const vect_t b ) [inline], [static]
```

#### 16.271.2.40 fnmaddx() [2/2]

```
static INLINE CONST vect_t fnmaddx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.271.2.41 fnmaddxin() [2/2]

```
static INLINE vect_t fnmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.271.2.42 fmsubx() [2/2]

```
static INLINE CONST vect_t fmsubx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.271.2.43 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.271.2.44 hadd\_to\_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

#### 16.271.2.45 valid() [1/2]

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

#### 16.271.2.46 valid() [2/2]

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

#### 16.271.2.47 compliant() [1/2]

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.271.2.48 compliant() [2/2]**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.271.2.49 set1() [3/3]**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```

**16.271.2.50 set() [3/4]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static], [inherited]
```

**16.271.2.51 set() [4/4]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7,
    const scalar_t x8,
    const scalar_t x9,
    const scalar_t x10,
    const scalar_t x11,
    const scalar_t x12,
    const scalar_t x13,
    const scalar_t x14,
    const scalar_t x15 ) [inline], [static], [inherited]
```

**16.271.2.52 gather() [3/3]**

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

**16.271.2.53 load() [3/3]**

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.271.2.54 loadu()** [3/3]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.271.2.55 store()** [3/3]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static], [inherited]
```

**16.271.2.56 storeu()** [3/3]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static], [inherited]
```

**16.271.2.57 stream()** [3/3]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static], [inherited]
```

**16.271.2.58 sll()** [1/2]

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.59 sll()** [2/2]

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.60 srl()** [1/2]

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.61 srl()** [2/2]

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.62 shuffle\_twice()** [1/2]

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.63 shuffle\_twice()** [2/2]

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.64 shuffle()** [1/2]

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.65 shuffle()** [2/2]

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.271.2.66 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.67 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.68 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.69 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.70 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & s1,  
    vect_t & s2,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.71 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.72 add()** [1/2]

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.73 add()** [2/2]

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.74 addin()** [1/2]

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.75 addin()** [2/2]

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.76 sub()** [1/2]

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.77 sub()** [2/2]

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.78 subin()** [1/2]

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.79 subin()** [2/2]

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```



**16.271.2.80 mullo()** [1/2]

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.81 mullo()** [2/2]

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.82 mul()** [1/2]

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.83 mul()** [2/2]

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.84 fmadd()** [1/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.85 fmadd()** [2/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.86 fmaddin()** [1/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.87 fmaddin()** [2/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.88 fnmadd()** [1/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.89 fnmadd()** [2/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.90 fnmaddin()** [1/2]

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.91 fnmaddin()** [2/2]

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.92 fmsub()** [1/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.93 fmsub()** [2/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.94 fmsubin()** [1/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.271.2.95 fmsubin()** [2/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,
```

```

    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

**16.271.2.96 eq() [1/2]**

```

static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

**16.271.2.97 eq() [2/2]**

```

static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]

```

**16.271.2.98 round() [1/2]**

```

static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]

```

**16.271.2.99 round() [2/2]**

```

static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]

```

**16.271.2.100 mod() [1/2]**

```

static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static], [inherited]

```

**16.271.2.101 mod() [2/2]**

```

static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static], [inherited]

```

**16.271.2.102 type\_string() [1/2]**

```

static const std::string type_string ( ) [inline], [static], [inherited]

```

**16.271.2.103 type\_string()** [2/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.271.2.104 zero()** [1/2]

```
static INLINE CONST vect\_t zero ( ) [inline], [static], [inherited]
```

**16.271.2.105 zero()** [2/2]

```
static INLINE CONST vect\_t zero ( ) [inline], [static], [inherited]
```

**16.271.2.106 vor()**

```
static INLINE CONST vect\_t vor (
    const vect\_t a,
    const vect\_t b ) [inline], [static], [inherited]
```

**16.271.2.107 vxor()**

```
static INLINE CONST vect\_t vxor (
    const vect\_t a,
    const vect\_t b ) [inline], [static], [inherited]
```

**16.271.2.108 vand()**

```
static INLINE CONST vect\_t vand (
    const vect\_t a,
    const vect\_t b ) [inline], [static], [inherited]
```

**16.271.2.109 vandnot()**

```
static INLINE CONST vect\_t vandnot (
    const vect\_t a,
    const vect\_t b ) [inline], [static], [inherited]
```

**16.271.3 Field Documentation****16.271.3.1 vect\_size**

```
static const constexpr size_t vect_size = 8 [static], [constexpr], [inherited]
```

**16.271.3.2 alignment**

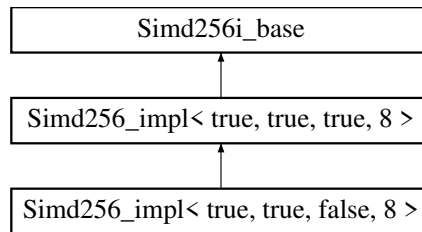
```
static const constexpr size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

## 16.272 Simd256\_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, false, 8 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [scalar\\_t](#) = [uint64\\_t](#)
- using [simdHalf](#) = [Simd128](#)< [scalar\\_t](#) >
- using [vect\\_t](#) = [\\_\\_m256i](#)
- using [half\\_t](#) = [\\_\\_m128i](#)

### Static Public Member Functions

- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmsubx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmsubxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST scalar\\_t hadd\\_to\\_scal](#) (const [vect\\_t](#) a)
- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)

- `template<class T >`  
`static INLINE PURE vect\_t gather (const scalar\_t *const p, const T *const idx)`
- `template<int idx>`  
`static INLINE CONST scalar\_t get (vect\_t v)`
- `static INLINE PURE vect\_t load (const scalar\_t *const p)`
- `static INLINE PURE vect\_t loadu (const scalar\_t *const p)`
- `static INLINE void store (scalar\_t *p, vect\_t v)`
- `static INLINE void storeu (scalar\_t *p, vect\_t v)`
- `static INLINE void stream (scalar\_t *p, const vect\_t v)`
- `template<int s>`  
`static INLINE CONST vect\_t sll (const vect\_t a)`
- `template<int s>`  
`static INLINE CONST vect\_t srl (const vect\_t a)`
- `template<uint8_t s>`  
`static INLINE CONST vect\_t shuffle (const vect\_t a)`
- `static INLINE CONST vect\_t unpacklo\_twice (const vect\_t a, const vect\_t b)`
- `static INLINE CONST vect\_t unpackhi\_twice (const vect\_t a, const vect\_t b)`
- `static INLINE CONST vect\_t unpacklo (const vect\_t a, const vect\_t b)`
- `static INLINE CONST vect\_t unpackhi (const vect\_t a, const vect\_t b)`
- `static INLINE void unpacklohi (vect\_t &l, vect\_t &h, const vect\_t a, const vect\_t b)`
- `template<uint8_t s>`  
`static INLINE CONST vect\_t blend (const vect\_t a, const vect\_t b)`
- `static INLINE CONST vect\_t add (const vect\_t a, const vect\_t b)`
- `static INLINE vect\_t addin (vect\_t &a, const vect\_t b)`
- `static INLINE CONST vect\_t sub (const vect\_t a, const vect\_t b)`
- `static INLINE vect\_t subin (vect\_t &a, const vect\_t b)`
- `static INLINE CONST vect\_t mul (const vect\_t a, const vect\_t b)`
- `static INLINE CONST vect\_t fmadd (const vect\_t c, const vect\_t a, const vect\_t b)`
- `static INLINE vect\_t fmaddin (vect\_t &c, const vect\_t a, const vect\_t b)`
- `static INLINE CONST vect\_t fnmadd (const vect\_t c, const vect\_t a, const vect\_t b)`
- `static INLINE vect\_t fnmaddin (vect\_t &c, const vect\_t a, const vect\_t b)`
- `static INLINE CONST vect\_t fmsub (const vect\_t c, const vect\_t a, const vect\_t b)`
- `static INLINE vect\_t fmsubin (vect\_t &c, const vect\_t a, const vect\_t b)`
- `static INLINE CONST vect\_t eq (const vect\_t a, const vect\_t b)`
- `static INLINE CONST vect\_t round (const vect\_t a)`
- `static INLINE CONST vect\_t mask\_high ()`
- `static INLINE CONST vect\_t mulhi\_fast (vect\_t x, vect\_t y)`
- `static INLINE vect\_t mod (vect\_t &C, const __m256d &P, const __m256d &INVP, const __m256d &NEGP, const vect\_t &POW50REM, const __m256d &MIN, const __m256d &MAX, __m256d &Q, __m256d &T)`
- `static const std::string type\_string ()`
- `static INLINE CONST vect\_t zero ()`

## Static Public Attributes

- `static const constexpr size_t vect\_size = 4`
- `static const constexpr size_t alignment = 32`

## Static Protected Member Functions

- `static INLINE CONST vect\_t signbits (const vect\_t x)`

## 16.272.1 Member Typedef Documentation

### 16.272.1.1 scalar\_t

```
using scalar_t = uint64_t
```

### 16.272.1.2 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

### 16.272.1.3 vect\_t

```
using vect_t = __m256i [inherited]
```

### 16.272.1.4 half\_t

```
using half_t = __m128i [inherited]
```

## 16.272.2 Member Function Documentation

### 16.272.2.1 set1() [1/2]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.272.2.2 set() [1/2]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static]
```

### 16.272.2.3 gather() [1/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.272.2.4 load() [1/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

### 16.272.2.5 loadu() [1/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.272.2.6 store()** [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.272.2.7 storeu()** [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.272.2.8 stream()** [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.272.2.9 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.272.2.10 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.272.2.11 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.272.2.12 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.13 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.14 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```



#### 16.272.2.15 mulx()

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.272.2.16 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.272.2.17 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.272.2.18 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.272.2.19 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.272.2.20 fmsubx()

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.272.2.21 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.272.2.22 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

**16.272.2.23 valid()**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.272.2.24 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.272.2.25 set1() [2/2]**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```

**16.272.2.26 set() [2/2]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3 ) [inline], [static], [inherited]
```

**16.272.2.27 gather() [2/2]**

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static], [inherited]
```

**16.272.2.28 get()**

```
static INLINE CONST scalar_t get (
    vect_t v ) [inline], [static], [inherited]
```

**16.272.2.29 load() [2/2]**

```
static INLINE PURE vect_t load (
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.272.2.30 loadu() [2/2]**

```
static INLINE PURE vect_t loadu (
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.272.2.31 store()** [2/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static], [inherited]
```

**16.272.2.32 storeu()** [2/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static], [inherited]
```

**16.272.2.33 stream()** [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static], [inherited]
```

**16.272.2.34 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.272.2.35 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.272.2.36 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.272.2.37 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.38 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.39 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.40 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.41 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.42 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.43 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.44 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.45 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.46 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.47 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.48 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.49 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.50 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.51 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.52 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.53 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.54 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.272.2.55 round()**

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.272.2.56 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

**16.272.2.57 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static], [inherited]
```

**16.272.2.58 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m256d & P,
    const __m256d & INVP,
    const __m256d & NEGP,
    const vect_t & POW5OREM,
    const __m256d & MIN,
    const __m256d & MAX,
    __m256d & Q,
    __m256d & T ) [static], [inherited]
```

**16.272.2.59 signbits()**

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected], [inherited]
```

**16.272.2.60 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.272.2.61 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.272.3 Field Documentation****16.272.3.1 vect\_size**

```
const constexpr size_t vect_size = 4 [static], [constexpr], [inherited]
```

**16.272.3.2 alignment**

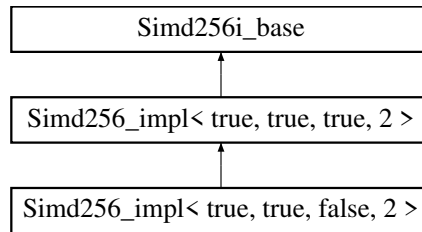
```
const constexpr size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd256\\_int64.inl](#)

## 16.273 Simd256\_impl< true, true, true, 2 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, true, 2 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [vect\\_t](#) = \_\_m256i
- using [half\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int16\_t
- using [simdHalf](#) = Simd128< [scalar\\_t](#) >

### Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7, const [scalar\\_t](#) x8, const [scalar\\_t](#) x9, const [scalar\\_t](#) x10, const [scalar\\_t](#) x11, const [scalar\\_t](#) x12, const [scalar\\_t](#) x13, const [scalar\\_t](#) x14, const [scalar\\_t](#) x15)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- template<uint64\_t s>  
static [INLINE CONST vect\\_t shuffle](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t unpacklo\\_twice](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi\\_twice](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE void unpacklohi](#) ([vect\\_t](#) &s1, [vect\\_t](#) &s2, const [vect\\_t](#) a, const [vect\\_t](#) b)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t blend\\_twice](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 16
- static const constexpr size\_t `alignment` = 32

## 16.273.1 Member Typedef Documentation

### 16.273.1.1 `vect_t`

```
using vect_t = __m256i
```

### 16.273.1.2 `half_t`

```
using half_t = __m128i
```

### 16.273.1.3 `scalar_t`

```
using scalar_t = int16_t
```



#### 16.273.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

### 16.273.2 Member Function Documentation

#### 16.273.2.1 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr]
```

#### 16.273.2.2 compliant()

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr]
```

#### 16.273.2.3 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

#### 16.273.2.4 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7,  
    const scalar_t x8,  
    const scalar_t x9,  
    const scalar_t x10,  
    const scalar_t x11,  
    const scalar_t x12,  
    const scalar_t x13,  
    const scalar_t x14,  
    const scalar_t x15 ) [inline], [static]
```

#### 16.273.2.5 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.273.2.6 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.273.2.7 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.273.2.8 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.273.2.9 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.273.2.10 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.273.2.11 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.273.2.12 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.273.2.13 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.273.2.14 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.273.2.15 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.16 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.17 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.18 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.19 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & s1,  
    vect_t & s2,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.20 blend\_twice()**

```
static INLINE CONST vect_t blend_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.21 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.22 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.23 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.24 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.25 mullo()**

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.26 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.27 mulhi()**

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.28 mulx()**

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.273.2.29 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.30 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.31 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.32 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.33 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.34 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.35 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.36 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.37 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.38 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.273.2.39 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.273.2.40 fmsubxin()

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.41 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.42 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.43 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.44 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.45 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.273.2.46 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.273.2.47 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

**16.273.2.48 mod()**

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

**16.273.2.49 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.273.2.50 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.273.3 Field Documentation****16.273.3.1 vect\_size**

```
const constexpr size_t vect_size = 16 [static], [constexpr]
```

**16.273.3.2 alignment**

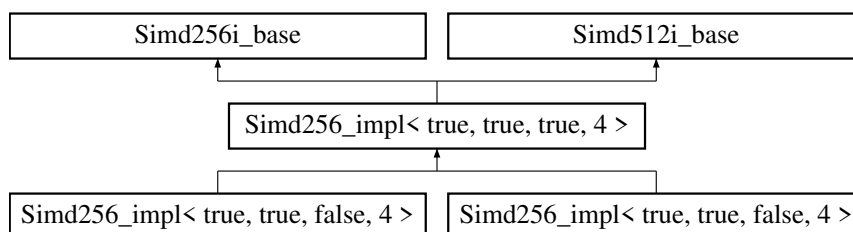
```
const constexpr size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd256\\_int16.inl](#)

**16.274 Simd256\_impl< true, true, true, 4 > Struct Reference**

Inheritance diagram for Simd256\_impl< true, true, true, 4 >:

**Data Structures**

- union [Converter](#)

## Public Types

- using `vect_t` = `__m256i`
- using `half_t` = `__m128i`
- using `scalar_t` = `int32_t`
- using `simdHalf` = `Simd128< scalar_t >`
- using `vect_t` = `__m512i`
- using `half_t` = `__m256i`
- using `scalar_t` = `int32_t`
- using `simdHalf` = `Simd256< scalar_t >`

## Static Public Member Functions

- `template<class T >`  
static constexpr bool `valid` (T \*p)
- `template<class T >`  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- `template<class T >`  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- `template<int s>`  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- `template<int s>`  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- `template<int s>`  
static `INLINE CONST vect_t sra` (const `vect_t` a)
- `template<uint8_t s>`  
static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- `template<uint32_t s>`  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &s1, `vect_t` &s2, const `vect_t` a, const `vect_t` b)
- `template<uint8_t s>`  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)



- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- template<class T >  
static constexpr bool `valid` (T \*p)
- template<class T >  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7, const `scalar_t` x8, const `scalar_t` x9, const `scalar_t` x10, const `scalar_t` x11, const `scalar_t` x12, const `scalar_t` x13, const `scalar_t` x14, const `scalar_t` x15)
- template<class T >  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t sra` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle_twice` (const `vect_t` a)
- template<uint64\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)

- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE vect_t mod` (`vect_t` &C, const `vect_t` &P, const `vect_t` &INVP, const `vect_t` &NEGP, const `vect_t` &MIN, const `vect_t` &MAX, `vect_t` &Q, `vect_t` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 8
- static const constexpr size\_t `alignment` = 32

## 16.274.1 Member Typedef Documentation

### 16.274.1.1 `vect_t` [1/2]

```
using vect_t = __m256i
```

### 16.274.1.2 `half_t` [1/2]

```
using half_t = __m128i
```

### 16.274.1.3 `scalar_t` [1/2]

```
using scalar_t = int32_t
```

### 16.274.1.4 `simdHalf` [1/2]

```
using simdHalf = Simd128<scalar_t>
```

**16.274.1.5 vect\_t** [2/2]

```
using vect_t = __m512i
```

**16.274.1.6 half\_t** [2/2]

```
using half_t = __m256i
```

**16.274.1.7 scalar\_t** [2/2]

```
using scalar_t = int32_t
```

**16.274.1.8 simdHalf** [2/2]

```
using simdHalf = Simd256<scalar_t>
```

**16.274.2 Member Function Documentation****16.274.2.1 valid()** [1/2]

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

**16.274.2.2 compliant()** [1/2]

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

**16.274.2.3 set1()** [1/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.274.2.4 set()** [1/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static]
```

**16.274.2.5 gather()** [1/2]

```
static INLINE PURE vect_t gather (
    const scalar_t *const p,
    const T *const idx ) [inline], [static]
```

**16.274.2.6 load()** [1/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.274.2.7 loadu()** [1/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.274.2.8 store()** [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.274.2.9 storeu()** [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.274.2.10 stream()** [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.274.2.11 sll()** [1/2]

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.274.2.12 srl()** [1/2]

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.274.2.13 sra()** [1/2]

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.274.2.14 shuffle\_twice()** [1/2]

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static]
```

**16.274.2.15 shuffle()** [1/2]

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.274.2.16 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.17 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.18 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.19 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.20 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & s1,  
    vect_t & s2,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.21 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.22 add()** [1/2]

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.23 addin()** [1/2]

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.24 sub()** [1/2]

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.25 subin()** [1/2]

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.26 mullo()** [1/2]

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.27 mul()** [1/2]

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.28 mulhi()** [1/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.29 mulx()** [1/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.274.2.30 fmadd()** [1/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.31 fmaddin()** [1/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.32 fmaddx()** [1/2]

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.33 fmaddxin()** [1/2]

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.34 fnmadd()** [1/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.35 fnmaddin()** [1/2]

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.36 fnmaddx()** [1/2]

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.37 fnmaddxin()** [1/2]

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.38 fmsub()** [1/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.274.2.39 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.40 fmsubx() [1/2]

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.41 fmsubxin() [1/2]

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.42 eq() [1/2]

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.43 greater() [1/2]

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.44 lesser() [1/2]

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.274.2.45 greater\_eq() [1/2]

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```



**16.274.2.46 lesser\_eq()** [1/2]

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.274.2.47 hadd\_to\_scal()** [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

**16.274.2.48 round()** [1/2]

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

**16.274.2.49 mod()** [1/2]

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

**16.274.2.50 valid()** [2/2]

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr]
```

**16.274.2.51 compliant()** [2/2]

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr]
```

**16.274.2.52 set1()** [2/2]

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static]
```

**16.274.2.53 set()** [2/2]

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
```

```
const scalar_t x7,  
const scalar_t x8,  
const scalar_t x9,  
const scalar_t x10,  
const scalar_t x11,  
const scalar_t x12,  
const scalar_t x13,  
const scalar_t x14,  
const scalar_t x15 ) [inline], [static]
```

#### 16.274.2.54 gather() [2/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

#### 16.274.2.55 load() [2/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.274.2.56 loadu() [2/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

#### 16.274.2.57 store() [2/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.274.2.58 storeu() [2/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

#### 16.274.2.59 stream() [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

#### 16.274.2.60 sll() [2/2]

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.274.2.61 srl()** [2/2]

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.274.2.62 sra()** [2/2]

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.274.2.63 shuffle\_twice()** [2/2]

```
static INLINE CONST vect_t shuffle_twice (  
    const vect_t a ) [inline], [static]
```

**16.274.2.64 shuffle()** [2/2]

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.274.2.65 add()** [2/2]

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.66 addin()** [2/2]

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.67 sub()** [2/2]

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.68 subin()** [2/2]

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.69 mullo()** [2/2]

```
static INLINE CONST vect_t mullo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.70 mul()** [2/2]

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.71 mulhi()** [2/2]

```
static INLINE CONST vect_t mulhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.72 mulx()** [2/2]

```
static INLINE CONST vect_t mulx (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.274.2.73 fmadd()** [2/2]

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.74 fmaddin()** [2/2]

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.75 fmaddx()** [2/2]

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.76 fmaddxin()** [2/2]

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.77 fnmadd()** [2/2]

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.78 fnmaddin()** [2/2]

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.79 fnmaddx()** [2/2]

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.80 fnmaddxin()** [2/2]

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.81 fmsub()** [2/2]

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.82 fmsubin()** [2/2]

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.83 fmsubx()** [2/2]

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.84 fmsubxin()** [2/2]

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.274.2.85 eq()** [2/2]

```
static INLINE CONST vect_t eq (  
    const vect_t a,
```

```
const vect_t b ) [inline], [static]
```

#### 16.274.2.86 greater() [2/2]

```
static INLINE CONST vect_t greater (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.274.2.87 lesser() [2/2]

```
static INLINE CONST vect_t lesser (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.274.2.88 greater\_eq() [2/2]

```
static INLINE CONST vect_t greater_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.274.2.89 lesser\_eq() [2/2]

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

#### 16.274.2.90 hadd\_to\_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

#### 16.274.2.91 round() [2/2]

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

#### 16.274.2.92 mod() [2/2]

```
static INLINE vect_t mod (
    vect_t & C,
    const vect_t & P,
    const vect_t & INVP,
    const vect_t & NEGP,
    const vect_t & MIN,
    const vect_t & MAX,
    vect_t & Q,
    vect_t & T ) [inline], [static]
```

#### 16.274.2.93 type\_string() [1/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.274.2.94 zero()** [1/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.274.2.95 type\_string()** [2/2]

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.274.2.96 zero()** [2/2]

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.274.2.97 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.274.2.98 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.274.2.99 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.274.2.100 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.274.3 Field Documentation****16.274.3.1 vect\_size**

```
static const constexpr size_t vect_size = 8 [static], [constexpr]
```

**16.274.3.2 alignment**

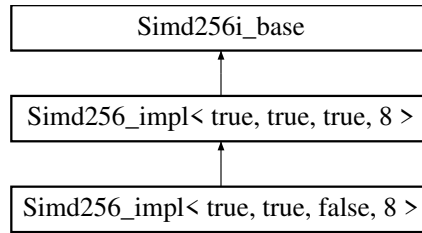
```
static const constexpr size_t alignment = 32 [static], [constexpr]
```

The documentation for this struct was generated from the following files:

- [simd256\\_int32.inl](#)
- [simd512\\_int32.inl](#)

## 16.275 Simd256\_impl< true, true, true, 8 > Struct Reference

Inheritance diagram for Simd256\_impl< true, true, true, 8 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [vect\\_t](#) = \_\_m256i
- using [half\\_t](#) = \_\_m128i
- using [scalar\\_t](#) = int64\_t
- using [simdHalf](#) = Simd128< [scalar\\_t](#) >

### Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- template<int idx>  
static [INLINE CONST scalar\\_t get](#) ([vect\\_t](#) v)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE](#) void [store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t srl](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t shuffle](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t unpacklo\\_twice](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi\\_twice](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpacklo](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t unpackhi](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE](#) void [unpacklohi](#) ([vect\\_t](#) &l, [vect\\_t](#) &h, const [vect\\_t](#) a, const [vect\\_t](#) b)
- template<uint8\_t s>  
static [INLINE CONST vect\\_t blend](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)



- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mullo` (`vect_t` a, `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulx` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmaddxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const `__m256d` &P, const `__m256d` &INVP, const `__m256d` &NEGP, const `vect_t` &POW50REM, const `__m256d` &MIN, const `__m256d` &MAX, `__m256d` &Q, `__m256d` &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 4
- static const constexpr size\_t `alignment` = 32

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

## 16.275.1 Member Typedef Documentation

### 16.275.1.1 vect\_t

```
using vect_t = __m256i
```

### 16.275.1.2 half\_t

```
using half_t = __m128i
```

### 16.275.1.3 scalar\_t

```
using scalar_t = int64_t
```

### 16.275.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

## 16.275.2 Member Function Documentation

### 16.275.2.1 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr]
```

### 16.275.2.2 compliant()

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr]
```

### 16.275.2.3 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.275.2.4 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static]
```

### 16.275.2.5 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.275.2.6 get()

```
static INLINE CONST scalar_t get (  
    vect_t v ) [inline], [static]
```

### 16.275.2.7 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.275.2.8 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.275.2.9 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.275.2.10 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.275.2.11 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.275.2.12 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.275.2.13 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.275.2.14 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.275.2.15 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.275.2.16 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.17 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.18 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.19 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.20 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.21 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.22 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.23 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.24 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.25 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.26 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.275.2.27 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.28 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.29 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.30 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.31 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.32 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.33 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.34 fnmaddin()**

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.35 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.36 fnmaddxin()**

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.37 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.38 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.39 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.275.2.40 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.275.2.41 eq()

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.275.2.42 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.275.2.43 lesser()

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.275.2.44 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.275.2.45 lesser\_eq()

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.275.2.46 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

#### 16.275.2.47 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

#### 16.275.2.48 mask\_high()

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

**16.275.2.49 mulhi\_fast()**

```

INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]

```

**16.275.2.50 mod()**

```

INLINE vect_t mod (
    vect_t & C,
    const __m256d & P,
    const __m256d & INV_P,
    const __m256d & NEG_P,
    const vect_t & POW50REM,
    const __m256d & MIN,
    const __m256d & MAX,
    __m256d & Q,
    __m256d & T ) [static]

```

**16.275.2.51 signbits()**

```

static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]

```

**16.275.2.52 type\_string()**

```

static const std::string type_string ( ) [inline], [static], [inherited]

```

**16.275.2.53 zero()**

```

static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]

```

**16.275.3 Field Documentation****16.275.3.1 vect\_size**

```

const constexpr size_t vect_size = 4 [static], [constexpr]

```

**16.275.3.2 alignment**

```

const constexpr size_t alignment = 32 [static], [constexpr]

```

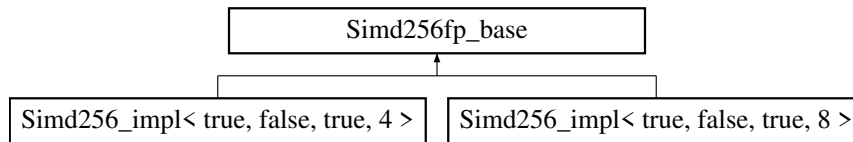
The documentation for this struct was generated from the following file:

- [simd256\\_int64.inl](#)

**16.276 Simd256fp\_base Struct Reference**

Inheritance diagram for Simd256fp\_base:



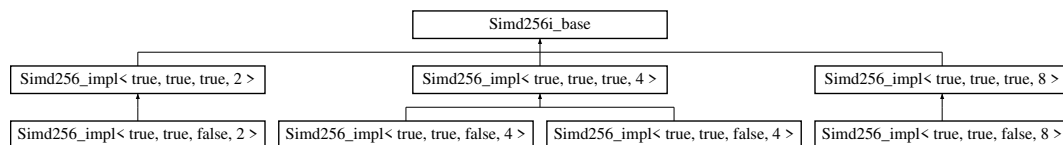


The documentation for this struct was generated from the following file:

- [simd256.inl](#)

## 16.277 Simd256i\_base Struct Reference

Inheritance diagram for Simd256i\_base:



### Public Types

- using [vect\\_t](#) = \_\_m256i

### Static Public Member Functions

- static const std::string [type\\_string](#) ()
- static [INLINE CONST vect\\_t zero](#) ()

## 16.277.1 Member Typedef Documentation

### 16.277.1.1 vect\_t

using [vect\\_t](#) = \_\_m256i

## 16.277.2 Member Function Documentation

### 16.277.2.1 type\_string()

```
static const std::string type_string ( ) [inline], [static]
```

### 16.277.2.2 zero()

```
static INLINE CONST vect\_t zero ( ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd256.inl](#)

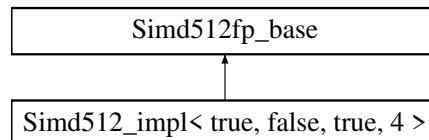
## 16.278 Simd512\_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- [simd512.inl](#)

## 16.279 Simd512\_impl< true, false, true, 4 > Struct Reference

Inheritance diagram for Simd512\_impl< true, false, true, 4 >:



### Static Public Member Functions

- static const std::string [type\\_string](#) ()

### 16.279.1 Member Function Documentation

#### 16.279.1.1 type\_string()

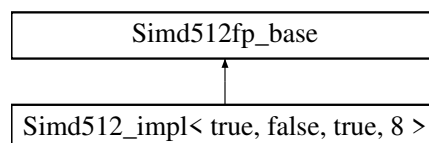
```
static const std::string type_string ( ) [inline], [static], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd512\\_float.inl](#)

## 16.280 Simd512\_impl< true, false, true, 8 > Struct Reference

Inheritance diagram for Simd512\_impl< true, false, true, 8 >:



### Public Types

- using [vect\\_t](#) = \_\_m512d
- using [scalar\\_t](#) = double

### Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST vect\\_t zero](#) ()
- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)

- static `INLINE CONST vect_t set` (const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7, const `scalar_t` x8)
- template<class T >  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (const `scalar_t` \*p, const `vect_t` v)
- static `INLINE void storeu` (const `scalar_t` \*p, const `vect_t` v)
- static `INLINE void stream` (const `scalar_t` \*p, const `vect_t` v)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t blendv` (const `vect_t` a, const `vect_t` b, const `vect_t` mask)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t mulin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t div` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmaddin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t lesser_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t greater_eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t floor` (const `vect_t` a)
- static `INLINE CONST vect_t ceil` (const `vect_t` a)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t hadd` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- static const std::string `type_string` ()

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 8
- static const constexpr size\_t `alignment` = 64

## 16.280.1 Member Typedef Documentation

### 16.280.1.1 vect\_t

using `vect_t` = \_\_m512d

### 16.280.1.2 scalar\_t

```
using scalar_t = double
```

## 16.280.2 Member Function Documentation

### 16.280.2.1 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr]
```

### 16.280.2.2 compliant()

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr]
```

### 16.280.2.3 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static]
```

### 16.280.2.4 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.280.2.5 set()

```
static INLINE CONST vect_t set (  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7,  
    const scalar_t x8 ) [inline], [static]
```

### 16.280.2.6 gather()

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

### 16.280.2.7 load()

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.280.2.8 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.280.2.9 store()**

```
static INLINE void store (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.280.2.10 storeu()**

```
static INLINE void storeu (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.280.2.11 stream()**

```
static INLINE void stream (  
    const scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.280.2.12 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.280.2.13 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.14 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.15 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.16 blendv()**

```
static INLINE CONST vect_t blendv (  
    const vect_t a,  
    const vect_t b,  
    const vect_t mask ) [inline], [static]
```

**16.280.2.17 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.18 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.19 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.20 subin()**

```
static INLINE CONST vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.21 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.22 mulin()**

```
static INLINE CONST vect_t mulin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.23 div()**

```
static INLINE CONST vect_t div (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.24 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.25 fmaddin()**

```
static INLINE CONST vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.26 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.27 fnmaddin()**

```
static INLINE CONST vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.28 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.29 fmsubin()**

```
static INLINE CONST vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.30 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.31 lesser()**

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.280.2.32 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

### 16.280.2.33 greater()

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

### 16.280.2.34 greater\_eq()

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

### 16.280.2.35 floor()

```
static INLINE CONST vect_t floor (  
    const vect_t a ) [inline], [static]
```

### 16.280.2.36 ceil()

```
static INLINE CONST vect_t ceil (  
    const vect_t a ) [inline], [static]
```

### 16.280.2.37 round()

```
static INLINE CONST vect_t round (  
    const vect_t a ) [inline], [static]
```

### 16.280.2.38 hadd()

```
static INLINE CONST vect_t hadd (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

### 16.280.2.39 hadd\_to\_scal()

```
static INLINE CONST scalar_t hadd_to_scal (  
    const vect_t a ) [inline], [static]
```

### 16.280.2.40 type\_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

## 16.280.3 Field Documentation

### 16.280.3.1 vect\_size

```
const constexpr size_t vect_size = 8 [static], [constexpr]
```



### 16.280.3.2 alignment

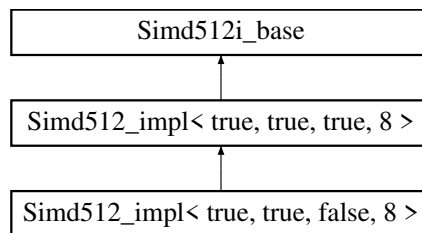
```
const constexpr size_t alignment = 64 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd512\\_double.inl](#)

## 16.281 Simd512\_impl< true, true, false, 8 > Struct Reference

Inheritance diagram for Simd512\_impl< true, true, false, 8 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [scalar\\_t](#) = [uint64\\_t](#)
- using [simdHalf](#) = [Simd256](#)< [scalar\\_t](#) >
- using [vect\\_t](#) = [\\_\\_m512i](#)
- using [half\\_t](#) = [\\_\\_m256i](#)

### Static Public Member Functions

- static [INLINE CONST vect\\_t set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST vect\\_t set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)
- template<class T >  
static [INLINE PURE vect\\_t gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE vect\\_t load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE vect\\_t loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE void store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- template<[uint8\\_t](#) k>  
static [INLINE void maskstore](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE void stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST vect\\_t sra](#) (const [vect\\_t](#) a)
- static [INLINE CONST vect\\_t greater](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t greater\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t lesser\\_eq](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mullo](#) ([vect\\_t](#) a, [vect\\_t](#) b)
- static [INLINE CONST vect\\_t mulx](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t fnmaddx](#) (const [vect\\_t](#) c, const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE vect\\_t fnmaddxin](#) ([vect\\_t](#) &c, const [vect\\_t](#) a, const [vect\\_t](#) b)

- static `INLINE CONST vect_t fmsubx` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubxin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST scalar_t hadd_to_scal` (const `vect_t` a)
- template<class T >  
static constexpr bool `valid` (T \*p)
- template<class T >  
static constexpr bool `compliant` (T n)
- static `INLINE CONST vect_t set1` (const `scalar_t` x)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3, const `scalar_t` x4, const `scalar_t` x5, const `scalar_t` x6, const `scalar_t` x7)
- static `INLINE CONST vect_t set` (const `scalar_t` x0, const `scalar_t` x1, const `scalar_t` x2, const `scalar_t` x3)
- template<class T >  
static `INLINE PURE vect_t gather` (const `scalar_t` \*const p, const T \*const idx)
- static `INLINE PURE vect_t load` (const `scalar_t` \*const p)
- static `INLINE PURE vect_t loadu` (const `scalar_t` \*const p)
- static `INLINE void store` (`scalar_t` \*p, `vect_t` v)
- template<uint8\_t k>  
static `INLINE void maskstore` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void storeu` (`scalar_t` \*p, `vect_t` v)
- static `INLINE void stream` (`scalar_t` \*p, const `vect_t` v)
- template<int s>  
static `INLINE CONST vect_t sll` (const `vect_t` a)
- template<int s>  
static `INLINE CONST vect_t srl` (const `vect_t` a)
- template<uint8\_t s>  
static `INLINE CONST vect_t shuffle` (const `vect_t` a)
- static `INLINE CONST vect_t unpacklo_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi_twice` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpacklo` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t unpackhi` (const `vect_t` a, const `vect_t` b)
- static `INLINE void unpacklohi` (`vect_t` &l, `vect_t` &h, const `vect_t` a, const `vect_t` b)
- template<uint8\_t s>  
static `INLINE CONST vect_t blend` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t add` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t addin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t sub` (const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t subin` (`vect_t` &a, const `vect_t` b)
- static `INLINE CONST vect_t mul` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmadin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fnmadd` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fnmadin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t fmsub` (const `vect_t` c, const `vect_t` a, const `vect_t` b)
- static `INLINE vect_t fmsubin` (`vect_t` &c, const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t eq` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t round` (const `vect_t` a)
- static `INLINE CONST vect_t mask_high` ()
- static `INLINE CONST vect_t mulhi_fast` (`vect_t` x, `vect_t` y)
- static `INLINE vect_t mod` (`vect_t` &C, const \_\_m512d &P, const \_\_m512d &INVP, const \_\_m512d &NEGP, const `vect_t` &POW50REM, const \_\_m512d &MIN, const \_\_m512d &MAX, \_\_m512d &Q, \_\_m512d &T)
- static const std::string `type_string` ()
- static `INLINE CONST vect_t zero` ()
- static `INLINE CONST vect_t vor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vxor` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vand` (const `vect_t` a, const `vect_t` b)
- static `INLINE CONST vect_t vandnot` (const `vect_t` a, const `vect_t` b)

## Static Public Attributes

- static const constexpr size\_t `vect_size` = 8
- static const constexpr size\_t `alignment` = 64

## Static Protected Member Functions

- static `INLINE CONST vect_t signbits` (const `vect_t` x)

## 16.281.1 Member Typedef Documentation

### 16.281.1.1 `scalar_t`

```
using scalar_t = uint64_t
```

### 16.281.1.2 `simdHalf`

```
using simdHalf = Simd256<scalar_t>
```

### 16.281.1.3 `vect_t`

```
using vect_t = __m512i [inherited]
```

### 16.281.1.4 `half_t`

```
using half_t = __m256i [inherited]
```

## 16.281.2 Member Function Documentation

### 16.281.2.1 `set1()` [1/2]

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.281.2.2 `set()` [1/3]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

**16.281.2.3 gather()** [1/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

**16.281.2.4 load()** [1/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.281.2.5 loadu()** [1/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.281.2.6 store()** [1/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.281.2.7 maskstore()** [1/2]

```
static INLINE void maskstore (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.281.2.8 storeu()** [1/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.281.2.9 stream()** [1/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.281.2.10 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.281.2.11 greater()**

```
static INLINE CONST vect_t greater (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.281.2.12 lesser()**

```
static INLINE CONST vect_t lesser (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.281.2.13 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.14 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.15 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.281.2.16 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.17 fmaddx()**

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.18 fmaddxin()**

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.19 fnmaddx()**

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.281.2.20 fmmaddxin()**

```
static INLINE vect_t fmmaddxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.281.2.21 fmsubx()**

```
static INLINE CONST vect_t fmsubx (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.281.2.22 fmsubxin()**

```
static INLINE vect_t fmsubxin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.281.2.23 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

**16.281.2.24 valid()**

```
static constexpr bool valid (
    T * p ) [inline], [static], [constexpr], [inherited]
```

**16.281.2.25 compliant()**

```
static constexpr bool compliant (
    T n ) [inline], [static], [constexpr], [inherited]
```

**16.281.2.26 set1() [2/2]**

```
static INLINE CONST vect_t set1 (
    const scalar_t x ) [inline], [static], [inherited]
```

**16.281.2.27 set() [2/3]**

```
static INLINE CONST vect_t set (
    const scalar_t x0,
    const scalar_t x1,
    const scalar_t x2,
    const scalar_t x3,
    const scalar_t x4,
    const scalar_t x5,
    const scalar_t x6,
    const scalar_t x7 ) [inline], [static], [inherited]
```

**16.281.2.28 set()** [3/3]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static], [inherited]
```

**16.281.2.29 gather()** [2/2]

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static], [inherited]
```

**16.281.2.30 load()** [2/2]

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.281.2.31 loadu()** [2/2]

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static], [inherited]
```

**16.281.2.32 store()** [2/2]

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static], [inherited]
```

**16.281.2.33 maskstore()** [2/2]

```
static INLINE void maskstore (  
    scalar_t * p,  
    vect_t v ) [inline], [static], [inherited]
```

**16.281.2.34 storeu()** [2/2]

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static], [inherited]
```

**16.281.2.35 stream()** [2/2]

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static], [inherited]
```

**16.281.2.36 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.281.2.37 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.281.2.38 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static], [inherited]
```

**16.281.2.39 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.40 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.41 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.42 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.43 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.44 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```



**16.281.2.45 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.46 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.47 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.48 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.49 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.50 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.51 fmaddin()**

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.52 fnmadd()**

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.53 fnmaddin()**

```
static INLINE vect_t fnmaddin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.54 fmsub()**

```
static INLINE CONST vect_t fmsub (
    const vect_t c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.55 fmsubin()**

```
static INLINE vect_t fmsubin (
    vect_t & c,
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.56 eq()**

```
static INLINE CONST vect_t eq (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.281.2.57 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static], [inherited]
```

**16.281.2.58 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static], [inherited]
```

**16.281.2.59 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static], [inherited]
```

**16.281.2.60 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m512d & P,
    const __m512d & INV_P,
    const __m512d & NEG_P,
    const vect_t & POW50REM,
    const __m512d & MIN,
    const __m512d & MAX,
```

```
__m512d & Q,  
__m512d & T ) [static], [inherited]
```

#### 16.281.2.61 signbits()

```
static INLINE CONST vect_t signbits (  
    const vect_t x ) [inline], [static], [protected], [inherited]
```

#### 16.281.2.62 type\_string()

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

#### 16.281.2.63 zero()

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

#### 16.281.2.64 vor()

```
static INLINE CONST vect_t vor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.281.2.65 vxor()

```
static INLINE CONST vect_t vxor (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.281.2.66 vand()

```
static INLINE CONST vect_t vand (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

#### 16.281.2.67 vandnot()

```
static INLINE CONST vect_t vandnot (  
    const vect_t a,  
    const vect_t b ) [inline], [static], [inherited]
```

### 16.281.3 Field Documentation

#### 16.281.3.1 vect\_size

```
const constexpr size_t vect_size = 8 [static], [constexpr], [inherited]
```

### 16.281.3.2 alignment

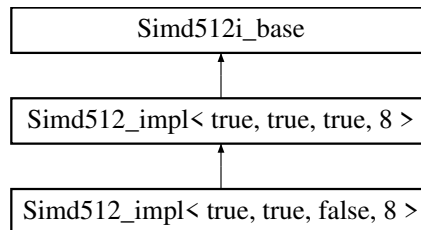
```
const constexpr size_t alignment = 64 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- [simd512\\_int64.inl](#)

## 16.282 Simd512\_impl< true, true, true, 8 > Struct Reference

Inheritance diagram for Simd512\_impl< true, true, true, 8 >:



### Data Structures

- union [Converter](#)

### Public Types

- using [vect\\_t](#) = \_\_m512i
- using [half\\_t](#) = \_\_m256i
- using [scalar\\_t](#) = [int64\\_t](#)
- using [simdHalf](#) = [Simd256](#)< [scalar\\_t](#) >

### Static Public Member Functions

- template<class T >  
static constexpr bool [valid](#) (T \*p)
- template<class T >  
static constexpr bool [compliant](#) (T n)
- static [INLINE CONST](#) [vect\\_t](#) [set1](#) (const [scalar\\_t](#) x)
- static [INLINE CONST](#) [vect\\_t](#) [set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3, const [scalar\\_t](#) x4, const [scalar\\_t](#) x5, const [scalar\\_t](#) x6, const [scalar\\_t](#) x7)
- static [INLINE CONST](#) [vect\\_t](#) [set](#) (const [scalar\\_t](#) x0, const [scalar\\_t](#) x1, const [scalar\\_t](#) x2, const [scalar\\_t](#) x3)
- template<class T >  
static [INLINE PURE](#) [vect\\_t](#) [gather](#) (const [scalar\\_t](#) \*const p, const T \*const idx)
- static [INLINE PURE](#) [vect\\_t](#) [load](#) (const [scalar\\_t](#) \*const p)
- static [INLINE PURE](#) [vect\\_t](#) [loadu](#) (const [scalar\\_t](#) \*const p)
- static [INLINE](#) void [store](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- template<uint8\_t k>  
static [INLINE](#) void [maskstore](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [storeu](#) ([scalar\\_t](#) \*p, [vect\\_t](#) v)
- static [INLINE](#) void [stream](#) ([scalar\\_t](#) \*p, const [vect\\_t](#) v)
- template<int s>  
static [INLINE CONST](#) [vect\\_t](#) [sll](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST](#) [vect\\_t](#) [srl](#) (const [vect\\_t](#) a)
- template<int s>  
static [INLINE CONST](#) [vect\\_t](#) [sra](#) (const [vect\\_t](#) a)

- `template<uint8_t s>`  
`static INLINE CONST vect_t shuffle (const vect_t a)`
- `static INLINE CONST vect_t unpacklo_twice (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi_twice (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpacklo (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t unpackhi (const vect_t a, const vect_t b)`
- `static INLINE void unpacklohi (vect_t &l, vect_t &h, const vect_t a, const vect_t b)`
- `template<uint8_t s>`  
`static INLINE CONST vect_t blend (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t add (const vect_t a, const vect_t b)`
- `static INLINE vect_t addin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t sub (const vect_t a, const vect_t b)`
- `static INLINE vect_t subin (vect_t &a, const vect_t b)`
- `static INLINE CONST vect_t mullo (vect_t a, vect_t b)`
- `static INLINE CONST vect_t mul (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t mulx (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmadddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmadddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmadd (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fnmaddx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fnmaddxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsub (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t fmsubx (const vect_t c, const vect_t a, const vect_t b)`
- `static INLINE vect_t fmsubxin (vect_t &c, const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t greater_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t lesser_eq (const vect_t a, const vect_t b)`
- `static INLINE CONST scalar_t hadd_to_scal (const vect_t a)`
- `static INLINE CONST vect_t round (const vect_t a)`
- `static INLINE CONST vect_t mask_high ()`
- `static INLINE CONST vect_t mulhi_fast (vect_t x, vect_t y)`
- `static INLINE vect_t mod (vect_t &C, const __m512d &P, const __m512d &INVP, const __m512d &NEGP, const vect_t &POW50REM, const __m512d &MIN, const __m512d &MAX, __m512d &Q, __m512d &T)`
- `static const std::string type_string ()`
- `static INLINE CONST vect_t zero ()`
- `static INLINE CONST vect_t vor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vxor (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vand (const vect_t a, const vect_t b)`
- `static INLINE CONST vect_t vandnot (const vect_t a, const vect_t b)`

## Static Public Attributes

- `static const constexpr size_t vect_size = 8`
- `static const constexpr size_t alignment = 64`

## Static Protected Member Functions

- `static INLINE CONST vect_t signbits (const vect_t x)`

## 16.282.1 Member Typedef Documentation

### 16.282.1.1 vect\_t

```
using vect_t = __m512i
```

### 16.282.1.2 half\_t

```
using half_t = __m256i
```

### 16.282.1.3 scalar\_t

```
using scalar_t = int64_t
```

### 16.282.1.4 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

## 16.282.2 Member Function Documentation

### 16.282.2.1 valid()

```
static constexpr bool valid (  
    T * p ) [inline], [static], [constexpr]
```

### 16.282.2.2 compliant()

```
static constexpr bool compliant (  
    T n ) [inline], [static], [constexpr]
```

### 16.282.2.3 set1()

```
static INLINE CONST vect_t set1 (  
    const scalar_t x ) [inline], [static]
```

### 16.282.2.4 set() [1/2]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3,  
    const scalar_t x4,  
    const scalar_t x5,  
    const scalar_t x6,  
    const scalar_t x7 ) [inline], [static]
```

**16.282.2.5 set()** [2/2]

```
static INLINE CONST vect_t set (  
    const scalar_t x0,  
    const scalar_t x1,  
    const scalar_t x2,  
    const scalar_t x3 ) [inline], [static]
```

**16.282.2.6 gather()**

```
static INLINE PURE vect_t gather (  
    const scalar_t *const p,  
    const T *const idx ) [inline], [static]
```

**16.282.2.7 load()**

```
static INLINE PURE vect_t load (  
    const scalar_t *const p ) [inline], [static]
```

**16.282.2.8 loadu()**

```
static INLINE PURE vect_t loadu (  
    const scalar_t *const p ) [inline], [static]
```

**16.282.2.9 store()**

```
static INLINE void store (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.282.2.10 maskstore()**

```
static INLINE void maskstore (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.282.2.11 storeu()**

```
static INLINE void storeu (  
    scalar_t * p,  
    vect_t v ) [inline], [static]
```

**16.282.2.12 stream()**

```
static INLINE void stream (  
    scalar_t * p,  
    const vect_t v ) [inline], [static]
```

**16.282.2.13 sll()**

```
static INLINE CONST vect_t sll (  
    const vect_t a ) [inline], [static]
```

**16.282.2.14 srl()**

```
static INLINE CONST vect_t srl (  
    const vect_t a ) [inline], [static]
```

**16.282.2.15 sra()**

```
static INLINE CONST vect_t sra (  
    const vect_t a ) [inline], [static]
```

**16.282.2.16 shuffle()**

```
static INLINE CONST vect_t shuffle (  
    const vect_t a ) [inline], [static]
```

**16.282.2.17 unpacklo\_twice()**

```
static INLINE CONST vect_t unpacklo_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.18 unpackhi\_twice()**

```
static INLINE CONST vect_t unpackhi_twice (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.19 unpacklo()**

```
static INLINE CONST vect_t unpacklo (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.20 unpackhi()**

```
static INLINE CONST vect_t unpackhi (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.21 unpacklohi()**

```
static INLINE void unpacklohi (  
    vect_t & l,  
    vect_t & h,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```



**16.282.2.22 blend()**

```
static INLINE CONST vect_t blend (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.23 add()**

```
static INLINE CONST vect_t add (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.24 addin()**

```
static INLINE vect_t addin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.25 sub()**

```
static INLINE CONST vect_t sub (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.26 subin()**

```
static INLINE vect_t subin (  
    vect_t & a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.27 mullo()**

```
static INLINE CONST vect_t mullo (  
    vect_t a,  
    vect_t b ) [inline], [static]
```

**16.282.2.28 mul()**

```
static INLINE CONST vect_t mul (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.29 mulx()**

```
static INLINE CONST vect_t mulx (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.30 fmadd()**

```
static INLINE CONST vect_t fmadd (  
    const vect_t c,
```

```
const vect_t a,  
const vect_t b ) [inline], [static]
```

#### 16.282.2.31 fmaddin()

```
static INLINE vect_t fmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.32 fmaddx()

```
static INLINE CONST vect_t fmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.33 fmaddxin()

```
static INLINE vect_t fmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.34 fnmadd()

```
static INLINE CONST vect_t fnmadd (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.35 fnmaddin()

```
static INLINE vect_t fnmaddin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.36 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

#### 16.282.2.37 fnmaddxin()

```
static INLINE vect_t fnmaddxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.38 fmsub()**

```
static INLINE CONST vect_t fmsub (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.39 fmsubin()**

```
static INLINE vect_t fmsubin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.40 fmsubx()**

```
static INLINE CONST vect_t fmsubx (  
    const vect_t c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.41 fmsubxin()**

```
static INLINE vect_t fmsubxin (  
    vect_t & c,  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.42 eq()**

```
static INLINE CONST vect_t eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.43 greater()**

```
static INLINE CONST vect_t greater (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.44 lesser()**

```
static INLINE CONST vect_t lesser (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.45 greater\_eq()**

```
static INLINE CONST vect_t greater_eq (  
    const vect_t a,  
    const vect_t b ) [inline], [static]
```

**16.282.2.46 lesser\_eq()**

```
static INLINE CONST vect_t lesser_eq (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.282.2.47 hadd\_to\_scal()**

```
static INLINE CONST scalar_t hadd_to_scal (
    const vect_t a ) [inline], [static]
```

**16.282.2.48 round()**

```
static INLINE CONST vect_t round (
    const vect_t a ) [inline], [static]
```

**16.282.2.49 mask\_high()**

```
static INLINE CONST vect_t mask_high ( ) [inline], [static]
```

**16.282.2.50 mulhi\_fast()**

```
INLINE CONST vect_t mulhi_fast (
    vect_t x,
    vect_t y ) [static]
```

**16.282.2.51 mod()**

```
INLINE vect_t mod (
    vect_t & C,
    const __m512d & P,
    const __m512d & INVP,
    const __m512d & NEGP,
    const vect_t & POW50REM,
    const __m512d & MIN,
    const __m512d & MAX,
    __m512d & Q,
    __m512d & T ) [static]
```

**16.282.2.52 signbits()**

```
static INLINE CONST vect_t signbits (
    const vect_t x ) [inline], [static], [protected]
```

**16.282.2.53 type\_string()**

```
static const std::string type_string ( ) [inline], [static], [inherited]
```

**16.282.2.54 zero()**

```
static INLINE CONST vect_t zero ( ) [inline], [static], [inherited]
```

**16.282.2.55 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.282.2.56 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.282.2.57 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.282.2.58 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static], [inherited]
```

**16.282.3 Field Documentation****16.282.3.1 vect\_size**

```
const constexpr size_t vect_size = 8 [static], [constexpr]
```

**16.282.3.2 alignment**

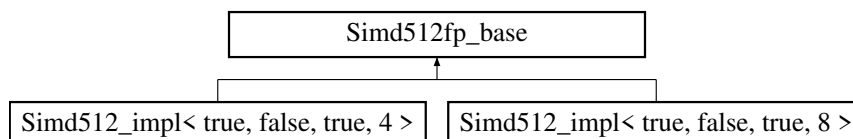
```
const constexpr size_t alignment = 64 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- [simd512\\_int64.inl](#)

**16.283 Simd512fp\_base Struct Reference**

Inheritance diagram for Simd512fp\_base:

**Static Public Member Functions**

- static const std::string [type\\_string](#) ()

## 16.283.1 Member Function Documentation

### 16.283.1.1 type\_string()

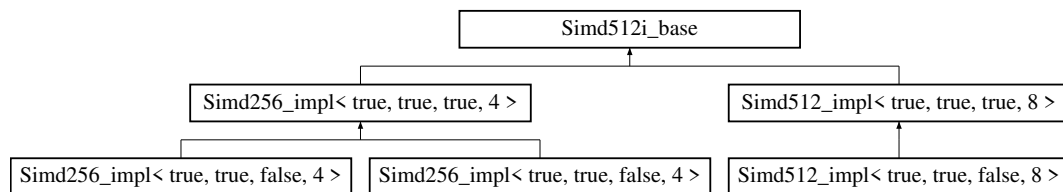
static const std::string type\_string ( ) [inline], [static]

The documentation for this struct was generated from the following file:

- [simd512.inl](#)

## 16.284 Simd512i\_base Struct Reference

Inheritance diagram for Simd512i\_base:



## Public Types

- using [vect\\_t](#) = \_\_m512i

## Static Public Member Functions

- static const std::string [type\\_string](#) ( )
- static [INLINE CONST vect\\_t zero](#) ( )
- static [INLINE CONST vect\\_t vor](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t vxor](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t vand](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)
- static [INLINE CONST vect\\_t vandnot](#) (const [vect\\_t](#) a, const [vect\\_t](#) b)

## 16.284.1 Member Typedef Documentation

### 16.284.1.1 vect\_t

using [vect\\_t](#) = \_\_m512i

## 16.284.2 Member Function Documentation

### 16.284.2.1 type\_string()

static const std::string type\_string ( ) [inline], [static]

### 16.284.2.2 zero()

static [INLINE CONST vect\\_t zero](#) ( ) [inline], [static]

**16.284.2.3 vor()**

```
static INLINE CONST vect_t vor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.284.2.4 vxor()**

```
static INLINE CONST vect_t vxor (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.284.2.5 vand()**

```
static INLINE CONST vect_t vand (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

**16.284.2.6 vandnot()**

```
static INLINE CONST vect_t vandnot (
    const vect_t a,
    const vect_t b ) [inline], [static]
```

The documentation for this struct was generated from the following file:

- [simd512.inl](#)

**16.285 SimdChooser< T, bool, bool > Struct Template Reference**

```
#include <fflas_simd.h>
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

**16.286 SimdChooser< T, false, b > Struct Template Reference**

```
#include <fflas_simd.h>
```

**Public Types**

- using [value](#) = NoSimd< T >

**16.286.1 Member Typedef Documentation****16.286.1.1 value**

```
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

**16.287 SimdChooser< T, true, false > Struct Template Reference**

```
#include <fflas_simd.h>
```

## Public Types

- using [value](#) = [NoSimd](#)< T >

### 16.287.1 Member Typedef Documentation

#### 16.287.1.1 value

using [value](#) = [NoSimd](#)<T>

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

### 16.288 [SimdChooser](#)< T, true, true > Struct Template Reference

```
#include <fflas_simd.h>
```

## Public Types

- using [value](#) = [NoSimd](#)< T >

### 16.288.1 Member Typedef Documentation

#### 16.288.1.1 value

using [value](#) = [NoSimd](#)<T>

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

### 16.289 [simdToType](#)< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

### 16.290 Single Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

### 16.291 [Sparse](#)< Field, SparseMatrix\_t, IdxT, PtrT > Struct Template Reference

The documentation for this struct was generated from the following file:

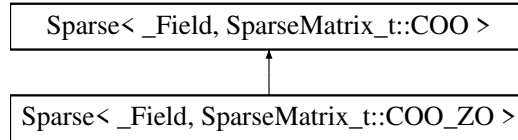
- [fflas\\_sparse.h](#)



## 16.292 Sparse< \_Field, SparseMatrix\_t::COO > Struct Template Reference

```
#include <coo.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::COO >:



### Public Types

- using [Field](#) = \_Field

### Data Fields

- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [row](#) = nullptr
- \_Field::Element\_ptr [dat](#)
- bool [delayed](#) = false
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0

### 16.292.1 Member Typedef Documentation

#### 16.292.1.1 Field

```
using Field = _Field
```

### 16.292.2 Field Documentation

#### 16.292.2.1 col

```
index\_t* col = nullptr
```

#### 16.292.2.2 row

```
index\_t* row = nullptr
```

#### 16.292.2.3 dat

```
_Field::Element_ptr dat
```

**16.292.2.4 delayed**

```
bool delayed = false
```

**16.292.2.5 kmax**

```
uint64_t kmax = 0
```

**16.292.2.6 m**

```
index_t m = 0
```

**16.292.2.7 n**

```
index_t n = 0
```

**16.292.2.8 nnz**

```
uint64_t nnz = 0
```

**16.292.2.9 nElements**

```
uint64_t nElements = 0
```

**16.292.2.10 maxrow**

```
uint64_t maxrow = 0
```

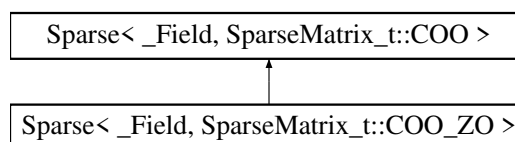
The documentation for this struct was generated from the following file:

- [coo.h](#)

## 16.293 Sparse<\_Field, SparseMatrix\_t::COO\_ZO > Struct Template Reference

```
#include <coo.h>
```

Inheritance diagram for Sparse<\_Field, SparseMatrix\_t::COO\_ZO >:

**Public Types**

- using [Field](#) = \_Field

**Data Fields**

- \_Field::Element [cst](#) = 1
- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [row](#) = nullptr

- `_Field::Element_ptr` `dat`
- `bool` `delayed` = false
- `uint64_t` `kmax` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `uint64_t` `maxrow` = 0

## 16.293.1 Member Typedef Documentation

### 16.293.1.1 Field

using `Field` = `_Field`

## 16.293.2 Field Documentation

### 16.293.2.1 cst

`_Field::Element` `cst` = 1

### 16.293.2.2 col

`index_t*` `col` = `nullptr` [inherited]

### 16.293.2.3 row

`index_t*` `row` = `nullptr` [inherited]

### 16.293.2.4 dat

`_Field::Element_ptr` `dat` [inherited]

### 16.293.2.5 delayed

`bool` `delayed` = false [inherited]

### 16.293.2.6 kmax

`uint64_t` `kmax` = 0 [inherited]

### 16.293.2.7 m

`index_t` `m` = 0 [inherited]

### 16.293.2.8 n

`index_t` `n` = 0 [inherited]

**16.293.2.9 nnz**

```
uint64_t nnz = 0 [inherited]
```

**16.293.2.10 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.293.2.11 maxrow**

```
uint64_t maxrow = 0 [inherited]
```

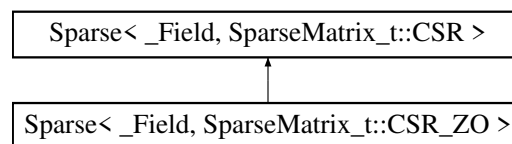
The documentation for this struct was generated from the following file:

- [coo.h](#)

## 16.294 Sparse<\_Field, SparseMatrix\_t::CSR> Struct Template Reference

```
#include <csr.h>
```

Inheritance diagram for Sparse<\_Field, SparseMatrix\_t::CSR>:

**Public Types**

- using [Field](#) = \_Field

**Data Fields**

- bool [delayed](#) = false
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0
- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [st](#) = nullptr
- [index\\_t](#) \* [stend](#) = nullptr
- \_Field::Element\_ptr [dat](#)

**16.294.1 Member Typedef Documentation****16.294.1.1 Field**

```
using Field = _Field
```

## 16.294.2 Field Documentation

### 16.294.2.1 delayed

```
bool delayed = false
```

### 16.294.2.2 kmax

```
uint64_t kmax = 0
```

### 16.294.2.3 m

```
index_t m = 0
```

### 16.294.2.4 n

```
index_t n = 0
```

### 16.294.2.5 nnz

```
uint64_t nnz = 0
```

### 16.294.2.6 nElements

```
uint64_t nElements = 0
```

### 16.294.2.7 maxrow

```
uint64_t maxrow = 0
```

### 16.294.2.8 col

```
index_t* col = nullptr
```

### 16.294.2.9 st

```
index_t* st = nullptr
```

### 16.294.2.10 stend

```
index_t* stend = nullptr
```

### 16.294.2.11 dat

```
_Field::Element_ptr dat
```

The documentation for this struct was generated from the following file:

- [csr.h](#)

## 16.295 Sparse<\_Field, SparseMatrix\_t::CSR\_HYB > Struct Template Reference

```
#include <csr_hyb.h>
```

### Public Types

- using [Field](#) = [\\_Field](#)

### Data Fields

- bool [delayed](#) = false
- [index\\_t](#) \* [col](#) = nullptr
- [index\\_t](#) \* [st](#) = nullptr
- [\\_Field::Element\\_ptr](#) [dat](#)
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0
- [uint64\\_t](#) [nOnes](#) = 0
- [uint64\\_t](#) [nMOnes](#) = 0
- [uint64\\_t](#) [nOthers](#) = 0

## 16.295.1 Member Typedef Documentation

### 16.295.1.1 Field

```
using Field = \_Field
```

## 16.295.2 Field Documentation

### 16.295.2.1 delayed

```
bool delayed = false
```

### 16.295.2.2 col

```
index\_t* col = nullptr
```

### 16.295.2.3 st

```
index\_t* st = nullptr
```

### 16.295.2.4 dat

```
\_Field::Element\_ptr dat
```

**16.295.2.5 kmax**

```
uint64_t kmax = 0
```

**16.295.2.6 m**

```
index_t m = 0
```

**16.295.2.7 n**

```
index_t n = 0
```

**16.295.2.8 nnz**

```
uint64_t nnz = 0
```

**16.295.2.9 nElements**

```
uint64_t nElements = 0
```

**16.295.2.10 maxrow**

```
uint64_t maxrow = 0
```

**16.295.2.11 nOnes**

```
uint64_t nOnes = 0
```

**16.295.2.12 nMOnes**

```
uint64_t nMOnes = 0
```

**16.295.2.13 nOthers**

```
uint64_t nOthers = 0
```

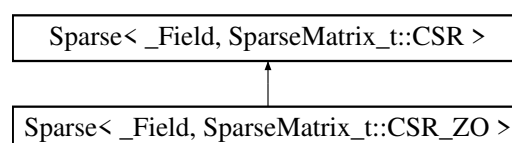
The documentation for this struct was generated from the following file:

- [csr\\_hyb.h](#)

## 16.296 Sparse<\_Field, SparseMatrix\_t::CSR\_ZO > Struct Template Reference

```
#include <csr.h>
```

Inheritance diagram for Sparse<\_Field, SparseMatrix\_t::CSR\_ZO >:



## Public Types

- using `Field` = `_Field`

## Data Fields

- `int64_t` `cst` = 1
- bool `delayed` = false
- `uint64_t` `kmax` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `uint64_t` `maxrow` = 0
- `index_t` \* `col` = nullptr
- `index_t` \* `st` = nullptr
- `index_t` \* `stend` = nullptr
- `_Field::Element_ptr` `dat`

## 16.296.1 Member Typedef Documentation

### 16.296.1.1 Field

using `Field` = `_Field`

## 16.296.2 Field Documentation

### 16.296.2.1 cst

`int64_t` `cst` = 1

### 16.296.2.2 delayed

bool `delayed` = false

### 16.296.2.3 kmax

`uint64_t` `kmax` = 0 [inherited]

### 16.296.2.4 m

`index_t` `m` = 0 [inherited]

### 16.296.2.5 n

`index_t` `n` = 0 [inherited]

### 16.296.2.6 nnz

`uint64_t` `nnz` = 0 [inherited]



**16.296.2.7 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.296.2.8 maxrow**

```
uint64_t maxrow = 0 [inherited]
```

**16.296.2.9 col**

```
index_t* col = nullptr [inherited]
```

**16.296.2.10 st**

```
index_t* st = nullptr [inherited]
```

**16.296.2.11 stend**

```
index_t* stend = nullptr [inherited]
```

**16.296.2.12 dat**

```
_Field::Element_ptr dat [inherited]
```

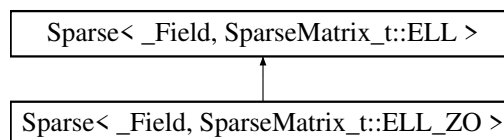
The documentation for this struct was generated from the following file:

- [csr.h](#)

## 16.297 Sparse<\_Field, SparseMatrix\_t::ELL> Struct Template Reference

```
#include <ell.h>
```

Inheritance diagram for Sparse<\_Field, SparseMatrix\_t::ELL>:

**Public Types**

- using [Field](#) = \_Field

**Data Fields**

- bool [delayed](#) = false
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [index\\_t](#) [ld](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0

- `uint64_t maxrow = 0`
- `index_t * col = nullptr`
- `_Field::Element_ptr dat`

## 16.297.1 Member Typedef Documentation

### 16.297.1.1 Field

```
using Field = _Field
```

## 16.297.2 Field Documentation

### 16.297.2.1 delayed

```
bool delayed = false
```

### 16.297.2.2 kmax

```
uint64_t kmax = 0
```

### 16.297.2.3 m

```
index_t m = 0
```

### 16.297.2.4 n

```
index_t n = 0
```

### 16.297.2.5 ld

```
index_t ld = 0
```

### 16.297.2.6 nnz

```
uint64_t nnz = 0
```

### 16.297.2.7 nElements

```
uint64_t nElements = 0
```

### 16.297.2.8 maxrow

```
uint64_t maxrow = 0
```

### 16.297.2.9 col

```
index_t* col = nullptr
```

**16.297.2.10 dat**

```
_Field::Element_ptr dat
```

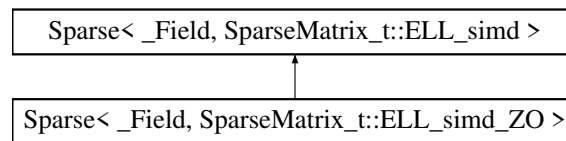
The documentation for this struct was generated from the following file:

- [ell.h](#)

**16.298 Sparse< \_Field, SparseMatrix\_t::ELL\_simd > Struct Template Reference**

```
#include <ell_simd.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::ELL\_simd >:

**Data Fields**

- bool [delayed](#) = false
- int [chunk](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [index\\_t](#) [ld](#) = 0
- [uint64\\_t](#) [kmax](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0
- [uint64\\_t](#) [nChunks](#) = 0
- [index\\_t](#) \* [col](#) = nullptr
- \_Field::Element\_ptr [dat](#)

**16.298.1 Field Documentation****16.298.1.1 delayed**

```
bool delayed = false
```

**16.298.1.2 chunk**

```
int chunk = 0
```

**16.298.1.3 m**

```
index_t m = 0
```

**16.298.1.4 n**

```
index_t n = 0
```

**16.298.1.5 ld**

```
index_t ld = 0
```

**16.298.1.6 kmax**

```
uint64_t kmax = 0
```

**16.298.1.7 nnz**

```
uint64_t nnz = 0
```

**16.298.1.8 nElements**

```
uint64_t nElements = 0
```

**16.298.1.9 maxrow**

```
uint64_t maxrow = 0
```

**16.298.1.10 nChunks**

```
uint64_t nChunks = 0
```

**16.298.1.11 col**

```
index_t* col = nullptr
```

**16.298.1.12 dat**

```
_Field::Element_ptr dat
```

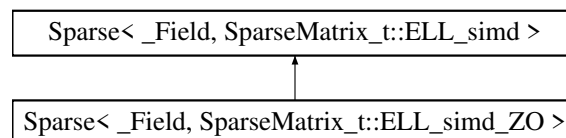
The documentation for this struct was generated from the following file:

- [ell\\_simd.h](#)

## 16.299 Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO > Struct Template Reference

```
#include <ell_simd.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >:



## Data Fields

- `_Field::Element` `cst` = 1
- `bool` `delayed` = false
- `int` `chunk` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `index_t` `ld` = 0
- `uint64_t` `kmax` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `uint64_t` `maxrow` = 0
- `uint64_t` `nChunks` = 0
- `index_t` \* `col` = nullptr
- `_Field::Element_ptr` `dat`

### 16.299.1 Field Documentation

#### 16.299.1.1 `cst`

```
_Field::Element cst = 1
```

#### 16.299.1.2 `delayed`

```
bool delayed = false [inherited]
```

#### 16.299.1.3 `chunk`

```
int chunk = 0 [inherited]
```

#### 16.299.1.4 `m`

```
index_t m = 0 [inherited]
```

#### 16.299.1.5 `n`

```
index_t n = 0 [inherited]
```

#### 16.299.1.6 `ld`

```
index_t ld = 0 [inherited]
```

#### 16.299.1.7 `kmax`

```
uint64_t kmax = 0 [inherited]
```

#### 16.299.1.8 `nnz`

```
uint64_t nnz = 0 [inherited]
```

**16.299.1.9 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.299.1.10 maxrow**

```
uint64_t maxrow = 0 [inherited]
```

**16.299.1.11 nChunks**

```
uint64_t nChunks = 0 [inherited]
```

**16.299.1.12 col**

```
index_t* col = nullptr [inherited]
```

**16.299.1.13 dat**

```
_Field::Element_ptr dat [inherited]
```

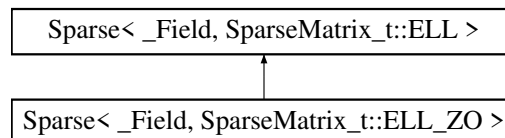
The documentation for this struct was generated from the following file:

- [ell\\_simd.h](#)

## 16.300 Sparse< \_Field, SparseMatrix\_t::ELL\_ZO > Struct Template Reference

```
#include <ell.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >:

**Public Types**

- using [Field](#) = \_Field

**Data Fields**

- \_Field::Element [cst](#) = 1
- bool [delayed](#) = false
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [index\\_t](#) [ld](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0
- [index\\_t](#) \* [col](#) = nullptr
- \_Field::Element\_ptr [dat](#)

## 16.300.1 Member Typedef Documentation

### 16.300.1.1 Field

using `Field` = `_Field`

## 16.300.2 Field Documentation

### 16.300.2.1 cst

`_Field::Element` `cst` = 1

### 16.300.2.2 delayed

`bool` `delayed` = `false` [inherited]

### 16.300.2.3 kmax

`uint64_t` `kmax` = 0 [inherited]

### 16.300.2.4 m

`index_t` `m` = 0 [inherited]

### 16.300.2.5 n

`index_t` `n` = 0 [inherited]

### 16.300.2.6 ld

`index_t` `ld` = 0 [inherited]

### 16.300.2.7 nnz

`uint64_t` `nnz` = 0 [inherited]

### 16.300.2.8 nElements

`uint64_t` `nElements` = 0 [inherited]

### 16.300.2.9 maxrow

`uint64_t` `maxrow` = 0 [inherited]

### 16.300.2.10 col

`index_t*` `col` = `nullptr` [inherited]

**16.300.2.11 dat**

```
_Field::Element_ptr dat [inherited]
```

The documentation for this struct was generated from the following file:

- [ell.h](#)

**16.301 Sparse<\_Field, SparseMatrix\_t::HYB\_ZO > Struct Template Reference**

```
#include <hyb_zo.h>
```

**Public Types**

- using [Field](#) = [\\_Field](#)
- typedef [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::HYB\\_ZO](#) > [Self\\_t](#)

**Data Fields**

- bool [delayed](#) = false
- [uint64\\_t](#) [kmax](#) = 0
- [index\\_t](#) [m](#) = 0
- [index\\_t](#) [n](#) = 0
- [uint64\\_t](#) [nnz](#) = 0
- [uint64\\_t](#) [maxrow](#) = 0
- [uint64\\_t](#) [nElements](#) = 0
- [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::CSR](#) > \* [dat](#) = nullptr
- [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > \* [one](#) = nullptr
- [Sparse](#)< [\\_Field](#), [SparseMatrix\\_t::CSR\\_ZO](#) > \* [mone](#) = nullptr

**16.301.1 Member Typedef Documentation****16.301.1.1 Field**

```
using Field = \_Field
```

**16.301.1.2 Self\_t**

```
typedef Sparse< \_Field, SparseMatrix\_t::HYB\_ZO > Self\_t
```

**16.301.2 Field Documentation****16.301.2.1 delayed**

```
bool delayed = false
```

**16.301.2.2 kmax**

```
uint64\_t kmax = 0
```



**16.301.2.3 m**

```
index_t m = 0
```

**16.301.2.4 n**

```
index_t n = 0
```

**16.301.2.5 nnz**

```
uint64_t nnz = 0
```

**16.301.2.6 maxrow**

```
uint64_t maxrow = 0
```

**16.301.2.7 nElements**

```
uint64_t nElements = 0
```

**16.301.2.8 dat**

```
Sparse<_Field, SparseMatrix_t::CSR>* dat = nullptr
```

**16.301.2.9 one**

```
Sparse<_Field, SparseMatrix_t::CSR_ZO>* one = nullptr
```

**16.301.2.10 mone**

```
Sparse<_Field, SparseMatrix_t::CSR_ZO>* mone = nullptr
```

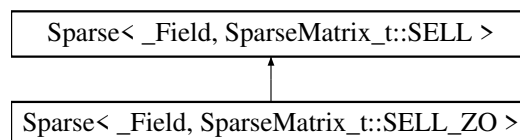
The documentation for this struct was generated from the following file:

- [hyb\\_zo.h](#)

## 16.302 Sparse< \_Field, SparseMatrix\_t::SELL > Struct Template Reference

```
#include <sell.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::SELL >:



### Public Types

- using [Field](#) = \_Field

## Data Fields

- bool `delayed` = false
- int `chunk` = 0
- `index_t` `kmax` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `index_t` `maxrow` = 0
- `index_t` `sigma` = 0
- `index_t` `nChunks` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `index_t` \* `perm` = nullptr
- `uint64_t` \* `st` = nullptr
- `index_t` \* `chunkSize` = nullptr
- `index_t` \* `col` = nullptr
- `_Field::Element_ptr` `dat`

### 16.302.1 Member Typedef Documentation

#### 16.302.1.1 Field

using `Field` = `_Field`

### 16.302.2 Field Documentation

#### 16.302.2.1 delayed

bool `delayed` = false

#### 16.302.2.2 chunk

int `chunk` = 0

#### 16.302.2.3 kmax

`index_t` `kmax` = 0

#### 16.302.2.4 m

`index_t` `m` = 0

#### 16.302.2.5 n

`index_t` `n` = 0

#### 16.302.2.6 maxrow

`index_t` `maxrow` = 0

**16.302.2.7 sigma**

```
index_t sigma = 0
```

**16.302.2.8 nChunks**

```
index_t nChunks = 0
```

**16.302.2.9 nnz**

```
uint64_t nnz = 0
```

**16.302.2.10 nElements**

```
uint64_t nElements = 0
```

**16.302.2.11 perm**

```
index_t* perm = nullptr
```

**16.302.2.12 st**

```
uint64_t* st = nullptr
```

**16.302.2.13 chunkSize**

```
index_t* chunkSize = nullptr
```

**16.302.2.14 col**

```
index_t* col = nullptr
```

**16.302.2.15 dat**

```
_Field::Element_ptr dat
```

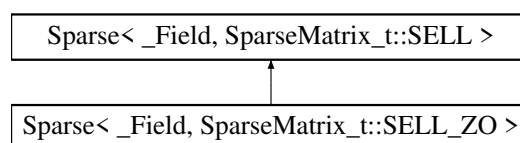
The documentation for this struct was generated from the following file:

- [sell.h](#)

**16.303 Sparse< \_Field, SparseMatrix\_t::SELL\_ZO > Struct Template Reference**

```
#include <sell.h>
```

Inheritance diagram for Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >:



## Public Types

- using `Field` = `_Field`

## Data Fields

- `_Field::Element` `cst` = 1
- bool `delayed` = false
- int `chunk` = 0
- `index_t` `kmax` = 0
- `index_t` `m` = 0
- `index_t` `n` = 0
- `index_t` `maxrow` = 0
- `index_t` `sigma` = 0
- `index_t` `nChunks` = 0
- `uint64_t` `nnz` = 0
- `uint64_t` `nElements` = 0
- `index_t` \* `perm` = nullptr
- `uint64_t` \* `st` = nullptr
- `index_t` \* `chunkSize` = nullptr
- `index_t` \* `col` = nullptr
- `_Field::Element_ptr` `dat`

## 16.303.1 Member Typedef Documentation

### 16.303.1.1 Field

using `Field` = `_Field`

## 16.303.2 Field Documentation

### 16.303.2.1 cst

`_Field::Element` `cst` = 1

### 16.303.2.2 delayed

bool `delayed` = false [inherited]

### 16.303.2.3 chunk

int `chunk` = 0 [inherited]

### 16.303.2.4 kmax

`index_t` `kmax` = 0 [inherited]

### 16.303.2.5 m

`index_t` `m` = 0 [inherited]

**16.303.2.6 n**

```
index_t n = 0 [inherited]
```

**16.303.2.7 maxrow**

```
index_t maxrow = 0 [inherited]
```

**16.303.2.8 sigma**

```
index_t sigma = 0 [inherited]
```

**16.303.2.9 nChunks**

```
index_t nChunks = 0 [inherited]
```

**16.303.2.10 nnz**

```
uint64_t nnz = 0 [inherited]
```

**16.303.2.11 nElements**

```
uint64_t nElements = 0 [inherited]
```

**16.303.2.12 perm**

```
index_t* perm = nullptr [inherited]
```

**16.303.2.13 st**

```
uint64_t* st = nullptr [inherited]
```

**16.303.2.14 chunkSize**

```
index_t* chunkSize = nullptr [inherited]
```

**16.303.2.15 col**

```
index_t* col = nullptr [inherited]
```

**16.303.2.16 dat**

```
_Field::Element_ptr dat [inherited]
```

The documentation for this struct was generated from the following file:

- [sell.h](#)

**16.304 SpMat< Field, flag > Struct Template Reference**

```
#include <fflas_sparse.h>
```

## Data Fields

- [FFLAS::CooMat< Field > \\* \\_coo](#) = nullptr
- [FFLAS::CsrMat< Field > \\* \\_csr](#) = nullptr
- [FFLAS::EllMat< Field > \\* \\_ell](#) = nullptr

### 16.304.1 Field Documentation

#### 16.304.1.1 \_coo

```
FFLAS::CooMat<Field>* _coo = nullptr
```

#### 16.304.1.2 \_csr

```
FFLAS::CsrMat<Field>* _csr = nullptr
```

#### 16.304.1.3 \_ell

```
FFLAS::EllMat<Field>* _ell = nullptr
```

The documentation for this struct was generated from the following file:

- [fflas\\_sparse.h](#)

## 16.305 Static\_error\_check< bool > Class Template Reference

```
#include <instrset.h>
```

### Public Member Functions

- [Static\\_error\\_check\(\)](#)

### 16.305.1 Constructor & Destructor Documentation

#### 16.305.1.1 Static\_error\_check()

```
Static_error_check ( ) [inline]
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

## 16.306 Static\_error\_check< false > Class Reference

```
#include <instrset.h>
```

The documentation for this class was generated from the following file:

- [instrset.h](#)

## 16.307 StatsMatrix Struct Reference

```
#include <utils.h>
```

## Data Fields

- `uint64_t rowdim` = 0
- `uint64_t coldim` = 0
- `uint64_t nOnes` = 0
- `uint64_t nMOnes` = 0
- `uint64_t nOthers` = 0
- `uint64_t nnz` = 0
- `uint64_t maxRow` = 0
- `uint64_t minRow` = 0
- `uint64_t averageRow` = 0
- `uint64_t deviationRow` = 0
- `uint64_t maxCol` = 0
- `uint64_t minCol` = 0
- `uint64_t averageCol` = 0
- `uint64_t deviationCol` = 0
- `uint64_t minColDifference` = 0
- `uint64_t maxColDifference` = 0
- `uint64_t averageColDifference` = 0
- `uint64_t deviationColDifference` = 0
- `uint64_t minRowDifference` = 0
- `uint64_t maxRowDifference` = 0
- `uint64_t averageRowDifference` = 0
- `uint64_t deviationRowDifference` = 0
- `uint64_t nDenseRows` = 0
- `uint64_t nDenseCols` = 0
- `uint64_t nEmptyRows` = 0
- `uint64_t nEmptyCols` = 0
- `uint64_t nEmptyColsEnd` = 0
- `std::vector< uint64_t > denseRows`
- `std::vector< uint64_t > denseCols`

### 16.307.1 Field Documentation

#### 16.307.1.1 rowdim

`uint64_t rowdim` = 0

#### 16.307.1.2 coldim

`uint64_t coldim` = 0

#### 16.307.1.3 nOnes

`uint64_t nOnes` = 0

#### 16.307.1.4 nMOnes

`uint64_t nMOnes` = 0

**16.307.1.5 nOthers**

```
uint64_t nOthers = 0
```

**16.307.1.6 nnz**

```
uint64_t nnz = 0
```

**16.307.1.7 maxRow**

```
uint64_t maxRow = 0
```

**16.307.1.8 minRow**

```
uint64_t minRow = 0
```

**16.307.1.9 averageRow**

```
uint64_t averageRow = 0
```

**16.307.1.10 deviationRow**

```
uint64_t deviationRow = 0
```

**16.307.1.11 maxCol**

```
uint64_t maxCol = 0
```

**16.307.1.12 minCol**

```
uint64_t minCol = 0
```

**16.307.1.13 averageCol**

```
uint64_t averageCol = 0
```

**16.307.1.14 deviationCol**

```
uint64_t deviationCol = 0
```

**16.307.1.15 minColDifference**

```
uint64_t minColDifference = 0
```

**16.307.1.16 maxColDifference**

```
uint64_t maxColDifference = 0
```



**16.307.1.17 averageColDifference**

```
uint64_t averageColDifference = 0
```

**16.307.1.18 deviationColDifference**

```
uint64_t deviationColDifference = 0
```

**16.307.1.19 minRowDifference**

```
uint64_t minRowDifference = 0
```

**16.307.1.20 maxRowDifference**

```
uint64_t maxRowDifference = 0
```

**16.307.1.21 averageRowDifference**

```
uint64_t averageRowDifference = 0
```

**16.307.1.22 deviationRowDifference**

```
uint64_t deviationRowDifference = 0
```

**16.307.1.23 nDenseRows**

```
uint64_t nDenseRows = 0
```

**16.307.1.24 nDenseCols**

```
uint64_t nDenseCols = 0
```

**16.307.1.25 nEmptyRows**

```
uint64_t nEmptyRows = 0
```

**16.307.1.26 nEmptyCols**

```
uint64_t nEmptyCols = 0
```

**16.307.1.27 nEmptyColsEnd**

```
uint64_t nEmptyColsEnd = 0
```

**16.307.1.28 denseRows**

```
std::vector<uint64_t> denseRows
```

### 16.307.1.29 denseCols

```
std::vector<uint64_t> denseCols
```

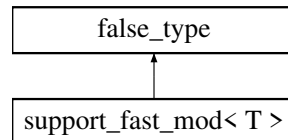
The documentation for this struct was generated from the following file:

- [utils.h](#)

## 16.308 support\_fast\_mod< T > Struct Template Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< T >:



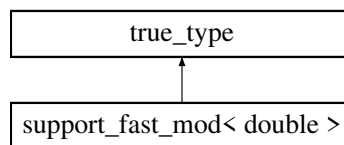
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

## 16.309 support\_fast\_mod< double > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< double >:



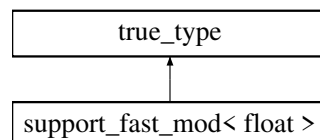
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

## 16.310 support\_fast\_mod< float > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< float >:



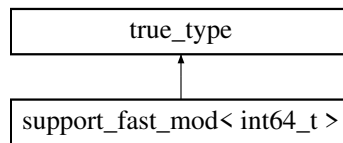
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

## 16.311 support\_fast\_mod< int64\_t > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_fast\_mod< int64\_t >:



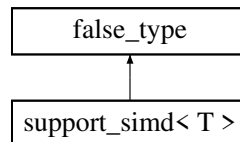
The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

## 16.312 support\_simd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

Inheritance diagram for support\_simd< T >:



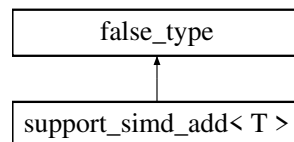
The documentation for this struct was generated from the following file:

- [fflas\\_simd.h](#)

## 16.313 support\_simd\_add< T > Struct Template Reference

```
#include <fflas_fadd.h>
```

Inheritance diagram for support\_simd\_add< T >:



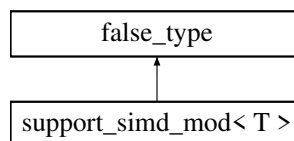
The documentation for this struct was generated from the following file:

- [fflas\\_fadd.h](#)

## 16.314 support\_simd\_mod< T > Struct Template Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support\_simd\_mod< T >:



The documentation for this struct was generated from the following file:

- [fflas\\_freduce.h](#)

## 16.315 tfn\_minus Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

### 16.315.1 Member Function Documentation

#### 16.315.1.1 `operator>()()`

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.316 tfn\_minus\_eq Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

### 16.316.1 Member Function Documentation

#### 16.316.1.1 `operator>()()`

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

## 16.317 tfn\_mul Struct Reference

```
#include <sparse_matrix_traits.h>
```

### Public Member Functions

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

### 16.317.1 Member Function Documentation

**16.317.1.1 operator>()()**

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

**16.318 tfn\_mul\_eq Struct Reference**

```
#include <sparse_matrix_traits.h>
```

**Public Member Functions**

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

**16.318.1 Member Function Documentation****16.318.1.1 operator>()()**

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

**16.319 tfn\_plus Struct Reference**

```
#include <sparse_matrix_traits.h>
```

**Public Member Functions**

- `template<typename... Args>`  
`auto operator\(\) (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

**16.319.1 Member Function Documentation****16.319.1.1 operator>()()**

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

**16.320 tfn\_plus\_eq Struct Reference**

```
#include <sparse_matrix_traits.h>
```

## Public Member Functions

- `template<typename... Args>`  
`auto operator() (Args &&... args) const -> decltype(operator+(std::forward< Args >(args)...))`

### 16.320.1 Member Function Documentation

#### 16.320.1.1 operator>()()

```
auto operator() (
    Args &&... args ) const -> decltype(operator+(std::forward<Args>(args)...))
[inline]
```

The documentation for this struct was generated from the following file:

- [sparse\\_matrix\\_traits.h](#)

### 16.321 Threads Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

### 16.322 ThreeD Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

### 16.323 ThreeDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

### 16.324 ThreeDInPlace Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

### 16.325 TRSMHelper< ReclterTrait, ParSeqTrait > Struct Template Reference

TRSM Helper.

## Public Member Functions

- `template<class Cut , class Param >`  
`TRSMHelper (ParSeqHelper::Parallel< Cut, Param > _PS)`
- `TRSMHelper (ParSeqHelper::Sequential _PS)`
- `template<typename RIT , typename PST >`  
`TRSMHelper (TRSMHelper< RIT, PST > &_TH)`
- `template<class Dom , class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS::ModeTraits<Dom>::value>`  
`FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (Dom &D, size_t m, size_t k, size_t n, ParSeqTrait p) const`

- `template<class Dom , class Algo = FFLAS::MMHelperAlgo::Winograd, class ModeT = typename FFLAS::ModeTraits<Dom>::value> FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (Dom &D, size_t m, size_t k, size_t n) const`

## Data Fields

- ParSeqTrait [parseq](#)

### 16.325.1 Detailed Description

```
template<typename RectrTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
struct FFLAS::TRSMHelper< RectrTrait, ParSeqTrait >
```

TRSM Helper.

### 16.325.2 Constructor & Destructor Documentation

#### 16.325.2.1 TRSMHelper() [1/3]

```
TRSMHelper (
    ParSeqHelper::Parallel< Cut, Param > _PS ) [inline]
```

#### 16.325.2.2 TRSMHelper() [2/3]

```
TRSMHelper (
    ParSeqHelper::Sequential _PS ) [inline]
```

#### 16.325.2.3 TRSMHelper() [3/3]

```
TRSMHelper (
    TRSMHelper< RIT, PST > & _TH ) [inline]
```

### 16.325.3 Member Function Documentation

#### 16.325.3.1 pMMH() [1/2]

```
FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (
    Dom & D,
    size_t m,
    size_t k,
    size_t n,
    ParSeqTrait p ) const [inline]
```

#### 16.325.3.2 pMMH() [2/2]

```
FFLAS::MMHelper< Dom, Algo, ModeT, ParSeqTrait > pMMH (
    Dom & D,
    size_t m,
    size_t k,
    size_t n ) const [inline]
```

### 16.325.4 Field Documentation

#### 16.325.4.1 parseq

`ParSeqTrait parseq`

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

### 16.326 TwoD Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

### 16.327 TwoDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- [blockcuts.inl](#)

### 16.328 UnparametricTag Struct Reference

If the field uses a representation with infix operators.

```
#include <field-traits.h>
```

#### 16.328.1 Detailed Description

If the field uses a representation with infix operators.

The documentation for this struct was generated from the following file:

- [field-traits.h](#)

### 16.329 Winograd Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)

### 16.330 WinogradPar Struct Reference

The documentation for this struct was generated from the following file:

- [fflas\\_helpers.inl](#)



# Chapter 17

## File Documentation

### 17.1 arithprog.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

#### Macros

- `#define CUBE(x) ((x)*(x)*(x))`
- `#define GFOPS(m, n, r, t) (2.7*CUBE(double(n)/1000.0))/t`

#### Typedefs

- `typedef Givaro::Timer TTimer`

#### Functions

- `int main (int argc, char **argv)`

#### 17.1.1 Macro Definition Documentation

##### 17.1.1.1 CUBE

```
#define CUBE(  
    x )  ( (x) * (x) * (x) )
```

##### 17.1.1.2 GFOPS

```
#define GFOPS(  
    m,  
    n,  
    r,  
    t )  ( 2.7*CUBE(double(n)/1000.0) ) / t
```

## 17.1.2 Typedef Documentation

### 17.1.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.1.3 Function Documentation

### 17.1.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.2 charpoly.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <ctime>
```

### Macros

- #define CUBE(x) ((x)\*(x)\*(x))
- #define GFOPS(m, n, r, t) (2.7\*CUBE(double(n)/1000.0))/t

### Typedefs

- typedef Givaro::Timer TTimer

### Functions

- int main ()

## 17.2.1 Macro Definition Documentation

### 17.2.1.1 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

### 17.2.1.2 GFOPS

```
#define GFOPS(
    m,
    n,
    r,
    t ) (2.7*CUBE(double(n)/1000.0))/t
```

## 17.2.2 Typedef Documentation

### 17.2.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.2.3 Function Documentation

### 17.2.3.1 main()

```
int main (
    void )
```

## 17.3 charpoly.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

### Functions

- int [main](#)(int argc, char \*\*argv)

*This example computes the characteristic polynomial of a matrix over a defined finite field.*

### 17.3.1 Function Documentation

#### 17.3.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the characteristic polynomial of a matrix over a defined finite field. Outputs the characteristic polynomial.

## 17.4 fsyrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utis/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utis/fflas_randommatrix.h"
#include <ctime>
```

### Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))
- #define [GFOPS](#)(n, t) ([CUBE](#)(double(n)/1000.0)/(3.0\*t))

## Typedefs

- typedef Givaro::Timer [TTimer](#)

## Functions

- int [main](#) ()

### 17.4.1 Macro Definition Documentation

#### 17.4.1.1 CUBE

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

#### 17.4.1.2 GFOPS

```
#define GFOPS(  
    n,  
    t )  (CUBE(double(n)/1000.0)/(3.0*t))
```

### 17.4.2 Typedef Documentation

#### 17.4.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

### 17.4.3 Function Documentation

#### 17.4.3.1 main()

```
int main (  
    void )
```

## 17.5 fsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular-balanced.h>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include "fflas-ffpack/utils/fflas_randommatrix.h"  
#include <ctime>
```

## Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))
- #define [GFOPS](#)(n, t) ([CUBE](#)(double(n)/1000.0)/(3.0\*t))

## Typedefs

- typedef Givaro::Timer [TTimer](#)

## Functions

- int [main](#) ()

## 17.5.1 Macro Definition Documentation

### 17.5.1.1 CUBE

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

### 17.5.1.2 GFOPS

```
#define GFOPS(  
    n,  
    t )  (CUBE(double(n)/1000.0)/(3.0*t))
```

## 17.5.2 Typedef Documentation

### 17.5.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.5.3 Function Documentation

### 17.5.3.1 main()

```
int main (  
    void )
```

## 17.6 ftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include "fflas-ffpack/utils/fflas_randommatrix.h"  
#include <iostream>  
#include <givaro/modular-balanced.h>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include <ctime>
```

## Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))
- #define [GFOPS](#)(n, t) ([CUBE](#)(double(n)/1000.0)/(3.0\*t))

## Typedefs

- typedef Givaro::Timer [TTimer](#)

## Functions

- int [main](#) ()

### 17.6.1 Macro Definition Documentation

#### 17.6.1.1 CUBE

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

#### 17.6.1.2 GFOPS

```
#define GFOPS(  
    n,  
    t )  (CUBE(double(n)/1000.0)/(3.0*t))
```

### 17.6.2 Typedef Documentation

#### 17.6.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

### 17.6.3 Function Documentation

#### 17.6.3.1 main()

```
int main (  
    void )
```

## 17.7 pluq.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include "fflas-ffpack/utils/fflas_randommatrix.h"  
#include <iostream>  
#include <givaro/modular-balanced.h>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include <ctime>
```

## Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))
- #define [GFOPS](#)(m, n, r, t) (2.0/3.0\*[CUBE](#)(double(n)/1000.0) +2\*m/1000.0\*n/1000.0\*double(r)/1000.0 - double(r)/1000.0\*double(r)/1000.0\*(m+n)/1000)/t

## Typedefs

- typedef Givaro::Timer [TTimer](#)

## Functions

- int [main](#) ()

## 17.7.1 Macro Definition Documentation

### 17.7.1.1 CUBE

```
#define CUBE(  
    x )  ((x)*(x)*(x))
```

### 17.7.1.2 GFOPS

```
#define GFOPS(  
    m,  
    n,  
    r,  
    t )  (2.0/3.0*CUBE(double(n)/1000.0) +2*m/1000.0*n/1000.0*double(r)/1000.0 - double(r)/1000.0←  
0*double(r)/1000.0*(m+n)/1000)/t
```

## 17.7.2 Typedef Documentation

### 17.7.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.7.3 Function Documentation

### 17.7.3.1 main()

```
int main (  
    void )
```

## 17.8 pluq.C File Reference

```
#include <iostream>  
#include <givaro/modular.h>  
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include "fflas-ffpack/fflas-ffpack.h"  
#include "fflas-ffpack/utils/fflas_io.h"
```

## Functions

- int [main](#) (int argc, char \*\*argv)

## 17.8.1 Function Documentation

### 17.8.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.9 winograd.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
#include <fstream>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include <ctime>
```

### Macros

- `#define DOUBLE_TO_FLOAT_CROSSOVER 0`
- `#define GFOPS(n, t) (2.0/t*(double)n/1000.0*(double)n/1000.0*(double)n/1000.0)`

### Typedefs

- `typedef Givaro::Timer TTimer`

### Functions

- `template<class Field >`  
`bool balanced (const Field &)`
- `template<class T >`  
`bool balanced (const Givaro::ModularBalanced< T > &)`
- `int main ()`

## 17.9.1 Macro Definition Documentation

### 17.9.1.1 DOUBLE\_TO\_FLOAT\_CROSSOVER

```
#define DOUBLE_TO_FLOAT_CROSSOVER 0
```

### 17.9.1.2 GFOPS

```
#define GFOPS(
    n,
    t ) (2.0/t*(double)n/1000.0*(double)n/1000.0*(double)n/1000.0)
```

## 17.9.2 Typedef Documentation



### 17.9.2.1 TTimer

```
typedef Givaro::Timer TTimer
```

## 17.9.3 Function Documentation

### 17.9.3.1 balanced() [1/2]

```
bool balanced (
    const Field & )
```

### 17.9.3.2 balanced() [2/2]

```
bool balanced (
    const Givaro::ModularBalanced< T > & )
```

### 17.9.3.3 main()

```
int main (
    void )
```

## 17.10 benchmark-charpoly-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/Matio.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define `__FFLASFFPACK_FORCE_SEQ`

### Functions

- int `main` (int argc, char \*\*argv)

## 17.10.1 Macro Definition Documentation

### 17.10.1.1 \_\_FFLASFFPACK\_FORCE\_SEQ

```
#define __FFLASFFPACK_FORCE_SEQ
```

## 17.10.2 Function Documentation

### 17.10.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.11 benchmark-charpoly.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include <givaro/givpoly1.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

### Functions

- `template<class Field >`  
`void run_with_field (int q, size_t bits, size_t n, size_t d, size_t iter, std::string file, int variant)`
- `int main (int argc, char **argv)`

### 17.11.1 Macro Definition Documentation

#### 17.11.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.11.2 Function Documentation

#### 17.11.2.1 run\_with\_field()

```
void run_with_field (
    int q,
    size_t bits,
    size_t n,
    size_t d,
    size_t iter,
    std::string file,
    int variant )
```

#### 17.11.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.12 benchmark-checkers.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/checkers/checkers_fflas.h"
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include <fstream>
```

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1
- #define [\\_NR\\_TESTS](#) 5
- #define [\\_MAX\\_SIZE\\_MATRICES](#) 1000
- #define [CUBE](#)(x) ((x)\*(x)\*(x))

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.12.1 Macro Definition Documentation

### 17.12.1.1 [ENABLE\\_ALL\\_CHECKINGS](#)

```
#define ENABLE_ALL_CHECKINGS 1
```

### 17.12.1.2 [\\_NR\\_TESTS](#)

```
#define _NR_TESTS 5
```

### 17.12.1.3 [\\_MAX\\_SIZE\\_MATRICES](#)

```
#define _MAX_SIZE_MATRICES 1000
```

### 17.12.1.4 [CUBE](#)

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```

## 17.12.2 Function Documentation

### 17.12.2.1 [main\(\)](#)

```
int main (  
    int argc,  
    char ** argv )
```

## 17.13 benchmark-dgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define [CBLAS\\_GEMM](#) `cblas_dgemm`

### Typedefs

- typedef [FFLAS::Timer](#) `TTimer`
- typedef double [Floats](#)

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.13.1 Macro Definition Documentation

### 17.13.1.1 CBLAS\_GEMM

```
#define CBLAS_GEMM cblas_dgemm
```

## 17.13.2 Typedef Documentation

### 17.13.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

### 17.13.2.2 Floats

```
typedef double Floats
```

## 17.13.3 Function Documentation

### 17.13.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.14 benchmark-dgetrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define __FFLASFFPACK_HAVE_DGETRF 1`

### Typedefs

- `typedef FFLAS::Timer TTimer`

### Functions

- `int main (int argc, char **argv)`

## 17.14.1 Macro Definition Documentation

### 17.14.1.1 \_\_FFLASFFPACK\_HAVE\_DGETRF

```
#define __FFLASFFPACK_HAVE_DGETRF 1
```

## 17.14.2 Typedef Documentation

### 17.14.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

## 17.14.3 Function Documentation

### 17.14.3.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.15 benchmark-dgetri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
```

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Typedefs

- typedef [FFLAS::Timer](#) [TTimer](#)

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.15.1 Typedef Documentation

#### 17.15.1.1 TTimer

```
typedef FFLAS::Timer TTimer
```

### 17.15.2 Function Documentation

#### 17.15.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.16 benchmark-dsytrf.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- #define [EFFGFF](#)(n, t, i) ( (double(n)/1000.\*double(n)/1000.\*double(n)/1000.0) / double(t) \* double(i) / 3.)

## Typedefs

- typedef [FFLAS::Timer](#) [TTimer](#)

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.16.1 Macro Definition Documentation

### 17.16.1.1 EFGFF

```
#define EFGFF(  
    n,  
    t,  
    i ) ( (double(n)/1000.*double(n)/1000.*double(n)/1000.0) / double(t) * double(i)  
/ 3.)
```

## 17.16.2 Typedef Documentation

### 17.16.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

## 17.16.3 Function Documentation

### 17.16.3.1 main()

```
int main (  
    int argc,  
    char ** argv )
```

## 17.17 benchmark-dtrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular.h>  
#include "fflas-ffpack/fflas-ffpack.h"  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/utils/fflas_io.h"  
#include "fflas-ffpack/utils/args-parser.h"
```

## Typedefs

- typedef [FFLAS::Timer](#) TTimer

## Functions

- int [main](#) (int argc, char \*\*argv)

## 17.17.1 Typedef Documentation

### 17.17.1.1 TTimer

```
typedef FFLAS::Timer TTimer
```

## 17.17.2 Function Documentation

### 17.17.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.18 benchmark-dtrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define __FFLASFFPACK_HAVE_DTRTRI 1`

### Typedefs

- `typedef FFLAS::Timer TTimer`

### Functions

- `int main (int argc, char **argv)`

## 17.18.1 Macro Definition Documentation

### 17.18.1.1 \_\_FFLASFFPACK\_HAVE\_DTRTRI

```
#define __FFLASFFPACK_HAVE_DTRTRI 1
```

## 17.18.2 Typedef Documentation

### 17.18.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

## 17.18.3 Function Documentation

### 17.18.3.1 main()

```
int main (
    int argc,
    char ** argv )
```



## 17.19 benchmark-fadd-lvl2.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

### Functions

- `int main (int argc, char **argv)`

#### 17.19.1 Macro Definition Documentation

##### 17.19.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

#### 17.19.2 Function Documentation

##### 17.19.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.20 benchmark-fdot.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include <givaro/givrational.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- template<class [Field](#) >  
[Field::Element run\\_with\\_field](#) (int q, size\_t iter, size\_t N, const size\_t BS, const size\_t p, const size\_t threads)
- int [main](#) (int argc, char \*\*argv)

## 17.20.1 Macro Definition Documentation

### 17.20.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.20.2 Function Documentation

### 17.20.2.1 run\_with\_field()

```
Field::Element run_with_field (
    int q,
    size_t iter,
    size_t N,
    const size_t BS,
    const size_t p,
    const size_t threads )
```

### 17.20.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.21 benchmark-fgemm-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <typeinfo>
#include <vector>
#include <string>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
#include "fflas-ffpack/paladin/parallel.h"
```

## Macros

- #define [\\_\\_FFLASFFPACK\\_OPENBLAS\\_NT\\_ALREADY\\_SET](#) 1
- #define [MG\\_DEFAULT](#) MG\_ACTIVE
- #define [STD\\_RECINT\\_SIZE](#) 8

## Functions

- `template<typename Ints >`  
`int tmain ()`
- `int main (int argc, char **argv)`

### 17.21.1 Macro Definition Documentation

#### 17.21.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

#### 17.21.1.2 MG\_DEFAULT

```
#define MG_DEFAULT MG_ACTIVE
```

#### 17.21.1.3 STD\_RECINT\_SIZE

```
#define STD_RECINT_SIZE 8
```

### 17.21.2 Function Documentation

#### 17.21.2.1 tmain()

```
int tmain ( )
```

#### 17.21.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.22 benchmark-fgemm-rns.C File Reference

```
#include "fflas-ffpack/fflas/fflas.h"
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Typedefs

- `typedef FFPACK::rns_double RNS`
- `typedef FFPACK::RNSInteger< RNS > Field`
- `typedef Field::Element_ptr Element_ptr`
- `typedef Field::ConstElement_ptr ConstElement_ptr`
- `typedef StrategyParameter::Threads THREADS`

- typedef [StrategyParameter::Grain](#) GRAIN
- typedef [StrategyParameter::TwoD](#) TWOD
- typedef [StrategyParameter::TwoDAdaptive](#) TWODA
- typedef [StrategyParameter::ThreeD](#) THREED
- typedef [StrategyParameter::ThreeDAdaptive](#) THREEDA
- typedef [StrategyParameter::ThreeDInPlace](#) THREEDIP
- typedef [ParSeqHelper::Sequential](#) PSeq

## Functions

- int [main](#) (int argc, char \*argv[])

## 17.22.1 Macro Definition Documentation

### 17.22.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.22.2 Typedef Documentation

### 17.22.2.1 RNS

```
typedef FFPACK::rns\_double RNS
```

### 17.22.2.2 Field

```
typedef FFPACK::RNSInteger<RNS> Field
```

### 17.22.2.3 Element\_ptr

```
typedef Field::Element\_ptr Element\_ptr
```

### 17.22.2.4 ConstElement\_ptr

```
typedef Field::ConstElement\_ptr ConstElement\_ptr
```

### 17.22.2.5 THREADS

```
typedef StrategyParameter::Threads THREADS
```

### 17.22.2.6 GRAIN

```
typedef StrategyParameter::Grain GRAIN
```

### 17.22.2.7 TWOD

```
typedef StrategyParameter::TwoD TWOD
```

### 17.22.2.8 TWODA

```
typedef StrategyParameter::TwoDAdaptive TWODA
```

### 17.22.2.9 THREED

```
typedef StrategyParameter::ThreeD THREED
```

### 17.22.2.10 THREEDA

```
typedef StrategyParameter::ThreeDAdaptive THREEDA
```

### 17.22.2.11 THREEDIP

```
typedef StrategyParameter::ThreeDInPlace THREEDIP
```

### 17.22.2.12 PSeq

```
typedef ParSeqHelper::Sequential PSeq
```

## 17.22.3 Function Documentation

### 17.22.3.1 main()

```
int main (
    int argc,
    char * argv[] )
```

## 17.23 benchmark-fgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define [CLASSIC\\_HYBRID](#)

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.23.1 Macro Definition Documentation

### 17.23.1.1 CLASSIC\_HYBRID

```
#define CLASSIC_HYBRID
```

## 17.23.2 Function Documentation

### 17.23.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.24 benchmark-fgemv-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <typeinfo>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
#include "fflas-ffpack/paladin/parallel.h"
```

## Macros

- `#define \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET 1`
- `#define MG\_DEFAULT MG_ACTIVE`
- `#define STD\_RECINT\_SIZE 8`

## Functions

- `template<typename T >`  
`std::ostream & write\_matrix (std::ostream &out, Givaro::Integer p, size_t m, size_t n, T *C, size_t ldc)`
- `template<typename Ints >`  
`int tmain ()`
- `int main (int argc, char **argv)`

## 17.24.1 Macro Definition Documentation

### 17.24.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.24.1.2 MG\_DEFAULT

```
#define MG_DEFAULT MG_ACTIVE
```

### 17.24.1.3 STD\_RECINT\_SIZE

```
#define STD_RECINT_SIZE 8
```

## 17.24.2 Function Documentation

### 17.24.2.1 write\_matrix()

```
std::ostream & write_matrix (
    std::ostream & out,
    Givaro::Integer p,
    size_t m,
    size_t n,
    T * C,
    size_t ldc )
```

### 17.24.2.2 tmain()

```
int tmain ( )
```

### 17.24.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.25 benchmark-fgemv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "givaro/modular-integer.h"
#include "givaro/givcaster.h"
```

## Data Structures

- struct [need\\_field\\_characteristic< Field >](#)
- struct [need\\_field\\_characteristic< Givaro::Modular< Field > >](#)
- struct [need\\_field\\_characteristic< Givaro::ModularBalanced< Field > >](#)
- struct [compatible\\_data\\_type< Field >](#)
- struct [compatible\\_data\\_type< Givaro::ZRing< float > >](#)
- struct [compatible\\_data\\_type< Givaro::ZRing< double > >](#)

## Macros

- [#define \\_\\_FFLASFFPACK\\_OPENBLAS\\_NT\\_ALREADY\\_SET 1](#)

## Functions

- `template<class Field, class RandIter, class Matrix, class Vector >`  
`void fill_value (Field &F, RandIter &Rand, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, int NBK)`
- `template<class Field, class Matrix, class Vector >`  
`void genData (Field &F, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, int NBK, int bitsize, uint64_t seed)`
- `template<class Field, class Matrix, class Vector >`  
`bool check_result (Field &F, size_t m, size_t lda, Matrix &A, Vector &X, size_t incX, Vector &Y, size_t incY)`
- `template<class Field, class Matrix, class Vector >`  
`bool benchmark_with_timer (Field &F, int p, Matrix &A, Vector &X, Vector &Y, size_t m, size_t k, size_t incX, size_t incY, size_t lda, size_t iters, int t, double &time, size_t GrainSize)`
- `template<class Field, class arg >`  
`void benchmark_disp (Field &F, bool pass, double &time, size_t iters, int p, size_t m, size_t k, arg &as)`
- `template<class Field, class arg >`  
`void benchmark_in_Field (Field &F, int p, size_t m, size_t k, int NBK, int bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)`
- `template<class Field, class arg >`  
`void benchmark_with_field (int p, size_t m, size_t k, int NBK, int bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)`
- `template<class Field, class arg >`  
`void benchmark_with_field (const Givaro::Integer &q, int p, size_t m, size_t k, int NBK, int bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)`
- `int main (int argc, char **argv)`

## 17.25.1 Macro Definition Documentation

### 17.25.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.25.2 Function Documentation

### 17.25.2.1 fill\_value()

```
void fill_value (
    Field & F,
    RandIter & Rand,
    Matrix & A,
    Vector & X,
    Vector & Y,
    size_t m,
    size_t k,
    size_t incX,
    size_t incY,
    size_t lda,
    int NBK )
```

### 17.25.2.2 genData()

```
void genData (
    Field & F,
    Matrix & A,
```



```
Vector & X,  
Vector & Y,  
size_t m,  
size_t k,  
size_t incX,  
size_t incY,  
size_t lda,  
int NBK,  
int bitsize,  
uint64_t seed )
```

### 17.25.2.3 check\_result()

```
bool check_result (   
    Field & F,  
    size_t m,  
    size_t lda,  
    Matrix & A,  
    Vector & X,  
    size_t incX,  
    Vector & Y,  
    size_t incY )
```

### 17.25.2.4 benchmark\_with\_timer()

```
bool benchmark_with_timer (   
    Field & F,  
    int p,  
    Matrix & A,  
    Vector & X,  
    Vector & Y,  
    size_t m,  
    size_t k,  
    size_t incX,  
    size_t incY,  
    size_t lda,  
    size_t iters,  
    int t,  
    double & time,  
    size_t GrainSize )
```

### 17.25.2.5 benchmark\_disp()

```
void benchmark_disp (   
    Field & F,  
    bool pass,  
    double & time,  
    size_t iters,  
    int p,  
    size_t m,  
    size_t k,  
    arg & as )
```

### 17.25.2.6 benchmark\_in\_Field()

```
void benchmark_in_Field (
    Field & F,
    int p,
    size_t m,
    size_t k,
    int NBK,
    int bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )
```

### 17.25.2.7 benchmark\_with\_field() [1/2]

```
void benchmark_with_field (
    int p,
    size_t m,
    size_t k,
    int NBK,
    int bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )
```

### 17.25.2.8 benchmark\_with\_field() [2/2]

```
void benchmark_with_field (
    const Givaro::Integer & q,
    int p,
    size_t m,
    size_t k,
    int NBK,
    int bitsize,
    uint64_t seed,
    size_t iters,
    int t,
    arg & as,
    size_t GrainSize )
```

### 17.25.2.9 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.26 benchmark-fgesv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
```

```
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `int main (int argc, char **argv)`

### 17.26.1 Macro Definition Documentation

#### 17.26.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.26.2 Function Documentation

#### 17.26.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.27 benchmark-fsyrrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`
- `#define CUBE(x) ((x)*(x)*(x))`

## Functions

- `int main (int argc, char **argv)`

### 17.27.1 Macro Definition Documentation

**17.27.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

**17.27.1.2 CUBE**

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```

**17.27.2 Function Documentation****17.27.2.1 main()**

```
int main (  
    int argc,  
    char ** argv )
```

**17.28 benchmark-fsytrf.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular.h>  
#include "fflas-ffpack/fflas-ffpack.h"  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/utils/fflas_io.h"  
#include "fflas-ffpack/utils/args-parser.h"
```

**Macros**

- [#define \\_\\_FFPACK\\_FSYTRF\\_BC\\_CROUT](#)
- [#define \\_\\_FFLASFFPACK\\_OPENBLAS\\_NT\\_ALREADY\\_SET 1](#)
- [#define CUBE\(x\) \(\(x\)\\*\(x\)\\*\(x\)\)](#)

**Functions**

- [int main](#) (int argc, char \*\*argv)

**17.28.1 Macro Definition Documentation****17.28.1.1 \_\_FFPACK\_FSYTRF\_BC\_CROUT**

```
#define __FFPACK_FSYTRF_BC_CROUT
```

**17.28.1.2 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

**17.28.1.3 CUBE**

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```

## 17.28.2 Function Documentation

### 17.28.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.29 benchmark-fftrsm-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <vector>
#include <string>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "givaro/modular-integer.h"
```

### Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

### Functions

- `int main (int argc, char **argv)`

## 17.29.1 Macro Definition Documentation

### 17.29.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

## 17.29.2 Function Documentation

### 17.29.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.30 benchmark-fftrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `int main (int argc, char **argv)`

### 17.30.1 Macro Definition Documentation

#### 17.30.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.30.2 Function Documentation

#### 17.30.2.1 main()

```
int main (  
    int argc,  
    char ** argv )
```

## 17.31 benchmark-fftrsv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <givaro/modular.h>  
#include "fflas-ffpack/fflas-ffpack.h"  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`

## Functions

- `int main (int argc, char **argv)`

### 17.31.1 Macro Definition Documentation

#### 17.31.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.31.2 Function Documentation

**17.31.2.1 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.32 benchmark-fftrtri.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

**Macros**

- `#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1`
- `#define CUBE(x) ((x)*(x)*(x))`

**Functions**

- `int main (int argc, char **argv)`

**17.32.1 Macro Definition Documentation****17.32.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET**

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

**17.32.1.2 CUBE**

```
#define CUBE(
    x ) ( (x)*(x)*(x) )
```

**17.32.2 Function Documentation****17.32.2.1 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.33 benchmark-inverse.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
```

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Macros

- `#define CUBE(x) ((x)*(x)*(x))`

## Functions

- `int main (int argc, char **argv)`

### 17.33.1 Macro Definition Documentation

#### 17.33.1.1 CUBE

```
#define CUBE(  
    x )  ( (x)*(x)*(x) )
```

### 17.33.2 Function Documentation

#### 17.33.2.1 main()

```
int main (  
    int  argc,  
    char ** argv )
```

## 17.34 benchmark-lqup-mp.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iostream>  
#include <vector>  
#include <string>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include "fflas-ffpack/utils/args-parser.h"  
#include "givaro/modular-integer.h"
```

## Functions

- `int main (int argc, char **argv)`

### 17.34.1 Function Documentation

#### 17.34.1.1 main()

```
int main (  
    int  argc,  
    char ** argv )
```



## 17.35 benchmark-lqup.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))

### Functions

- int [main](#) (int argc, char \*\*argv)

## 17.35.1 Macro Definition Documentation

### 17.35.1.1 CUBE

```
#define CUBE(
    x )  ((x)*(x)*(x))
```

## 17.35.2 Function Documentation

### 17.35.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.36 benchmark-pluq.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular.h>
#include <givaro/givranditer.h>
#include <iostream>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_OPENBLAS\\_NT\\_ALREADY\\_SET](#) 1
- #define [CUBE](#)(x) ((x)\*(x)\*(x))

## Typedefs

- typedef [Givaro::ModularBalanced](#)< double > [Field](#)

## Functions

- void [verification\\_PLUQ](#) (const [Field](#) &F, typename [Field::Element](#) \*B, typename [Field::Element](#) \*A, size\_t \*P, size\_t \*Q, size\_t m, size\_t n, size\_t R)
- void [Rec\\_Initialize](#) ([Field](#) &F, [Field::Element](#) \*C, size\_t m, size\_t n, size\_t ldc)
- int [main](#) (int argc, char \*\*argv)

## 17.36.1 Macro Definition Documentation

### 17.36.1.1 \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

### 17.36.1.2 CUBE

```
#define CUBE(  
    x ) ((x)*(x)*(x))
```

## 17.36.2 Typedef Documentation

### 17.36.2.1 Field

```
typedef Givaro::ModularBalanced<double> Field
```

## 17.36.3 Function Documentation

### 17.36.3.1 verification\_PLUQ()

```
void verification_PLUQ (  
    const Field & F,  
    typename Field::Element * B,  
    typename Field::Element * A,  
    size_t * P,  
    size_t * Q,  
    size_t m,  
    size_t n,  
    size_t R )
```

### 17.36.3.2 Rec\_Initialize()

```
void Rec_Initialize (  
    Field & F,  
    Field::Element * C,  
    size_t m,  
    size_t n,  
    size_t ldc )
```

### 17.36.3.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.37 benchmark-wino.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <fstream>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Macros

- #define [CUBE](#)(x) ((x)\*(x)\*(x))

### Functions

- template<class [Field](#) >  
void [launch\\_wino](#) (const [Field](#) &F, const size\_t &n, const size\_t &NB, const size\_t &wino, const bool &asmax, const size\_t &seed, const bool compare)
- int [main](#) (int argc, char \*\*argv)

## 17.37.1 Macro Definition Documentation

### 17.37.1.1 CUBE

```
#define CUBE(
    x ) ((x)*(x)*(x))
```

## 17.37.2 Function Documentation

### 17.37.2.1 launch\_wino()

```
void launch_wino (
    const Field & F,
    const size_t & n,
    const size_t & NB,
    const size_t & wino,
    const bool & asmax,
    const size_t & seed,
    const bool compare )
```

### 17.37.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.38 config.h File Reference

### Macros

- #define [HAVE\\_BLAS](#) 1
- #define [HAVE\\_CBLAS](#) 1
- #define [HAVE\\_CXX11](#) 1
- #define [HAVE\\_DLFCN\\_H](#) 1
- #define [HAVE\\_FLOAT\\_H](#) 1
- #define [HAVE\\_INT128](#) 1
- #define [HAVE\\_INTPYPES\\_H](#) 1
- #define [HAVE\\_LAPACK](#) 1
- #define [HAVE\\_LIMITS\\_H](#) 1
- #define [HAVE\\_LITTLE\\_ENDIAN](#) 1
- #define [HAVE\\_PTHREAD\\_H](#) 1
- #define [HAVE\\_STDDEF\\_H](#) 1
- #define [HAVE\\_STDINT\\_H](#) 1
- #define [HAVE\\_STDIO\\_H](#) 1
- #define [HAVE\\_STDLIB\\_H](#) 1
- #define [HAVE\\_STRINGS\\_H](#) 1
- #define [HAVE\\_STRING\\_H](#) 1
- #define [HAVE\\_SYS\\_STAT\\_H](#) 1
- #define [HAVE\\_SYS\\_TIME\\_H](#) 1
- #define [HAVE\\_SYS\\_TYPES\\_H](#) 1
- #define [HAVE\\_UNISTD\\_H](#) 1
- #define [LT\\_OBJDIR](#) ".libs/"
- #define [OPENBLAS\\_NUM\\_THREADS](#) 1
- #define [PACKAGE](#) "fflas-ffpack"
- #define [PACKAGE\\_BUGREPORT](#) "ffpack-devel@googlegroups.com"
- #define [PACKAGE\\_NAME](#) "FFLAS-FFPACK"
- #define [PACKAGE\\_STRING](#) "FFLAS-FFPACK 2.4.3"
- #define [PACKAGE\\_TARNAME](#) "fflas-ffpack"
- #define [PACKAGE\\_URL](#) "https://github.com/linbox-team/fflas-ffpack"
- #define [PACKAGE\\_VERSION](#) "2.4.3"
- #define [SIZEOF\\_CHAR](#) 1
- #define [SIZEOF\\_INT](#) 4
- #define [SIZEOF\\_LONG](#) 8
- #define [SIZEOF\\_LONG\\_LONG](#) 8
- #define [SIZEOF\\_SHORT](#) 2
- #define [SIZEOF\\_\\_\\_INT64](#) 0
- #define [STDC\\_HEADERS](#) 1
- #define [USE\\_OPENMP](#) 1
- #define [VERSION](#) "2.4.3"

### 17.38.1 Macro Definition Documentation

#### 17.38.1.1 HAVE\_BLAS

```
#define HAVE_BLAS 1
```

#### 17.38.1.2 HAVE\_CBLAS

```
#define HAVE_CBLAS 1
```

**17.38.1.3 HAVE\_CXX11**

```
#define HAVE_CXX11 1
```

**17.38.1.4 HAVE\_DLFCN\_H**

```
#define HAVE_DLFCN_H 1
```

**17.38.1.5 HAVE\_FLOAT\_H**

```
#define HAVE_FLOAT_H 1
```

**17.38.1.6 HAVE\_INT128**

```
#define HAVE_INT128 1
```

**17.38.1.7 HAVE\_INTTYPES\_H**

```
#define HAVE_INTTYPES_H 1
```

**17.38.1.8 HAVE\_LAPACK**

```
#define HAVE_LAPACK 1
```

**17.38.1.9 HAVE\_LIMITS\_H**

```
#define HAVE_LIMITS_H 1
```

**17.38.1.10 HAVE\_LITTLE\_ENDIAN**

```
#define HAVE_LITTLE_ENDIAN 1
```

**17.38.1.11 HAVE\_PTHREAD\_H**

```
#define HAVE_PTHREAD_H 1
```

**17.38.1.12 HAVE\_STDDEF\_H**

```
#define HAVE_STDDEF_H 1
```

**17.38.1.13 HAVE\_STDINT\_H**

```
#define HAVE_STDINT_H 1
```

**17.38.1.14 HAVE\_STDIO\_H**

```
#define HAVE_STDIO_H 1
```

**17.38.1.15 HAVE\_STDLIB\_H**

```
#define HAVE_STDLIB_H 1
```

**17.38.1.16 HAVE\_STRINGS\_H**

```
#define HAVE_STRINGS_H 1
```

**17.38.1.17 HAVE\_STRING\_H**

```
#define HAVE_STRING_H 1
```

**17.38.1.18 HAVE\_SYS\_STAT\_H**

```
#define HAVE_SYS_STAT_H 1
```

**17.38.1.19 HAVE\_SYS\_TIME\_H**

```
#define HAVE_SYS_TIME_H 1
```

**17.38.1.20 HAVE\_SYS\_TYPES\_H**

```
#define HAVE_SYS_TYPES_H 1
```

**17.38.1.21 HAVE\_UNISTD\_H**

```
#define HAVE_UNISTD_H 1
```

**17.38.1.22 LT\_OBJDIR**

```
#define LT_OBJDIR ".libs/"
```

**17.38.1.23 OPENBLAS\_NUM\_THREADS**

```
#define OPENBLAS_NUM_THREADS 1
```

**17.38.1.24 PACKAGE**

```
#define PACKAGE "fflas-ffpack"
```

**17.38.1.25 PACKAGE\_BUGREPORT**

```
#define PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

**17.38.1.26 PACKAGE\_NAME**

```
#define PACKAGE_NAME "FFLAS-FFPACK"
```

**17.38.1.27 PACKAGE\_STRING**

```
#define PACKAGE_STRING "FFLAS-FFPACK 2.4.3"
```

**17.38.1.28 PACKAGE\_TARNAME**

```
#define PACKAGE_TARNAME "fflas-ffpack"
```

**17.38.1.29 PACKAGE\_URL**

```
#define PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

**17.38.1.30 PACKAGE\_VERSION**

```
#define PACKAGE_VERSION "2.4.3"
```

**17.38.1.31 SIZEOF\_CHAR**

```
#define SIZEOF_CHAR 1
```

**17.38.1.32 SIZEOF\_INT**

```
#define SIZEOF_INT 4
```

**17.38.1.33 SIZEOF\_LONG**

```
#define SIZEOF_LONG 8
```

**17.38.1.34 SIZEOF\_LONG\_LONG**

```
#define SIZEOF_LONG_LONG 8
```

**17.38.1.35 SIZEOF\_SHORT**

```
#define SIZEOF_SHORT 2
```

**17.38.1.36 SIZEOF\_\_\_INT64**

```
#define SIZEOF___INT64 0
```

**17.38.1.37 STDC\_HEADERS**

```
#define STDC_HEADERS 1
```

**17.38.1.38 USE\_OPENMP**

```
#define USE_OPENMP 1
```

### 17.38.1.39 VERSION

```
#define VERSION "2.4.3"
```

## 17.39 config.h File Reference

### Macros

- [#define \\_\\_FFLASFFPACK\\_HAVE\\_BLAS 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_CBLAS 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_CXX11 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_DLFCN\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_FLOAT\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_INT128 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_INTTYPES\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_LAPACK 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_LIMITS\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_LITTLE\\_ENDIAN 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_PTHREAD\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STDDEF\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STDINT\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STDIO\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STDLIB\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STRINGS\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_STRING\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_STAT\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_TIME\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_TYPES\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_HAVE\\_UNISTD\\_H 1](#)
- [#define \\_\\_FFLASFFPACK\\_LT\\_OBJDIR ".libs/"](#)
- [#define \\_\\_FFLASFFPACK\\_OPENBLAS\\_NUM\\_THREADS 1](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE "fflas-ffpack"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_BUGREPORT "ffpack-devel@googlegroups.com"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_NAME "FFLAS-FFPACK"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_STRING "FFLAS-FFPACK 2.4.3"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_TARNAME "fflas-ffpack"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_URL "https://github.com/linbox-team/fflas-ffpack"](#)
- [#define \\_\\_FFLASFFPACK\\_PACKAGE\\_VERSION "2.4.3"](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_CHAR 1](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_INT 4](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_LONG 8](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_LONG\\_LONG 8](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_SHORT 2](#)
- [#define \\_\\_FFLASFFPACK\\_SIZEOF\\_\\_INT64 0](#)
- [#define \\_\\_FFLASFFPACK\\_STDC\\_HEADERS 1](#)
- [#define \\_\\_FFLASFFPACK\\_USE\\_OPENMP 1](#)
- [#define \\_\\_FFLASFFPACK\\_VERSION "2.4.3"](#)

### 17.39.1 Macro Definition Documentation

#### 17.39.1.1 \_\_FFLASFFPACK\_HAVE\_BLAS

```
#define __FFLASFFPACK_HAVE_BLAS 1
```



**17.39.1.2 \_\_FFLASFFPACK\_HAVE\_CBLAS**

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

**17.39.1.3 \_\_FFLASFFPACK\_HAVE\_CXX11**

```
#define __FFLASFFPACK_HAVE_CXX11 1
```

**17.39.1.4 \_\_FFLASFFPACK\_HAVE\_DLFCN\_H**

```
#define __FFLASFFPACK_HAVE_DLFCN_H 1
```

**17.39.1.5 \_\_FFLASFFPACK\_HAVE\_FLOAT\_H**

```
#define __FFLASFFPACK_HAVE_FLOAT_H 1
```

**17.39.1.6 \_\_FFLASFFPACK\_HAVE\_INT128**

```
#define __FFLASFFPACK_HAVE_INT128 1
```

**17.39.1.7 \_\_FFLASFFPACK\_HAVE\_INTTYPES\_H**

```
#define __FFLASFFPACK_HAVE_INTTYPES_H 1
```

**17.39.1.8 \_\_FFLASFFPACK\_HAVE\_LAPACK**

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

**17.39.1.9 \_\_FFLASFFPACK\_HAVE\_LIMITS\_H**

```
#define __FFLASFFPACK_HAVE_LIMITS_H 1
```

**17.39.1.10 \_\_FFLASFFPACK\_HAVE\_LITTLE\_ENDIAN**

```
#define __FFLASFFPACK_HAVE_LITTLE_ENDIAN 1
```

**17.39.1.11 \_\_FFLASFFPACK\_HAVE\_PTHREAD\_H**

```
#define __FFLASFFPACK_HAVE_PTHREAD_H 1
```

**17.39.1.12 \_\_FFLASFFPACK\_HAVE\_STDDEF\_H**

```
#define __FFLASFFPACK_HAVE_STDDEF_H 1
```

**17.39.1.13 \_\_FFLASFFPACK\_HAVE\_STDINT\_H**

```
#define __FFLASFFPACK_HAVE_STDINT_H 1
```

**17.39.1.14 \_\_FFLASFFPACK\_HAVE\_STDIO\_H**

```
#define __FFLASFFPACK_HAVE_STDIO_H 1
```

**17.39.1.15 \_\_FFLASFFPACK\_HAVE\_STDLIB\_H**

```
#define __FFLASFFPACK_HAVE_STDLIB_H 1
```

**17.39.1.16 \_\_FFLASFFPACK\_HAVE\_STRINGS\_H**

```
#define __FFLASFFPACK_HAVE_STRINGS_H 1
```

**17.39.1.17 \_\_FFLASFFPACK\_HAVE\_STRING\_H**

```
#define __FFLASFFPACK_HAVE_STRING_H 1
```

**17.39.1.18 \_\_FFLASFFPACK\_HAVE\_SYS\_STAT\_H**

```
#define __FFLASFFPACK_HAVE_SYS_STAT_H 1
```

**17.39.1.19 \_\_FFLASFFPACK\_HAVE\_SYS\_TIME\_H**

```
#define __FFLASFFPACK_HAVE_SYS_TIME_H 1
```

**17.39.1.20 \_\_FFLASFFPACK\_HAVE\_SYS\_TYPES\_H**

```
#define __FFLASFFPACK_HAVE_SYS_TYPES_H 1
```

**17.39.1.21 \_\_FFLASFFPACK\_HAVE\_UNISTD\_H**

```
#define __FFLASFFPACK_HAVE_UNISTD_H 1
```

**17.39.1.22 \_\_FFLASFFPACK\_LT\_OBJDIR**

```
#define __FFLASFFPACK_LT_OBJDIR ".libs/"
```

**17.39.1.23 \_\_FFLASFFPACK\_OPENBLAS\_NUM\_THREADS**

```
#define __FFLASFFPACK_OPENBLAS_NUM_THREADS 1
```

**17.39.1.24 \_\_FFLASFFPACK\_PACKAGE**

```
#define __FFLASFFPACK_PACKAGE "fflas-ffpack"
```

**17.39.1.25 \_\_FFLASFFPACK\_PACKAGE\_BUGREPORT**

```
#define __FFLASFFPACK_PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

**17.39.1.26 \_\_FFLASFFPACK\_PACKAGE\_NAME**

```
#define __FFLASFFPACK_PACKAGE_NAME "FFLAS-FFPACK"
```

**17.39.1.27 \_\_FFLASFFPACK\_PACKAGE\_STRING**

```
#define __FFLASFFPACK_PACKAGE_STRING "FFLAS-FFPACK 2.4.3"
```

**17.39.1.28 \_\_FFLASFFPACK\_PACKAGE\_TARNAME**

```
#define __FFLASFFPACK_PACKAGE_TARNAME "fflas-ffpack"
```

**17.39.1.29 \_\_FFLASFFPACK\_PACKAGE\_URL**

```
#define __FFLASFFPACK_PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

**17.39.1.30 \_\_FFLASFFPACK\_PACKAGE\_VERSION**

```
#define __FFLASFFPACK_PACKAGE_VERSION "2.4.3"
```

**17.39.1.31 \_\_FFLASFFPACK\_SIZEOF\_CHAR**

```
#define __FFLASFFPACK_SIZEOF_CHAR 1
```

**17.39.1.32 \_\_FFLASFFPACK\_SIZEOF\_INT**

```
#define __FFLASFFPACK_SIZEOF_INT 4
```

**17.39.1.33 \_\_FFLASFFPACK\_SIZEOF\_LONG**

```
#define __FFLASFFPACK_SIZEOF_LONG 8
```

**17.39.1.34 \_\_FFLASFFPACK\_SIZEOF\_LONG\_LONG**

```
#define __FFLASFFPACK_SIZEOF_LONG_LONG 8
```

**17.39.1.35 \_\_FFLASFFPACK\_SIZEOF\_SHORT**

```
#define __FFLASFFPACK_SIZEOF_SHORT 2
```

**17.39.1.36 \_\_FFLASFFPACK\_SIZEOF\_\_INT64**

```
#define __FFLASFFPACK_SIZEOF__INT64 0
```

**17.39.1.37 \_\_FFLASFFPACK\_STDC\_HEADERS**

```
#define __FFLASFFPACK_STDC_HEADERS 1
```

**17.39.1.38 \_\_FFLASFFPACK\_USE\_OPENMP**

```
#define __FFLASFFPACK_USE_OPENMP 1
```

**17.39.1.39 \_\_FFLASFFPACK\_VERSION**

```
#define __FFLASFFPACK_VERSION "2.4.3"
```

**17.40 mainpage.doxy File Reference****17.41 det.C File Reference**

```
#include <givaro/modular.h>
#include <iostream>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

**Functions**

- int [main](#) (int argc, char \*\*argv)

*This example computes the determinant of a matrix over a defined finite field.*

**17.41.1 Function Documentation****17.41.1.1 main()**

```
int main (
    int argc,
    char ** argv )
```

This example computes the determinant of a matrix over a defined finite field.  
Outputs the determinant.

**17.42 matmul.C File Reference**

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

**Functions**

- int [main](#) (int argc, char \*\*argv)

*This example computes the matrix multiplication over a defined finite field.*

**17.42.1 Function Documentation**

### 17.42.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the matrix multiplication over a defined finite field.  
Outputs the product of the matrix given as input.

## 17.43 rank.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

*This example computes the rank of a matrix over a defined finite field.*

### 17.43.1 Function Documentation

#### 17.43.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example computes the rank of a matrix over a defined finite field.  
Outputs the rank.

## 17.44 solve.C File Reference

```
#include <iostream>
#include "fflas-ffpack/fflas-ffpack.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

*This example solve the quare system defined by the input over a defined finite field.*

### 17.44.1 Function Documentation

#### 17.44.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

This example solve the quare system defined by the input over a defined finite field.

## 17.45 checker\_charpoly.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

## Data Structures

- class [CheckerImplem\\_charpoly](#)< Field, Polynomial >

## Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_charpoly\\_INL](#)

### 17.45.1 Macro Definition Documentation

#### 17.45.1.1 [\\_\\_FFLASFFPACK\\_checker\\_charpoly\\_INL](#)

```
#define __FFLASFFPACK_checker_charpoly_INL
```

## 17.46 checker\_det.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

## Data Structures

- class [CheckerImplem\\_Det](#)< Field >

## Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_det\\_INL](#)

### 17.46.1 Macro Definition Documentation

#### 17.46.1.1 [\\_\\_FFLASFFPACK\\_checker\\_det\\_INL](#)

```
#define __FFLASFFPACK_checker_det_INL
```

## 17.47 checker\_empty.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
```

## Data Structures

- struct [Checker\\_Empty](#)< Field >

## Namespaces

- namespace [FFLAS](#)

## 17.48 checker\_fgemm.inl File Reference

### Data Structures

- class [CheckerImplem\\_fgemm](#)< Field >

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_fgemm\\_INL](#)

#### 17.48.1 Macro Definition Documentation

##### 17.48.1.1 \_\_FFLASFFPACK\_checker\_fgemm\_INL

```
#define __FFLASFFPACK_checker_fgemm_INL
```

## 17.49 checker\_ftrsm.inl File Reference

### Data Structures

- class [CheckerImplem\\_ftrsm](#)< Field >

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_checker\\_ftrsm\\_INL](#)

#### 17.49.1 Macro Definition Documentation

##### 17.49.1.1 \_\_FFLASFFPACK\_checker\_ftrsm\_INL

```
#define __FFLASFFPACK_checker_ftrsm_INL
```

## 17.50 checker\_invert.inl File Reference

### Data Structures

- class [CheckerImplem\\_invert](#)< Field >

### Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define __FFLASFFPACK_checker_invert_INL`

### 17.50.1 Macro Definition Documentation

#### 17.50.1.1 \_\_FFLASFFPACK\_checker\_invert\_INL

```
#define __FFLASFFPACK_checker_invert_INL
```

## 17.51 checker\_pluq.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

## Data Structures

- class [CheckerImplem\\_PLUQ< Field >](#)

## Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define __FFLASFFPACK_checker_pluq_INL`

### 17.51.1 Macro Definition Documentation

#### 17.51.1.1 \_\_FFLASFFPACK\_checker\_pluq\_INL

```
#define __FFLASFFPACK_checker_pluq_INL
```

## 17.52 checkers.doxy File Reference

## 17.53 checkers\_fflas.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "checker_empty.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_enum.h"
#include "fflas-ffpack/utils/fflas_memory.h"
```

## Data Structures

- class [FailureFgemmCheck](#)
- class [FailureTrsmCheck](#)



## Namespaces

- namespace [FFLAS](#)

## Typedefs

- template<class [Field](#) >  
using [Checker\\_fgemm](#) = [FFLAS::Checker\\_Empty](#)< [Field](#) >
- template<class [Field](#) >  
using [Checker\\_ftrsm](#) = [FFLAS::Checker\\_Empty](#)< [Field](#) >

## 17.54 checkers\_fflas.inl File Reference

```
#include "checker_fgemm.inl"  
#include "checker_ftrsm.inl"
```

## Namespaces

- namespace [FFLAS](#)

## Macros

- #define [FFLASFFPACK\\_checkers\\_fflas\\_inl\\_H](#)

## Typedefs

- template<class [Field](#) >  
using [ForceCheck\\_fgemm](#) = [CheckerImplem\\_fgemm](#)< [Field](#) >
- template<class [Field](#) >  
using [ForceCheck\\_ftrsm](#) = [CheckerImplem\\_ftrsm](#)< [Field](#) >

### 17.54.1 Macro Definition Documentation

#### 17.54.1.1 FFLASFFPACK\_checkers\_fflas\_inl\_H

```
#define FFLASFFPACK_checkers_fflas_inl_H
```

## 17.55 checkers\_ffpack.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include "checker_empty.h"  
#include "fflas-ffpack/ffpack/ffpack.h"
```

## Data Structures

- class [FailurePLUQCheck](#)
- class [FailureDetCheck](#)
- class [FailureInvertCheck](#)
- class [FailureCharpolyCheck](#)

## Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Typedefs

- `template<class Field >`  
using [Checker\\_PLUQ](#) = [FFLAS::Checker\\_Empty](#)< [Field](#) >
- `template<class Field >`  
using [Checker\\_Det](#) = [FFLAS::Checker\\_Empty](#)< [Field](#) >
- `template<class Field >`  
using [Checker\\_invert](#) = [FFLAS::Checker\\_Empty](#)< [Field](#) >
- `template<class Field , class Polynomial >`  
using [Checker\\_charpoly](#) = [FFLAS::Checker\\_Empty](#)< [Field](#) >

## 17.56 checkers\_ffpack.inl File Reference

```
#include "checker_pluq.inl"
#include "checker_det.inl"
#include "checker_invert.inl"
#include "checker_charpoly.inl"
```

## Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- `#define FFLASFFPACK\_checkers\_ffpack\_inl\_H`

## Typedefs

- `template<class Field >`  
using [ForceCheck\\_PLUQ](#) = [CheckerImplem\\_PLUQ](#)< [Field](#) >
- `template<class Field >`  
using [ForceCheck\\_Det](#) = [CheckerImplem\\_Det](#)< [Field](#) >
- `template<class Field >`  
using [ForceCheck\\_invert](#) = [CheckerImplem\\_invert](#)< [Field](#) >
- `template<class Field , class Polynomial >`  
using [ForceCheck\\_charpoly](#) = [CheckerImplem\\_charpoly](#)< [Field](#), Polynomial >

### 17.56.1 Macro Definition Documentation

#### 17.56.1.1 FFLASFFPACK\_checkers\_ffpack\_inl\_H

```
#define FFLASFFPACK_checkers_ffpack_inl_H
```

## 17.57 config-blas.h File Reference

### Macros

- #define `CBLAS_INT` int
- #define `CBLAS_ENUM_DEFINED_H`
- #define `CBLAS_EXTERNALS`
- #define `blas_enum` enum

### Enumerations

- enum `CBLAS_ORDER` { `CblasRowMajor` =101 , `CblasColMajor` =102 }
- enum `CBLAS_TRANSPOSE` { `CblasNoTrans` =111 , `CblasTrans` =112 , `CblasConjTrans` =113 , `AtlasConj` =114 }
- enum `CBLAS_UPLO` { `CblasUpper` =121 , `CblasLower` =122 }
- enum `CBLAS_DIAG` { `CblasNonUnit` =131 , `CblasUnit` =132 }
- enum `CBLAS_SIDE` { `CblasLeft` =141 , `CblasRight` =142 }

### Functions

- void `daxpy_` (const int \*, const double \*, const double \*, const int \*, double \*, const int \*)
- void `saxpy_` (const int \*, const float \*, const float \*, const int \*, float \*, const int \*)
- double `ddot_` (const int \*, const double \*, const int \*, const double \*, const int \*)
- float `sdot_` (const int \*, const float \*, const int \*, const float \*, const int \*)
- double `dasum_` (const int \*, const double \*, const int \*)
- int `idamax_` (const int \*, const double \*, const int \*)
- double `dnrm2_` (const int \*, const double \*, const int \*)
- void `dgemv_` (const char \*, const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, const double \*, double \*, const int \*)
- void `sgemv_` (const char \*, const int \*, const int \*, const float \*, const float \*, const int \*, const float \*, const int \*, const float \*, float \*, const int \*)
- void `dger_` (const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, double \*, const int \*)
- void `sger_` (const int \*, const int \*, const float \*, const float \*, const int \*, const float \*, const int \*, float \*, const int \*)
- void `dcopy_` (const int \*, const double \*, const int \*, double \*, const int \*)
- void `scopy_` (const int \*, const float \*, const int \*, float \*, const int \*)
- void `dscal_` (const int \*, const double \*, double \*, const int \*)
- void `sscal_` (const int \*, const float \*, float \*, const int \*)
- void `dtrsm_` (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const double \*, const double \*, const int \*, double \*, const int \*)
- void `strsm_` (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const float \*, const float \*, const int \*, float \*, const int \*)
- void `dtrmm_` (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const double \*, const double \*, const int \*, double \*, const int \*)
- void `strmm_` (const char \*, const char \*, const char \*, const char \*, const int \*, const int \*, const float \*, const float \*, const int \*, float \*, const int \*)
- void `sgemm_` (const char \*, const char \*, const int \*, const int \*, const int \*, const float \*, const float \*, const int \*, const float \*, const int \*, const float \*, float \*, const int \*)
- void `dgemm_` (const char \*, const char \*, const int \*, const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, const double \*, double \*, const int \*)

#### 17.57.1 Macro Definition Documentation

**17.57.1.1 CBLAS\_INT**

```
#define CBLAS_INT int
```

**17.57.1.2 CBLAS\_ENUM\_DEFINED\_H**

```
#define CBLAS_ENUM_DEFINED_H
```

**17.57.1.3 CBLAS\_EXTERNALS**

```
#define CBLAS_EXTERNALS
```

**17.57.1.4 blas\_enum**

```
#define blas_enum enum
```

**17.57.2 Enumeration Type Documentation****17.57.2.1 CBLAS\_ORDER**

```
enum CBLAS_ORDER
```

**Enumerator**

|               |  |
|---------------|--|
| CblasRowMajor |  |
| CblasColMajor |  |

**17.57.2.2 CBLAS\_TRANSPOSE**

```
enum CBLAS_TRANSPOSE
```

**Enumerator**

|                |  |
|----------------|--|
| CblasNoTrans   |  |
| CblasTrans     |  |
| CblasConjTrans |  |
| AtlasConj      |  |

**17.57.2.3 CBLAS\_UPLO**

```
enum CBLAS_UPLO
```

**Enumerator**

|            |  |
|------------|--|
| CblasUpper |  |
| CblasLower |  |

### 17.57.2.4 CBLAS\_DIAG

enum [CBLAS\\_DIAG](#)

#### Enumerator

|              |  |
|--------------|--|
| CblasNonUnit |  |
| CblasUnit    |  |

### 17.57.2.5 CBLAS\_SIDE

enum [CBLAS\\_SIDE](#)

#### Enumerator

|            |  |
|------------|--|
| CblasLeft  |  |
| CblasRight |  |

## 17.57.3 Function Documentation

### 17.57.3.1 daxpy\_()

```
void daxpy_ (
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    double * ,
    const int * )
```

### 17.57.3.2 saxpy\_()

```
void saxpy_ (
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```

### 17.57.3.3 ddot\_()

```
double ddot_ (
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    const int * )
```

#### 17.57.3.4 sdot\_()

```
float sdot_ (
    const int * ,
    const float * ,
    const int * ,
    const float * ,
    const int * )
```

#### 17.57.3.5 dasum\_()

```
double dasum_ (
    const int * ,
    const double * ,
    const int * )
```

#### 17.57.3.6 idamax\_()

```
int idamax_ (
    const int * ,
    const double * ,
    const int * )
```

#### 17.57.3.7 dnorm2\_()

```
double dnorm2_ (
    const int * ,
    const double * ,
    const int * )
```

#### 17.57.3.8 dgemv\_()

```
void dgemv_ (
    const char * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    double * ,
    const int * )
```

#### 17.57.3.9 sgemv\_()

```
void sgemv_ (
    const char * ,
    const int * ,
    const int * ,
    const float * ,
    const float * ,
    const int * ,
```

```
const float * ,  
const int * ,  
const float * ,  
float * ,  
const int * )
```

#### 17.57.3.10 dger\_()

```
void dger_ (  
    const int * ,  
    const int * ,  
    const double * ,  
    const double * ,  
    const int * ,  
    const double * ,  
    const int * ,  
    double * ,  
    const int * )
```

#### 17.57.3.11 sger\_()

```
void sger_ (  
    const int * ,  
    const int * ,  
    const float * ,  
    const float * ,  
    const int * ,  
    const float * ,  
    const int * ,  
    float * ,  
    const int * )
```

#### 17.57.3.12 dcopy\_()

```
void dcopy_ (  
    const int * ,  
    const double * ,  
    const int * ,  
    double * ,  
    const int * )
```

#### 17.57.3.13 scopy\_()

```
void scopy_ (  
    const int * ,  
    const float * ,  
    const int * ,  
    float * ,  
    const int * )
```

#### 17.57.3.14 dscal\_()

```
void dscal_ (  
    const int * ,
```

```
    const double * ,  
    double * ,  
    const int * )
```

#### 17.57.3.15 sscal\_()

```
void sscal_ (  
    const int * ,  
    const float * ,  
    float * ,  
    const int * )
```

#### 17.57.3.16 dtrsm\_()

```
void dtrsm_ (  
    const char * ,  
    const char * ,  
    const char * ,  
    const char * ,  
    const int * ,  
    const int * ,  
    const double * ,  
    const double * ,  
    const int * ,  
    double * ,  
    const int * )
```

#### 17.57.3.17 strsm\_()

```
void strsm_ (  
    const char * ,  
    const char * ,  
    const char * ,  
    const char * ,  
    const int * ,  
    const int * ,  
    const float * ,  
    const float * ,  
    const int * ,  
    float * ,  
    const int * )
```

#### 17.57.3.18 dtrmm\_()

```
void dtrmm_ (  
    const char * ,  
    const char * ,  
    const char * ,  
    const char * ,  
    const int * ,  
    const int * ,  
    const double * ,  
    const double * ,  
    const int * ,
```



```
double * ,
const int * )
```

### 17.57.3.19 strmm\_()

```
void strmm_ (
    const char * ,
    const char * ,
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    float * ,
    const int * )
```

### 17.57.3.20 sgemm\_()

```
void sgemm_ (
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const int * ,
    const float * ,
    const float * ,
    const int * ,
    const float * ,
    const int * ,
    const float * ,
    float * ,
    const int * )
```

### 17.57.3.21 dgemm\_()

```
void dgemm_ (
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    double * ,
    const int * )
```

## 17.58 fflas-ffpack-config.h File Reference

Defaults for optimised values.

```
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/fflas-ffpack-thresholds.h"
#include "fflas-ffpack/fflas-ffpack-default-thresholds.h"
#include "givaro/givconfig.h"
```

## Macros

- #define [GCC\\_VERSION](#) (`__GNUC__` \* 10000 + `__GNUC_MINOR__` \* 100 + `__GNUC_PATCHLEVEL__`)

### 17.58.1 Detailed Description

Defaults for optimised values.

While `fflas-ffpack-optimise.h` is created by `configure` script, (either left blank or filled by optimiser), this file produces the defaults for the optimised values. If `fflas-ffpack-optimise.h` is not empty, then its values preceeds the defaults here.

### 17.58.2 Macro Definition Documentation

#### 17.58.2.1 GCC\_VERSION

```
#define GCC_VERSION ( __GNUC__ * 10000 + __GNUC_MINOR__ * 100 + __GNUC_PATCHLEVEL__ )
```

## 17.59 fflas-ffpack-default-thresholds.h File Reference

## Macros

- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD](#) 1000
- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD\\_FLT](#) 2000
- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD\\_BAL](#) 1000
- #define [\\_\\_FFLASFFPACK\\_WINOTHRESHOLD\\_BAL\\_FLT](#) 2000
- #define [\\_\\_FFLASFFPACK\\_PLUQ\\_THRESHOLD](#) 256
- #define [\\_\\_FFLASFFPACK\\_CHARPOLY\\_LUKrylov\\_ArithProg\\_THRESHOLD](#) 1000
- #define [\\_\\_FFLASFFPACK\\_CHARPOLY\\_Danilevskii\\_LUKrylov\\_THRESHOLD](#) 16
- #define [\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#) 30
- #define [\\_\\_FFLASFFPACK\\_FTRTRI\\_THRESHOLD](#) 32
- #define [\\_\\_FFLASFFPACK\\_FSYTRF\\_THRESHOLD](#) 64
- #define [\\_\\_FFLASFFPACK\\_FSYRK\\_THRESHOLD](#) 3000

### 17.59.1 Macro Definition Documentation

#### 17.59.1.1 \_\_FFLASFFPACK\_WINOTHRESHOLD

```
#define __FFLASFFPACK_WINOTHRESHOLD 1000
```

#### 17.59.1.2 \_\_FFLASFFPACK\_WINOTHRESHOLD\_FLT

```
#define __FFLASFFPACK_WINOTHRESHOLD_FLT 2000
```

**17.59.1.3 \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL**

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL 1000
```

**17.59.1.4 \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL\_FLT**

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT 2000
```

**17.59.1.5 \_\_FFLASFFPACK\_PLUQ\_THRESHOLD**

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 256
```

**17.59.1.6 \_\_FFLASFFPACK\_CHARPOLY\_LUKrylov\_ArithProg\_THRESHOLD**

```
#define __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD 1000
```

**17.59.1.7 \_\_FFLASFFPACK\_CHARPOLY\_Danilevskii\_LUKrylov\_THRESHOLD**

```
#define __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD 16
```

**17.59.1.8 \_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD**

```
#define __FFLASFFPACK_ARITHPROG_THRESHOLD 30
```

**17.59.1.9 \_\_FFLASFFPACK\_FTRTRI\_THRESHOLD**

```
#define __FFLASFFPACK_FTRTRI_THRESHOLD 32
```

**17.59.1.10 \_\_FFLASFFPACK\_FSYTRF\_THRESHOLD**

```
#define __FFLASFFPACK_FSYTRF_THRESHOLD 64
```

**17.59.1.11 \_\_FFLASFFPACK\_FSYRK\_THRESHOLD**

```
#define __FFLASFFPACK_FSYRK_THRESHOLD 3000
```

**17.60 fflas-ffpack-thresholds.h File Reference****17.61 fflas-ffpack.dox File Reference****17.62 fflas-ffpack.h File Reference**

Includes [FFLAS](#) and [FFPACK](#).

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas/fflas.h"
#include "ffpack/ffpack.h"
```

**17.62.1 Detailed Description**

Includes [FFLAS](#) and [FFPACK](#).

## 17.63 fflas.doxy File Reference

### 17.64 fflas.h File Reference

#### Finite Field Linear Algebra Subroutines

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/config-blas.h"
#include <cmath>
#include <cstring>
#include <float.h>
#include <algorithm>
#include "fflas_enum.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas_level1.inl"
#include "fflas_level2.inl"
#include "fflas_level3.inl"
#include "fflas-ffpack/checkers/checkers_fflas.h"
#include "fflas_freduce.h"
#include "fflas_fadd.h"
#include "fflas_fscal.h"
#include "fflas_fassign.h"
#include "fflas_fgemm.inl"
#include "fflas_pfgemm.inl"
#include "fflas_fgemv.inl"
#include "fflas-ffpack/paladin/pfgemv.inl"
#include "fflas_freivalds.inl"
#include "fflas_fger.inl"
#include "fflas_fsyrk.inl"
#include "fflas_fsyr2k.inl"
#include "fflas_ftrsm.inl"
#include "fflas_pftrsm.inl"
#include "fflas_ftrmm.inl"
#include "fflas_ftrsv.inl"
#include "fflas_faxpy.inl"
#include "fflas_fdot.inl"
#include "fflas-ffpack/field/rns.h"
#include "fflas_fscal_mp.inl"
#include "fflas_freduce_mp.inl"
#include "fflas-ffpack/fflas/fflas_fger_mp.inl"
#include "fflas_fgemm/fgemm_classical_mp.inl"
#include "fflas_ftrsm_mp.inl"
#include "fflas_fgemv_mp.inl"
#include "fflas-ffpack/field/rns.inl"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include "fflas_sparse.h"
#include "fflas-ffpack/checkers/checkers_fflas.inl"
```

#### Macros

- `#define WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD`
- `#define DOUBLE_TO_FLOAT_CROSSOVER 800`

*Thresholds determining which floating point representation to use, depending on the cardinality of the finite field.*

### 17.64.1 Detailed Description

Finite Field Linear Algebra Subroutines

Author

Clément Pernet.

### 17.64.2 Macro Definition Documentation

#### 17.64.2.1 WINOTHRESHOLD

```
#define WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD
```

#### 17.64.2.2 DOUBLE\_TO\_FLOAT\_CROSSOVER

```
#define DOUBLE_TO_FLOAT_CROSSOVER 800
```

Thresholds determining which floating point representation to use, depending on the cardinality of the finite field. This is only used when the element representation is not a floating point type.

**Bug** to be benchmarked.

## 17.65 fflas\_bounds.inl File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/flimits.h"
#include <givaro/udl.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_bounds\\_INL](#)
- #define [FFLAS\\_INT\\_TYPE](#) uint64\_t

### Functions

- template<class [Field](#) >  
double [computeFactorClassic](#) (const [Field](#) &F)
- template<> double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< double > &F)
- template<> double [computeFactorClassic](#) (const [Givaro::ModularBalanced](#)< float > &F)
- template<class [Field](#) >  
size\_t [DotProdBoundClassic](#) (const [Field](#) &F, const typename [Field::Element](#) &beta)
- [Givaro::Integer](#) [InfNorm](#) (const size\_t M, const size\_t N, const [Givaro::Integer](#) \*A, const size\_t lda)
- template<class [Field](#) >  
size\_t [TRSMBound](#) (const [Field](#) &)  
*TRSMBound.*
- template<class [Element](#) >  
size\_t [TRSMBound](#) (const [Givaro::Modular](#)< [Element](#) > &F)

*Specialization for positive modular representation over float.*

- template<class Element >  
size\_t [TRSMBound](#) (const [Givaro::ModularBalanced](#)< Element > &F)

*Specialization for balanced modular representation over double.*

## 17.65.1 Macro Definition Documentation

### 17.65.1.1 \_\_FFLASFFPACK\_fflas\_bounds\_INL

```
#define __FFLASFFPACK_fflas_bounds_INL
```

### 17.65.1.2 FFLAS\_INT\_TYPE

```
#define FFLAS_INT_TYPE uint64_t
```

## 17.66 fflas\_enum.h File Reference

```
#include <algorithm>
```

### Data Structures

- class [AreEqual](#)< X, Y >
- class [AreEqual](#)< X, X >

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

### Enumerations

- enum [FFLAS\\_ORDER](#) { [FflasRowMajor](#) = 101 , [FflasColMajor](#) = 102 }  
*Storage by row or col ?*
- enum [FFLAS\\_TRANSPOSE](#) { [FflasNoTrans](#) = 111 , [FflasTrans](#) = 112 }  
*Is matrix transposed ?*
- enum [FFLAS\\_UPLO](#) { [FflasUpper](#) = 121 , [FflasLower](#) = 122 }  
*Is triangular matrix's shape upper ?*
- enum [FFLAS\\_DIAG](#) { [FflasNonUnit](#) = 131 , [FflasUnit](#) = 132 }  
*Is the triangular matrix implicitly unit diagonal ?*
- enum [FFLAS\\_SIDE](#) { [FflasLeft](#) = 141 , [FflasRight](#) = 142 }  
*On what side ?*
- enum [FFLAS\\_BASE](#) { [FflasDouble](#) = 151 , [FflasFloat](#) = 152 , [FflasGeneric](#) = 153 }  
*FFLAS\_BASE determines the type of the element representation for Matrix Mult kernel.*

### Functions

- template<class T >  
const T & [min3](#) (const T &m, const T &n, const T &k)
- template<class T >  
const T & [max3](#) (const T &m, const T &n, const T &k)
- template<class T >  
const T & [min4](#) (const T &m, const T &n, const T &k, const T &l)

- `template<class T >`  
`const T & max4 (const T &m, const T &n, const T &k, const T &l)`

## 17.67 fflas\_fadd.h File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas_fadd.inl"
```

### Data Structures

- struct `support_simd_add< T >`

### Namespaces

- namespace `FFLAS`

### Functions

- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void faddin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`  
*fadd : matrix addition.*
- `template<class Field >`  
`void fsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)`

- fsub* : matrix subtraction.

```
template<class Field >
void faddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

    faddin
```
- ```
template<class Field >
void fsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

    fsubin C = C - B
```
- ```
template<class Field >
void fadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

    fadd : matrix addition with scaling.
```

## 17.68 fflas\_fadd.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::vectorised](#)
- namespace [FFLAS::details](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fadd\\_INL](#)

### Functions

- ```
template<class SimdT , class Element , bool positive>
std::enable_if< is_simd< SimdT >::value, void >::type VEC_ADD (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
```
- ```
template<bool positive, class Element , class T1 , class T2 >
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type addp (Element *T, const Element *TA, const Element *TB, size_t n, Element p, T1 min_, T2 max_)
```
- ```
template<class SimdT , class Element , bool positive>
std::enable_if< is_simd< SimdT >::value, void >::type VEC_SUB (SimdT &C, SimdT &A, SimdT &B, SimdT &Q, SimdT &T, SimdT &P, SimdT &NEGP, SimdT &MIN, SimdT &MAX)
```
- ```
template<bool positive, class Element , class T1 , class T2 >
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type subp (Element *T, const Element *TA, const Element *TB, const size_t n, const Element p, const T1 min_, const T2 max_)
```
- ```
template<class Element >
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type add (Element *T, const Element *TA, const Element *TB, size_t n)
```
- ```
template<class Element >
std::enable_if< FFLAS::support_simd_add< Element >::value, void >::type sub (Element *T, const Element *TA, const Element *TB, size_t n)
```
- ```
template<class Field , bool ADD>
std::enable_if< FFLAS::support_simd_add< typename Field::Element >::value, void >::type fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc, Field Categories::ModularTag)
```



- template<class [Field](#), bool ADD>  
std::enable\_if<![FFLAS::support\\_simd\\_add](#)< typename [Field::Element](#) >::value, void >::type [fadd](#)  
(const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [Field](#)↔  
Categories::ModularTag)
- template<class [Field](#), bool ADD>  
void [fadd](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, type-  
name [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [Field](#)↔  
Categories::GenericTag)
- template<class [Field](#), bool ADD>  
std::enable\_if<![FFLAS::support\\_simd\\_add](#)< typename [Field::Element](#) >::value, void >::type [fadd](#)  
(const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [Field](#)↔  
Categories::UnparametricTag)
- template<class [Field](#), bool ADD>  
std::enable\_if<[FFLAS::support\\_simd\\_add](#)< typename [Field::Element](#) >::value, void >::type [fadd](#)  
(const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t inca, typename [Field::ConstElement\\_ptr](#) B, const size\_t incb, typename [Field::Element\\_ptr](#) C, const size\_t incc, [Field](#)↔  
Categories::UnparametricTag)

## 17.68.1 Macro Definition Documentation

### 17.68.1.1 \_\_FFLASFFPACK\_fadd\_INL

```
#define __FFLASFFPACK_fadd_INL
```

## 17.69 fflas\_fassign.h File Reference

```
#include "fflas_fassign.inl"
```

## 17.70 fflas\_fassign.inl File Reference

```
#include <string.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <givaro/zring.h>
#include "fflas-ffpack/utils/debug.h"
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fassign\\_INL](#)

### Functions

- template<class [Field](#) >  
void [fassign](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
*fassign* :  $x \leftarrow y$ .

- `template<> void fassign (const Givaro::Modular< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< float > &F, const size_t N, const float *Y, const size_t incY, float *X, const size_t incX)`
- `template<> void fassign (const Givaro::Modular< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ModularBalanced< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<> void fassign (const Givaro::ZRing< double > &F, const size_t N, const double *Y, const size_t incY, double *X, const size_t incX)`
- `template<class Field >`  
`void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`  

$$fassign : A \leftarrow B.$$

## 17.70.1 Macro Definition Documentation

### 17.70.1.1 \_\_FFLASFFPACK\_fassign\_INL

```
#define __FFLASFFPACK_fassign_INL
```

## 17.71 fflas\_faxpy.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_faxpy\\_INL](#)

### Functions

- `template<class Field >`  
`void faxpy (const Field &F, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- `template<> void faxpy (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void faxpy (const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<class Field >`  
`void faxpy (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr X, const size_t idx, typename Field::Element_ptr Y, const size_t ldy)`  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$

### 17.71.1 Macro Definition Documentation

## 17.71.1.1 \_\_FFLASFFPACK\_faxpy\_INL

```
#define __FFLASFFPACK_faxpy_INL
```

## 17.72 fflas\_fdot.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_helpers.inl"
```

## Namespaces

- namespace [FFLAS](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fdot\\_INL](#)

## Functions

- template<class [Field](#) >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultTag &MT)
- template<class [Field](#) >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DelayedTag &MT)
- template<> Givaro::DoubleDomain::Element [fdot](#) (const Givaro::DoubleDomain &, const size\_t N, [Givaro::DoubleDomain::ConstElement\\_ptr](#) x, const size\_t incx, [Givaro::DoubleDomain::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultTag &MT)
- template<> Givaro::FloatDomain::Element [fdot](#) (const Givaro::FloatDomain &, const size\_t N, [Givaro::FloatDomain::ConstElement\\_ptr](#) x, const size\_t incx, [Givaro::FloatDomain::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultTag &MT)
- template<class [Field](#) , class T >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::ConvertTo< T > &MT)
- template<class [Field](#) >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, ModeCategories::DefaultBoundedTag &dbt)
- template<class [Field](#) >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, const ParSeqHelper::Sequential seq)
- template<class [Field](#) >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY)  
*fdot: dot product  $x^T y$ .*

## 17.72.1 Macro Definition Documentation

## 17.72.1.1 \_\_FFLASFFPACK\_fdot\_INL

```
#define __FFLASFFPACK_fdot_INL
```

## 17.73 fflas\_fgemm.inl File Reference

```
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/debug.h"
#include "fflas_fgemm/fgemm_classical.inl"
#include "fflas_fgemm/fgemm_winograd.inl"
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fgemm\\_INL](#)

### Functions

- template<class NewField , class [Field](#) , class FieldMode >  
[Field::Element\\_ptr fgemm\\_convert](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldMode > &H)
- template<class [Field](#) , class Element , class AlgoT , class ParSeqTrait >  
bool [NeedPreAddReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< [Field](#), AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class [Field](#) , class Element , class AlgoT , class ModeT , class ParSeqTrait >  
bool [NeedPreAddReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< [Field](#), AlgoT, ModeT, ParSeqTrait > &WH)
- template<class [Field](#) , class Element , class AlgoT , class ParSeqTrait >  
bool [NeedPreSubReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< [Field](#), AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class [Field](#) , class Element , class AlgoT , class ModeT , class ParSeqTrait >  
bool [NeedPreSubReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, MMHelper< [Field](#), AlgoT, ModeT, ParSeqTrait > &WH)
- template<class [Field](#) , class Element , class AlgoT , class ParSeqTrait >  
bool [NeedDoublePreAddReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< [Field](#), AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class [Field](#) , class Element , class AlgoT , class ModeT , class ParSeqTrait >  
bool [NeedDoublePreAddReduction](#) (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, Element beta, MMHelper< [Field](#), AlgoT, ModeT, ParSeqTrait > &WH)
- template<class [Field](#) , class AlgoT , class ParSeqTrait >  
void [ScalAndReduce](#) (const [Field](#) &F, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) X, const size\_t incX, const MMHelper< [Field](#), AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)
- template<class [Field](#) , class AlgoT , class ParSeqTrait >  
void [ScalAndReduce](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, const MMHelper< [Field](#), AlgoT, ModeCategories::LazyTag, ParSeqTrait > &H)

- template<class [Field](#) >  
[Field::Element\\_ptr](#) [fgemm](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >, ParSeqHelper::Sequential > &H)
- template<typename [Field](#) >  
[Field::Element\\_ptr](#) [fgemm](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const ParSeqHelper::Sequential seq)
- template<typename [Field](#) , class Cut , class Param >  
[Field::Element\\_ptr](#) [fgemm](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const ParSeqHelper::Parallel< Cut, Param > par)
- template<typename [Field](#) >  
[Field::Element\\_ptr](#) [fgemm](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*fgemm: **Field** **GE**neral **Matrix** **M**ultiply.*
- template<typename [Field](#) , class ModeT , class ParSeq >  
[Field::Element\\_ptr](#) [fgemm](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Auto, ModeT, ParSeq > &H)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [fgemm](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [fsquare](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*fsquare: **S**quares a matrix.*
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [fsquareCommon](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)
- template<> double \* [fsquare](#) (const [Givaro::ModularBalanced](#)< double > &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const double alpha, const double \*A, const size\_t lda, const double beta, double \*C, const size\_t ldc)
- template<> float \* [fsquare](#) (const [Givaro::ModularBalanced](#)< float > &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const float alpha, const float \*A, const size\_t lda, const float beta, float \*C, const size\_t ldc)
- template<> double \* [fsquare](#) (const [Givaro::Modular](#)< double > &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const double alpha, const double \*A, const size\_t lda, const double beta, double \*C, const size\_t ldc)
- template<> float \* [fsquare](#) (const [Givaro::Modular](#)< float > &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const float alpha, const float \*A, const size\_t lda, const float beta, float \*C, const size\_t ldc)

## 17.73.1 Macro Definition Documentation

### 17.73.1.1 \_\_FFLASFFPACK\_fgemm\_INL

```
#define __FFLASFFPACK_fgemm_INL
```

## 17.74 fgemm\_classical.inl File Reference

```
#include <cmath>
#include "fflas-ffpack/field/field-traits.h"
```

### Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_fflas\\_fgemm\\_classical\\_INL](#)

## 17.74.1 Macro Definition Documentation

### 17.74.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_classical\_INL

```
#define __FFLASFFPACK_fflas_fflas_fgemm_classical_INL
```

## 17.75 fgemm\_classical\_mp.inl File Reference

matrix multiplication with multiprecision input (either over  $\mathbb{Z}$  or over  $\mathbb{Z}/p\mathbb{Z}$ )

```
#include <givaro/modular-integer.h>
#include <givaro/zring.h>
#include "fflas-ffpack/field/rns-double.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas-ffpack/fflas/fflas_bounds.inl"
```

### Data Structures

- struct [MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >](#)
- struct [MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >](#)
- struct [MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >](#)

### Namespaces

- namespace [FFLAS](#)

### Macros

- [#define \\_\\_FFPACK\\_fgemm\\_classical\\_INL](#)

## Functions

- `template<typename RNS , typename ParSeqTrait >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Sequential, ParSeqTrait > > &H)`
- `template<typename RNS >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Sequential > &H)`
- `template<typename RNS , typename ParSeqTrait >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Compose< ParSeqHelper::Parallel< CuttingStrategy::RNSModulus, StrategyParameter::Threads >, ParSeqTrait > > &H)`
- `template<typename RNS , typename Cut , typename Param >`  
`FFPACK::RNSInteger< RNS >::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename FFPACK::RNSInteger< RNS >::Element alpha, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Ad, const size_t lda, typename FFPACK::RNSInteger< RNS >::ConstElement_ptr Bd, const size_t ldb, const typename FFPACK::RNSInteger< RNS >::Element beta, typename FFPACK::RNSInteger< RNS >::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Classic, ModeCategories::DefaultTag, ParSeqHelper::Parallel< Cut, Param > > &H)`
- `template<class ParSeq >`  
`Givaro::Integer * fgemm (const Givaro::ZRing< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, Givaro::Integer beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::ZRing< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<typename RNS , class ModeT >`  
`RNS::Element_ptr fgemm (const FFPACK::RNSInteger< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSInteger< RNS >, MMHelperAlgo::Winograd, ModeT, ParSeqHelper::Sequential > &H)`
- `template<typename RNS >`  
`RNS::Element_ptr fgemm (const FFPACK::RNSIntegerMod< RNS > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename RNS::Element alpha, typename RNS::ConstElement_ptr Ad, const size_t lda, typename RNS::ConstElement_ptr Bd, const size_t ldb, const typename RNS::Element beta, typename RNS::Element_ptr Cd, const size_t ldc, MMHelper< FFPACK::RNSIntegerMod< RNS >, MMHelperAlgo::Winograd > &H)`
- `Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer`

```
beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelper<
Algo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
```

- `template<class ParSeq >`  
`Givaro::Integer * fgemm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_TRANSPOSE ta,`  
`const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const Givaro::Integer alpha,`  
`const Givaro::Integer *A, const size_t lda, const Givaro::Integer *B, const size_t ldb, const Givaro::Integer`  
`beta, Givaro::Integer *C, const size_t ldc, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelper<`  
`Algo::Auto, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`
- `template<size_t K1, size_t K2, class ParSeq >`  
`Reclnt::ruint< K1 > * fgemm (const Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > > &F,`  
`const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t`  
`k, const Reclnt::ruint< K1 > alpha, const Reclnt::ruint< K1 > *A, const size_t lda, const Reclnt::ruint< K1`  
`> *B, const size_t ldb, Reclnt::ruint< K1 > beta, Reclnt::ruint< K1 > *C, const size_t ldc, MMHelper<`  
`Givaro::Modular< Reclnt::ruint< K1 >, Reclnt::ruint< K2 > >, MMHelperAlgo::Classic, ModeCategories<`  
`::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > &H)`

### 17.75.1 Detailed Description

matrix multiplication with multiprecision input (either over Z or over  $Z/pZ$ )

### 17.75.2 Macro Definition Documentation

#### 17.75.2.1 `__FFPACK_fgemm_classical_INL`

```
#define __FFPACK_fgemm_classical_INL
```

## 17.76 `fgemm_winograd.inl` File Reference

```
#include <stdint.h>
#include <givaro/modular.h>
#include <givaro/zring.h>
#include "fgemm_classical.inl"
#include "schedule_winograd.inl"
#include "schedule_winograd_acc.inl"
#include "schedule_winograd_acc_ip.inl"
#include "schedule_winograd_ip.inl"
#include "fflas-ffpack/fflas-ffpack-config.h"
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

### Macros

- `#define \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL`
- `#define NEWWINO`

### Functions

- `template<class Field >`  
`int WinogradThreshold (const Field &F)`  
*Computes the number of recursive levels to perform.*
- `template<> int WinogradThreshold (const Givaro::Modular< float > &F)`



- `template<> int WinogradThreshold (const Givaro::ModularBalanced< double > &F)`
- `template<> int WinogradThreshold (const Givaro::ModularBalanced< float > &F)`
- `template<class Field >`  
`int WinogradSteps (const Field &F, const size_t &m)`  
*Computes the number of recursive levels to perform.*
- `template<class Field , class FieldMode >`  
`void DynamicPeeling (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)`
- `template<class Field , class FieldMode >`  
`void DynamicPeeling2 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmin, const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::DelayedField::Element Cmax)`
- `template<class Field , class FieldMode >`  
`void WinogradCalc (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldMode > &H)`
- `template<class Field , class ModeT >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeT > &H)`
- `template<class Field , class ModeT , class Cut , class Param >`  
`Field::Element_ptr fgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::WinogradPar, ModeT, ParSeqHelper::Parallel< Cut, Param > > &H)`

## 17.76.1 Macro Definition Documentation

### 17.76.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL

```
#define __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL
```

### 17.76.1.2 NEWWINO

```
#define NEWWINO
```

## 17.77 matmul.doxy File Reference

## 17.78 schedule\_bini.inl File Reference

Bini implementation.

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::BLAS3](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fgemm\\_bini\\_INL](#)

### Functions

- template<class [Field](#) >  
void [Bini](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, const typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, const typename [Field::Element\\_ptr](#) C, const size\_t ldc, const size\_t kmax, const size\_t w, const FFLAS\_BASE base, const size\_t rec\_level)

### 17.78.1 Detailed Description

Bini implementation.

### 17.78.2 Macro Definition Documentation

#### 17.78.2.1 \_\_FFLASFFPACK\_fgemm\_bini\_INL

```
#define __FFLASFFPACK_fgemm_bini_INL
```

## 17.79 schedule\_winograd.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::BLAS3](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fgemm\\_winograd\\_INL](#)

### Functions

- template<class [Field](#) , class [FieldTrait](#) , class [Strat](#) , class [Param](#) >  
[Field::Element\\_ptr](#) [WinoPar](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, const typename [Field::Element\\_ptr](#) C, const size\_t ldc, [MMHelper](#)< [Field](#), [MMHelperAlgo](#)::WinogradPar, [FieldTrait](#), [ParSeqHelper](#)::Parallel< [Strat](#), [Param](#) > > &WH)
- template<class [Field](#) , class [FieldTrait](#) >  
void [Winograd](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#)

beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, [Field](#)↔  
Trait > &WH)

## 17.79.1 Macro Definition Documentation

### 17.79.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_INL

```
#define __FFLASFFPACK_fgemm_winograd_INL
```

## 17.80 schedule\_winograd\_acc.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::BLAS3](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fgemm\\_winograd\\_acc\\_INL](#)

### Functions

- template<class [Field](#) , class FieldTrait >  
void [WinogradAcc\\_3\\_23](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class [Field](#) , class FieldTrait >  
void [WinogradAcc\\_3\\_21](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::Winograd, FieldTrait > &WH)
- template<class [Field](#) , class FieldTrait >  
void [WinogradAcc\\_2\\_24](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, const typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::↔  
Winograd, FieldTrait > &WH)
- template<class [Field](#) , class FieldTrait >  
void [WinogradAcc\\_2\\_27](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, const typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, MMHelper< [Field](#), MMHelperAlgo::↔  
Winograd, FieldTrait > &WH)

## 17.80.1 Macro Definition Documentation

### 17.80.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_INL
```

## 17.81 schedule\_winograd\_acc\_ip.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::BLAS3](#)

### Macros

- `#define __FFLASFFPACK_fgemm_winograd_acc_ip_INL`

### Functions

- `template<class Field , class FieldTrait >`  
`void WinogradAcc\_LR (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element\_ptr A, const size_t lda, typename Field::Element\_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, Field↔Trait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc\_R\_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, const typename Field::Element\_ptr A, const size_t lda, typename Field::Element\_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size_t ldc, const MMHelper< Field, MMHelper↔Algo::Winograd, FieldTrait > &WH)`
- `template<class Field , class FieldTrait >`  
`void WinogradAcc\_L\_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element\_ptr A, const size_t lda, const typename Field::Element\_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size_t ldc, const MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)`

### 17.81.1 Macro Definition Documentation

#### 17.81.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_ip\_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_ip_INL
```

## 17.82 schedule\_winograd\_ip.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::BLAS3](#)

### Macros

- `#define __FFLASFFPACK_fgemm_winograd_ip_INL`

### Functions

- `template<class Field , class FieldTrait >`  
`void Winograd\_LR\_S (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t mr, const size_t nr, const size_t kr, const typename Field::Element alpha, typename Field::Element\_ptr A, const size_t lda, typename Field::Element\_ptr B, const size_t ldb, const typename Field::Element beta,`

typename [Field::Element\\_ptr](#) C, const size\_t ldc, const MMHelper< [Field](#), MMHelperAlgo::Winograd, [Field](#)↵  
Trait > &WH)

- template<class [Field](#) , class FieldTrait >  
void [Winograd\\_L\\_S](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const  
size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#)  
A, const size\_t lda, const typename [Field::Element\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#)  
beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const MMHelper< [Field](#), MMHelperAlgo::Winograd,  
FieldTrait > &WH)
- template<class [Field](#) , class FieldTrait >  
void [Winograd\\_R\\_S](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb,  
const size\_t mr, const size\_t nr, const size\_t kr, const typename [Field::Element](#) alpha, const typename  
[Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const typename  
[Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc, const MMHelper< [Field](#), MMHelper↵  
Algo::Winograd, FieldTrait > &WH)

## 17.82.1 Macro Definition Documentation

### 17.82.1.1 \_\_FFLASFFPACK\_fgemm\_winograd\_ip\_INL

```
#define __FFLASFFPACK_fgemm_winograd_ip_INL
```

## 17.83 fflas\_fgmv.inl File Reference

```
#include <givaro/zring.h>
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fgmv\\_INL](#)

### Functions

- template<typename FloatElement , class [Field](#) >  
[Field::Element\\_ptr fgmv\\_convert](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const  
size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda,  
typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename  
[Field::Element\\_ptr](#) Y, const size\_t incY)
- template<class [Field](#) >  
[Field::Element\\_ptr fgmv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t  
N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, type-  
name [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename  
[Field::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [Field](#), MMHelperAlgo::Classic, ModeCategories::↵  
ConvertTo< ElementCategories::MachineFloatTag > > &H)
- template<class [Field](#) >  
[Field::Element\\_ptr fgmv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t M, const size\_t  
N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, type-  
name [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename  
[Field::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [Field](#), MMHelperAlgo::Classic, ModeCategories::↵  
DelayedTag > &H)

- `template<class Field >`  
`Field::Element_ptr fgemv` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` X, const `size_t` incX, const typename `Field::Element` beta, typename `Field::Element_ptr` Y, const `size_t` incY, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
- `template<class Field >`  
`Field::Element_ptr fgemv` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` X, const `size_t` incX, const typename `Field::Element` beta, typename `Field::Element_ptr` Y, const `size_t` incY, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::LazyTag` > &H)
- `template<class Field >`  
`Field::Element_ptr fgemv` (const `Field` &F, const `FFLAS_TRANSPOSE` TransA, const `size_t` M, const `size_t` N, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const `size_t` lda, typename `Field::ConstElement_ptr` X, const `size_t` incX, const typename `Field::Element` beta, typename `Field::Element_ptr` Y, const `size_t` incY)  
*finite prime Field GEneral Matrix Vector multiplication.*
- `Givaro::ZRing< int64_t >::Element_ptr fgemv` (const `Givaro::ZRing< int64_t >` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const `int64_t` alpha, const `int64_t` \*A, const `size_t` lda, const `int64_t` \*X, const `size_t` incX, const `int64_t` beta, `int64_t` \*Y, const `size_t` incY, `MMHelper`< `Givaro::ZRing< int64_t >`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
- `Givaro::DoubleDomain::Element_ptr fgemv` (const `Givaro::DoubleDomain` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const `Givaro::DoubleDomain::Element` alpha, const `Givaro::DoubleDomain::ConstElement_ptr` A, const `size_t` lda, const `Givaro::DoubleDomain::ConstElement_ptr` X, const `size_t` incX, const `Givaro::DoubleDomain::Element` beta, `Givaro::DoubleDomain::Element_ptr` Y, const `size_t` incY, `MMHelper`< `Givaro::DoubleDomain`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
- `template<class Field >`  
`Field::Element_ptr fgemv` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const typename `Field::Element` alpha, const typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::ConstElement_ptr` X, const `size_t` incX, const typename `Field::Element` beta, typename `Field::Element_ptr` Y, const `size_t` incY, `MMHelper`< `Field`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultBoundedTag` > &H)
- `Givaro::FloatDomain::Element_ptr fgemv` (const `Givaro::FloatDomain` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` M, const `size_t` N, const `Givaro::FloatDomain::Element` alpha, const `Givaro::FloatDomain::ConstElement_ptr` A, const `size_t` lda, const `Givaro::FloatDomain::ConstElement_ptr` X, const `size_t` incX, const `Givaro::FloatDomain::Element` beta, `Givaro::FloatDomain::Element_ptr` Y, const `size_t` incY, `MMHelper`< `Givaro::FloatDomain`, `MMHelperAlgo::Classic`, `ModeCategories::DefaultTag` > &H)
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr fgemv` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` m, const `size_t` n, const typename `Field::Element` alpha, const typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::ConstElement_ptr` X, const `size_t` incX, const typename `Field::Element` beta, typename `Field::Element_ptr` Y, const `size_t` incY, `ParSeqHelper::Parallel`< `Cut`, `Param` > &parH)
- `template<class Field >`  
`Field::Element_ptr fgemv` (const `Field` &F, const `FFLAS_TRANSPOSE` ta, const `size_t` m, const `size_t` n, const typename `Field::Element` alpha, const typename `Field::ConstElement_ptr` A, const `size_t` lda, const typename `Field::ConstElement_ptr` X, const `size_t` incX, const typename `Field::Element` beta, typename `Field::Element_ptr` Y, const `size_t` incY, `ParSeqHelper::Sequential` &seqH)

## 17.83.1 Macro Definition Documentation

### 17.83.1.1 \_\_FFLASFFPACK\_fgemv\_INL

```
#define __FFLASFFPACK_fgemv_INL
```

## 17.84 fflas\_fgmv\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fgmv\\_mp\\_INL](#)

### Functions

- [FFPACK::rns\\_double::Element\\_ptr fgmv](#) (const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::ConstElement\\_ptr](#) X, const size\_t incX, const [FFPACK::rns\\_double::Element](#) beta, [FFPACK::rns\\_double::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
- [FFPACK::rns\\_double::Element\\_ptr fgmv](#) (const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t M, const size\_t N, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::ConstElement\\_ptr](#) X, const size\_t incX, const [FFPACK::rns\\_double::Element](#) beta, [FFPACK::rns\\_double::Element\\_ptr](#) Y, const size\_t incY, [MMHelper](#)< [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::DefaultTag](#) > &H)
- [Givaro::Integer \\* fgmv](#) (const [Givaro::ZRing](#)< [Givaro::Integer](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const [Givaro::Integer](#) alpha, [Givaro::Integer \\*A](#), const size\_t lda, [Givaro::Integer \\*X](#), const size\_t ldx, [Givaro::Integer](#) beta, [Givaro::Integer \\*Y](#), const size\_t ldy, [MMHelper](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) > > &H)
- [Givaro::Integer \\* fgmv](#) (const [Givaro::Modular](#)< [Givaro::Integer](#) > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const [Givaro::Integer](#) alpha, [Givaro::Integer \\*A](#), const size\_t lda, [Givaro::Integer \\*X](#), const size\_t ldx, [Givaro::Integer](#) beta, [Givaro::Integer \\*Y](#), const size\_t ldy, [MMHelper](#)< [Givaro::Modular](#)< [Givaro::Integer](#) >, [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) > > &H)
- [template<size\\_t K1, size\\_t K2, class ParSeq>](#)  
[Reclnt::ruint](#)< K1 > \* [fgmv](#) (const [Givaro::Modular](#)< [Reclnt::ruint](#)< K1 >, [Reclnt::ruint](#)< K2 > > &F, const [FFLAS\\_TRANSPOSE](#) ta, const size\_t m, const size\_t n, const [Reclnt::ruint](#)< K1 > alpha, const [Reclnt::ruint](#)< K1 > \*A, const size\_t lda, const [Reclnt::ruint](#)< K1 > \*X, const size\_t incx, [Reclnt::ruint](#)< K1 > beta, [Reclnt::ruint](#)< K1 > \*Y, const size\_t incy, [MMHelper](#)< [Givaro::Modular](#)< [Reclnt::ruint](#)< K1 >, [Reclnt::ruint](#)< K2 > >, [MMHelperAlgo::Classic](#), [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) >, [ParSeq](#) > &H)

### 17.84.1 Macro Definition Documentation

#### 17.84.1.1 \_\_FFLASFFPACK\_fgmv\_mp\_INL

```
#define __FFLASFFPACK_fgmv_mp_INL
```

## 17.85 fflas\_fger.inl File Reference

### Namespaces

- namespace [FFLAS](#)



- namespace [FFLAS::Protected](#)

## Macros

- `#define __FFLASFFPACK_fger_INL`

## Functions

- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)`  
*fger: rank one update of a general matrix*
- `template<class FloatElement , class Field >`  
`void fger_convert (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > > &H)`
- `template<class Field , class AnyTag >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, AnyTag > &H)`
- `void fger (const Givaro::DoubleDomain &F, const size_t M, const size_t N, const Givaro::DoubleDomain::Element alpha, const Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, const Givaro::DoubleDomain::ConstElement_ptr y, const size_t incy, Givaro::DoubleDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, const typename Field::ConstElement_ptr x, const size_t incx, const typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)`
- `void fger (const Givaro::FloatDomain &F, const size_t M, const size_t N, const Givaro::FloatDomain::Element alpha, const Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, const Givaro::FloatDomain::ConstElement_ptr y, const size_t incy, Givaro::FloatDomain::Element_ptr A, const size_t lda, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)`
- `template<class Field >`  
`void fger (const Field &F, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element_ptr A, const size_t lda, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)`

## 17.85.1 Macro Definition Documentation

### 17.85.1.1 \_\_FFLASFFPACK\_fger\_INL

```
#define __FFLASFFPACK_fger_INL
```



## 17.86 fflas\_fger\_mp.inl File Reference

```
#include <givaro/modular-integer.h>
#include <givaro/zring.h>
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas-ffpack/fflas/fflas_fgemm/fgemm_classical_mp.inl"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/field/rns-integer-mod.h"
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFPACK\\_fger\\_mp\\_INL](#)

### Functions

- void [fger](#) (const Givaro::Modular< Givaro::Integer > &F, const size\_t M, const size\_t N, const typename Givaro::Integer alpha, typename Givaro::Integer \*x, const size\_t incx, typename Givaro::Integer \*y, const size\_t incy, typename Givaro::Integer \*A, const size\_t lda, MMHelper< Givaro::Modular< Givaro::Integer >, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::RNSElementTag > > &H)
- template<typename [RNS](#) >  
void [fger](#) (const [FFPACK::RNSInteger](#)< [RNS](#) > &F, const size\_t M, const size\_t N, const typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element alpha, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr x, const size\_t incx, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr y, const size\_t incy, typename [FFPACK::RNSInteger](#)< [RNS](#) >::Element\_ptr A, const size\_t lda, MMHelper< [FFPACK::RNSInteger](#)< [RNS](#) >, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- template<typename [RNS](#) >  
void [fger](#) (const [FFPACK::RNSIntegerMod](#)< [RNS](#) > &F, const size\_t M, const size\_t N, const typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element alpha, typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element\_ptr x, const size\_t incx, typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element\_ptr y, const size\_t incy, typename [FFPACK::RNSIntegerMod](#)< [RNS](#) >::Element\_ptr A, const size\_t lda, MMHelper< [FFPACK::RNSIntegerMod](#)< [RNS](#) >, MMHelperAlgo::Classic > &H)

### 17.86.1 Macro Definition Documentation

#### 17.86.1.1 [\\_\\_FFPACK\\_fger\\_mp\\_INL](#)

```
#define \_\_FFPACK\_fger\_mp\_INL
```

## 17.87 fflas\_freduce.h File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/utils/cast.h"
#include "fflas-ffpack/fflas/fflas_freduce.inl"
```

### Data Structures

- struct [support\\_simd\\_mod](#)< T >
- struct [support\\_fast\\_mod](#)< T >

- struct `support_fast_mod< float >`
- struct `support_fast_mod< double >`
- struct `support_fast_mod< int64_t >`

## Namespaces

- namespace `FFLAS`

## Functions

- template<class `Field` >  
void `freduce` (const `Field` &F, const size\_t n, typename `Field::ConstElement_ptr` Y, const size\_t incY, typename `Field::Element_ptr` X, const size\_t incX)  
 $freduce\ x \leftarrow ymodF.$
- template<class `Field` >  
void `freduce` (const `Field` &F, const size\_t n, typename `Field::Element_ptr` X, const size\_t incX)  
 $freduce\ x \leftarrow xmodF.$
- template<class `Field` >  
void `freduce_constoverride` (const `Field` &F, const size\_t m, typename `Field::ConstElement_ptr` A, const size\_t incX)
- template<class `Field`, class `ConstOtherElement_ptr` >  
void `finit` (const `Field` &F, const size\_t n, `ConstOtherElement_ptr` Y, const size\_t incY, typename `Field::Element_ptr` X, const size\_t incX)
- template<class `Field` >  
void `finit` (const `Field` &F, const size\_t n, typename `Field::Element_ptr` X, const size\_t incX)  
 $finit\ Initializes\ X\ in\ F\$.$
- template<class `Field` >  
void `freduce` (const `Field` &F, const size\_t m, const size\_t n, typename `Field::Element_ptr` A, const size\_t lda)  
 $freduce\ A \leftarrow AmodF.$
- template<class `Field` >  
void `pfreduce` (const `Field` &F, const size\_t m, const size\_t n, typename `Field::Element_ptr` A, const size\_t lda, const size\_t numths)
- template<class `Field` >  
void `freduce` (const `Field` &F, const size\_t m, const size\_t n, typename `Field::ConstElement_ptr` B, const size\_t ldb, typename `Field::Element_ptr` A, const size\_t lda)  
 $freduce\ A \leftarrow BmodF.$
- template<class `Field` >  
void `freduce_constoverride` (const `Field` &F, const size\_t m, const size\_t n, typename `Field::ConstElement_ptr` A, const size\_t lda)
- template<class `Field`, class `OtherElement_ptr` >  
void `finit` (const `Field` &F, const size\_t m, const size\_t n, const `OtherElement_ptr` B, const size\_t ldb, typename `Field::Element_ptr` A, const size\_t lda)  
 $finit\ A \leftarrow BmodF.$
- template<class `Field` >  
void `finit` (const `Field` &F, const size\_t m, const size\_t n, typename `Field::Element_ptr` A, const size\_t lda)

## 17.88 fflas\_freduce.inl File Reference

```
#include <givaro/udl.h>
#include "fflas-ffpack/fflas/fflas_fassign.h"
```

## Data Structures

- struct [HelperMod](#)< Field, ElementCategories::MachineIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::MachineFloatTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >
- struct [HelperMod](#)< Field, FFLAS::ElementCategories::FixedPrecIntTag >

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::vectorised](#)
- namespace [FFLAS::vectorised::unswitch](#)
- namespace [FFLAS::details](#)

## Macros

- `#define \_\_FFLASFFPACK\_fflas\_freduce\_INL`
- `#define FFLASFFPACK\_COPY\_REDUCE 32 /* TODO TO BENCHMARK LATER */`

## Functions

- `template<class T >`  
`std::enable_if<!std::is_integral< T >::value, T >::type reduce (T A, T B)`
- `template<class T >`  
`std::enable_if< std::is_integral< T >::value, T >::type reduce (T A, T B)`
- `template<> Givaro::Integer reduce (Givaro::Integer A, Givaro::Integer B)`
- `float reduce (float A, float B, float invB, float min, float max)`
- `double reduce (double A, double B, double invB, double min, double max)`
- `int64_t reduce (int64_t A, int64_t p, double invp, double min, double max, int64_t pow50rem)`
- `template<class Field >`  
`Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::MachineIntTag > &H)`
- `template<class Field >`  
`Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::MachineFloatTag > &H)`
- `template<class Field >`  
`Field::Element reduce (typename Field::Element A, HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > &H)`
- `template<class Field >`  
`std::enable_if< !FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, typename Field::Element_ptr T, HelperMod< Field > &H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, const size_t &incX, typename Field::Element_ptr T, HelperMod< Field > &H)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, typename Field::Element_ptr T)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type modp (const Field &F, typename Field::ConstElement_ptr U, const size_t &n, const size_t &incX, typename Field::Element_ptr T)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce (const Field &F, const size_t m, typename Field::Element_ptr A, const size_t incX, FieldCategories::ModularTag)`

- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type freduce`  
`(const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, typename`  
`Field::Element_ptr A, const size_t incX, FieldCategories::ModularTag)`
- `template<class Field, class FC >`  
`void freduce (const Field &F, const size_t m, typename Field::Element_ptr A, const size_t incX, FC)`
- `template<class Field, class FC >`  
`void freduce (const Field &F, const size_t m, typename Field::ConstElement_ptr B, const size_t incY, type-`  
`name Field::Element_ptr A, const size_t incX, FC)`

## 17.88.1 Macro Definition Documentation

### 17.88.1.1 \_\_FFLASFFPACK\_fflas\_freduce\_INL

```
#define __FFLASFFPACK_fflas_freduce_INL
```

### 17.88.1.2 FFLASFFPACK\_COPY\_REDUCE

```
#define FFLASFFPACK_COPY_REDUCE 32 /* TODO TO BENCHMARK LATER */
```

## 17.89 fflas\_freduce\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_freduce_mp_INL`

### Functions

- `template<> void freduce (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t n,`  
`FFPACK::RNSIntegerMod< FFPACK::rns_double >::Element_ptr A, size_t inc)`
- `template<> void freduce (const FFPACK::RNSIntegerMod< FFPACK::rns_double > &F, const size_t m,`  
`const size_t n, FFPACK::rns_double::Element_ptr A, size_t lda)`

## 17.89.1 Macro Definition Documentation

### 17.89.1.1 \_\_FFLASFFPACK\_fflas\_freduce\_mp\_INL

```
#define __FFLASFFPACK_fflas_freduce_mp_INL
```

## 17.90 fflas\_freivalds.inl File Reference

### Namespaces

- namespace [FFLAS](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_freivalds\\_INL](#)

## Functions

- template<class [Field](#) >  
bool [freivalds](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::ConstElement\\_ptr](#) C, const size\_t ldc)  
*freivalds: Freivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.*

## 17.90.1 Macro Definition Documentation

### 17.90.1.1 [\\_\\_FFLASFFPACK\\_freivalds\\_INL](#)

```
#define __FFLASFFPACK_freivalds_INL
```

## 17.91 fflas\_fscal.h File Reference

```
#include "fflas_fscal.inl"
```

## 17.92 fflas\_fscal.inl File Reference

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::vectorised](#)
- namespace [FFLAS::vectorised::unswitch](#)
- namespace [FFLAS::details](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fscal\\_INL](#)

## Functions

- template<class [Field](#) >  
std::enable\_if<![FFLAS::support\\_simd\\_mod](#)< typename [Field::Element](#) >::value &&[FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [scalp](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) T, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) U, const size\_t n, HelperMod< [Field](#) > &H)
- template<class [Field](#) >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [scalp](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) T, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) U, const size\_t n, const size\_t &incX, HelperMod< [Field](#) > &H)
- template<class [Field](#) >  
std::enable\_if< [FFLAS::support\\_fast\\_mod](#)< typename [Field::Element](#) >::value, void >::type [scalp](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) T, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) U, const size\_t n)

- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type scalp`  
`(const Field &F, typename Field::Element_ptr T, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr U, const size_t n, const size_t &incX)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscaln`  
`(const Field &F, const size_t N, const typename Field::Element a, typename Field::Element_ptr X, const size_t incX,`  
`FieldCategories::ModularTag)`
- `template<class Field >`  
`std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type fscal`  
`(const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr X, const size_t incX,`  
`typename Field::Element_ptr Y, const size_t incY, FieldCategories::ModularTag)`
- `template<class Field, class FC >`  
`void fscaln`  
`(const Field &F, const size_t n, const typename Field::Element a, typename Field::Element_ptr X,`  
`const size_t incX, FC)`
- `template<class Field, class FC >`  
`void fscal`  
`(const Field &F, const size_t N, const typename Field::Element a, typename Field::ConstElement_ptr`  
`X, const size_t incX, typename Field::Element_ptr Y, const size_t incY, FC)`
- `template<class Field >`  
`void fscaln`  
`(const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr`  
`X, const size_t incX)`  

$$fscaln\ x \leftarrow \alpha \cdot x.$$
- `template<class Field >`  
`void fscal`  
`(const Field &F, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr`  
`X, const size_t incX, typename Field::Element_ptr Y, const size_t incY)`  

$$fscal\ y \leftarrow \alpha \cdot x.$$
- `template<> void fscal`  
`(const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a,`  
`Givaro::DoubleDomain::ConstElement_ptr x, const size_t incx, Givaro::DoubleDomain::Element_ptr`  
`y, const size_t incy)`
- `template<> void fscal`  
`(const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element`  
`a, Givaro::FloatDomain::ConstElement_ptr x, const size_t incx, Givaro::FloatDomain::Element_ptr y, const`  
`size_t incy)`
- `template<> void fscaln`  
`(const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a,`  
`Givaro::DoubleDomain::Element_ptr y, const size_t incy)`
- `template<> void fscaln`  
`(const Givaro::FloatDomain &, const size_t N, const Givaro::FloatDomain::Element`  
`a, Givaro::FloatDomain::Element_ptr y, const size_t incy)`
- `template<class Field >`  
`void fscaln`  
`(const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename`  
`Field::Element_ptr A, const size_t lda)`  

$$fscaln\ A \leftarrow a \cdot A.$$
- `template<class Field >`  
`void fscal`  
`(const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, typename`  
`Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  

$$fscal\ B \leftarrow a \cdot A.$$

## 17.92.1 Macro Definition Documentation

### 17.92.1.1 \_\_FFLASFFPACK\_fscal\_INL

```
#define __FFLASFFPACK_fscal_INL
```

## 17.93 fflas\_fscal\_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas_fscal.h"
#include "fflas_fgemm.inl"
#include "fflas-ffpack/fflas/fflas_freduce_mp.inl"
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fscal\\_mp\\_INL](#)

### Functions

- template<> void [fscal](#)(const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::Element\\_ptr](#) A, const size\_t inc)
- template<> void [fscal](#)(const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t Ainc, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t Binc)
- template<> void [fscal](#)(const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::Element\\_ptr](#) A, const size\_t lda)
- template<> void [fscal](#)(const [FFPACK::RNSInteger](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t ldb)
- template<> void [fscal](#)(const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, const typename [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >::Element alpha, typename [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) >::Element\_ptr A, const size\_t inc)
- template<> void [fscal](#)(const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t Ainc, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t Binc)
- template<> void [fscal](#)(const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::Element\\_ptr](#) A, const size\_t lda)
- template<> void [fscal](#)(const [FFPACK::RNSIntegerMod](#)< [FFPACK::rns\\_double](#) > &F, const size\_t m, const size\_t n, const [FFPACK::rns\\_double::Element](#) alpha, [FFPACK::rns\\_double::ConstElement\\_ptr](#) A, const size\_t lda, [FFPACK::rns\\_double::Element\\_ptr](#) B, const size\_t ldb)

### 17.93.1 Macro Definition Documentation

#### 17.93.1.1 \_\_FFLASFFPACK\_fscal\_mp\_INL

```
#define __FFLASFFPACK_fscal_mp_INL
```

## 17.94 fflas\_fsyr2k.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_fsyr2k\\_INL](#)

## Functions

- `template<class Field >`  
`Field::Element_ptr fsyr2k` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` B, const size\_t ldb, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc)  
*fsyr2k: Symmetric Rank 2K update*

## 17.94.1 Macro Definition Documentation

### 17.94.1.1 \_\_FFLASFFPACK\_fflas\_fsyr2k\_INL

```
#define __FFLASFFPACK_fflas_fsyr2k_INL
```

## 17.95 fflas\_fsyrk.inl File Reference

### Namespaces

- namespace `FFLAS`

### Macros

- #define `__FFLASFFPACK_fflas_fsyrk_INL`

## Functions

- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::ConstElement_ptr` A, const size\_t lda, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc)  
*fsyrk: Symmetric Rank K update*
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const size\_t threshold=\_\_FFLASFFPACK\_FSYRK\_THRESHOLD)  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const ParSeqHelper::Sequential seq, const size\_t threshold)
- `template<class Field , class Cut , class Param >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size\_t threshold)
- `template<class Field >`  
`Field::Element_ptr fsyrk` (const `Field` &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename `Field::Element` alpha, typename `Field::Element_ptr` A, const size\_t lda, typename `Field::ConstElement_ptr` D, const size\_t incD, const std::vector< bool > &two←Block, const typename `Field::Element` beta, typename `Field::Element_ptr` C, const size\_t ldc, const size\_t threshold=\_\_FFLASFFPACK\_FSYRK\_THRESHOLD)  
*fsyrk: Symmetric Rank K update with diagonal scaling*



## 17.95.1 Macro Definition Documentation

### 17.95.1.1 \_\_FFLASFFPACK\_fflas\_fsyrrk\_INL

```
#define __FFLASFFPACK_fflas_fsyrrk_INL
```

## 17.96 fflas\_ftrmm.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ftrmm\\_INL](#)

### Functions

- template<class [Field](#) >  
void [ftrmm](#) (const [Field](#) &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)  
*ftrmm: **TR**angular **M**atrix **M**ultiply.*
- template<class [Field](#) >  
void [ftrmm](#) (const [Field](#) &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
*ftrmm: **TR**angular **M**atrix **M**ultiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$  or  $C \leftarrow \alpha B \text{op}(A) + \text{beta}C$ .*

## 17.96.1 Macro Definition Documentation

### 17.96.1.1 \_\_FFLASFFPACK\_ftrmm\_INL

```
#define __FFLASFFPACK_ftrmm_INL
```

## 17.97 fflas\_ftrsm.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ftrsm\\_INL](#)

### Functions

- template<class [Field](#) >  
void [ftrsm](#) (const [Field](#) &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)

- `template<class Field >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Sequential &PSH)`
- `template<class Field, class Cut, class Param >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const ParSeqHelper::Parallel< Cut, Param > &PSH)`
- `template<class Field, class ParSeqTrait = ParSeqHelper::Sequential>`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Recursive, ParSeqTrait > &H)`

## 17.97.1 Macro Definition Documentation

### 17.97.1.1 \_\_FFLASFFPACK\_ftrsm\_INL

```
#define __FFLASFFPACK_ftrsm_INL
```

## 17.98 fflas\_ftrsm\_mp.inl File Reference

triangular system with matrix right hand side over multiprecision domain (either over  $\mathbb{Z}$  or over  $\mathbb{Z}/p\mathbb{Z}$ )

```
#include <cmath>
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include "fflas-ffpack/fflas/fflas_bounds.inl"
#include "fflas-ffpack/fflas/fflas_level3.inl"
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
```

## Namespaces

- namespace [FFLAS](#)

## Macros

- `#define __FFPACK_ftrsm_mp_INL`

## Functions

- `void ftrsm (const Givaro::Modular< Givaro::Integer > &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const Givaro::Integer alpha, const Givaro::Integer *A, const size_t lda, Givaro::Integer *B, const size_t ldb)`
- `void cblas_impftrsm (const enum FFLAS_ORDER Order, const enum FFLAS_SIDE Side, const enum FFLAS_UPLO Uplo, const enum FFLAS_TRANSPOSE TransA, const enum FFLAS_DIAG Diag, const int M, const int N, const FFPACK::rns_double_elt alpha, FFPACK::rns_double_elt_cstptr A, const int lda, FFPACK::rns_double_elt_ptr B, const int ldb)`

### 17.98.1 Detailed Description

triangular system with matrix right hand side over multiprecision domain (either over  $\mathbb{Z}$  or over  $\mathbb{Z}/p\mathbb{Z}$ )

## 17.98.2 Macro Definition Documentation

### 17.98.2.1 \_\_FFPACK\_ftrsm\_mp\_INL

```
#define __FFPACK_ftrsm_mp_INL
```

## 17.99 fflas\_ftrsv.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ftrsv\\_INL](#)

### Functions

- template<class [Field](#) >  
void [ftrsv](#) (const [Field](#) &F, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, int incX)  
*ftrsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*

## 17.99.1 Macro Definition Documentation

### 17.99.1.1 \_\_FFLASFFPACK\_ftrsv\_INL

```
#define __FFLASFFPACK_ftrsv_INL
```

## 17.100 fflas\_helpers.inl File Reference

```
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/utils/flimits.h"
#include <algorithm>
```

### Data Structures

- struct [Auto](#)
- struct [Classic](#)
- struct [Winograd](#)
- struct [WinogradPar](#)
- struct [Bini](#)
- struct [AlgoChooser](#)< [ModeT](#), [ParSeq](#) >
- struct [AlgoChooser](#)< [ModeCategories::ConvertTo](#)< [ElementCategories::RNSElementTag](#) >, [ParSeq](#) >
- struct [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::DefaultTag](#), [ParSeqTrait](#) >  
*FGEMM Helper for Default and ConvertTo modes of operation.*
- struct [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeCategories::ConvertTo](#)< [Dest](#) >, [ParSeqTrait](#) >
- struct [MMHelper](#)< [Field](#), [AlgoTrait](#), [ModeTrait](#), [ParSeqTrait](#) >
- struct [Recursive](#)

- struct [Iterative](#)
- struct [Hybrid](#)
- struct [TRSMHelper](#)< [ReclterTrait](#), [ParSeqTrait](#) >

*TRSM Helper.*

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)
- namespace [FFLAS::MMHelperAlgo](#)
- namespace [FFLAS::StructureHelper](#)

*StructureHelper for ftrsm.*

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_mmhelper\\_INL](#)

## Functions

- template<class [Field](#) >  
int [WinogradSteps](#) (const [Field](#) &F, const size\_t &m)  
*Computes the number of recursive levels to perform.*
- template<class DFE >  
size\_t [min\\_types](#) (const DFE &k)
- template<> size\_t [min\\_types](#) (const [Reclnt::rint](#)< 6 > &k)
- template<> size\_t [min\\_types](#) (const [Reclnt::rint](#)< 7 > &k)
- template<> size\_t [min\\_types](#) (const [Reclnt::rint](#)< 8 > &k)
- template<> size\_t [min\\_types](#) (const [Reclnt::rint](#)< 9 > &k)
- template<> size\_t [min\\_types](#) (const [Reclnt::rint](#)< 10 > &k)
- template<> size\_t [min\\_types](#) (const [Givaro::Integer](#) &k)
- template<class T >  
bool [unfit](#) (T x)
- template<> bool [unfit](#) ([int64\\_t](#) x)
- template<size\_t K>  
bool [unfit](#) ([Reclnt::rint](#)< K > x)
- template<> bool [unfit](#) ([Reclnt::rint](#)< 6 > x)

## 17.100.1 Macro Definition Documentation

### 17.100.1.1 [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_mmhelper\\_INL](#)

```
#define __FFLASFFPACK_fflas_fflas_mmhelper_INL
```

## 17.101 igemm.doxy File Reference

## 17.102 igemm.h File Reference

```
#include "igemm_kernels.h"
#include "igemm_tools.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "igemm.inl"
```

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

## Enumerations

- enum [number\\_kind](#) { [zero](#) =0 , [one](#) =1 , [mone](#) =-1 , [other](#) =2 }

## Functions

- template<enum FFLAS\_TRANSPOSE tA, enum FFLAS\_TRANSPOSE tB>  
void [igemm\\_colmajor](#) (size\_t rows, size\_t cols, size\_t depth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*A, size\_t lda, const [int64\\_t](#) \*B, size\_t ldb, [int64\\_t](#) \*C, size\_t ldc)
- template<enum FFLAS\_TRANSPOSE tA, enum FFLAS\_TRANSPOSE tB, enum number\_kind alpha\_kind>  
void [igemm\\_colmajor](#) (size\_t rows, size\_t cols, size\_t depth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*A, size\_t lda, const [int64\\_t](#) \*B, size\_t ldb, [int64\\_t](#) \*C, size\_t ldc)
- void [igemm](#) (const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, size\_t rows, size\_t cols, size\_t depth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*A, size\_t lda, const [int64\\_t](#) \*B, size\_t ldb, const [int64\\_t](#) beta, [int64\\_t](#) \*C, size\_t ldc)
- void [igemm\\_](#) (const enum FFLAS\_ORDER Order, const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, const size\_t M, const size\_t N, const size\_t K, const [int64\\_t](#) alpha, const [int64\\_t](#) \*A, const size\_t lda, const [int64\\_t](#) \*B, const size\_t ldb, const [int64\\_t](#) beta, [int64\\_t](#) \*C, const size\_t ldc)

## 17.103 igemm.inl File Reference

```
#include "fflas-ffpack/utils/fflas_memory.h"
```

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::Protected](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_igemm\\_igemm\\_INL](#)

## Functions

- template<enum FFLAS\_TRANSPOSE tA, enum FFLAS\_TRANSPOSE tB>  
void [igemm\\_colmajor](#) (size\_t rows, size\_t cols, size\_t depth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*A, size\_t lda, const [int64\\_t](#) \*B, size\_t ldb, [int64\\_t](#) \*C, size\_t ldc)
- template<enum FFLAS\_TRANSPOSE tA, enum FFLAS\_TRANSPOSE tB, enum number\_kind alpha\_kind>  
void [igemm\\_colmajor](#) (size\_t rows, size\_t cols, size\_t depth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*A, size\_t lda, const [int64\\_t](#) \*B, size\_t ldb, [int64\\_t](#) \*C, size\_t ldc)
- void [igemm](#) (const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, size\_t rows, size\_t cols, size\_t depth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*A, size\_t lda, const [int64\\_t](#) \*B, size\_t ldb, const [int64\\_t](#) beta, [int64\\_t](#) \*C, size\_t ldc)
- void [igemm\\_](#) (const enum FFLAS\_ORDER Order, const enum FFLAS\_TRANSPOSE TransA, const enum FFLAS\_TRANSPOSE TransB, const size\_t M, const size\_t N, const size\_t K, const [int64\\_t](#) alpha, const [int64\\_t](#) \*A, const size\_t lda, const [int64\\_t](#) \*B, const size\_t ldb, const [int64\\_t](#) beta, [int64\\_t](#) \*C, const size\_t ldc)

### 17.103.1 Macro Definition Documentation

### 17.103.1.1 \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_INL
```

## 17.104 igemm\_kernels.h File Reference

```
#include "igemm_kernels.inl"
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::details](#)

### Functions

- template<enum number\_kind K>  
void [igebb44](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum number\_kind K>  
void [igebb24](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum number\_kind K>  
void [igebb14](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum number\_kind K>  
void [igebb41](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum number\_kind K>  
void [igebb21](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum number\_kind K>  
void [igebb11](#) (size\_t i, size\_t j, size\_t depth, size\_t pdeth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blA, const [int64\\_t](#) \*blB, [int64\\_t](#) \*C, size\_t ldc)
- template<enum number\_kind K>  
void [igebp](#) (size\_t rows, size\_t cols, size\_t depth, const [int64\\_t](#) alpha, const [int64\\_t](#) \*blockA, size\_t lda, const [int64\\_t](#) \*blockB, size\_t ldb, [int64\\_t](#) \*C, size\_t ldc)

## 17.105 igemm\_kernels.inl File Reference

```
#include "fflas-ffpack/utils/fflas_memory.h"
#include "igemm_tools.h"
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::details](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_igemm\\_igemm\\_kernels\\_INL](#)

## Functions

- `template<enum number_kind K>`  
`void igebb44 (size_t i, size_t j, size_t depth, size_t pdeth, const int64\_t alpha, const int64\_t *blA, const int64\_t *blB, int64\_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb24 (size_t i, size_t j, size_t depth, size_t pdeth, const int64\_t alpha, const int64\_t *blA, const int64\_t *blB, int64\_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb14 (size_t i, size_t j, size_t depth, size_t pdeth, const int64\_t alpha, const int64\_t *blA, const int64\_t *blB, int64\_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb41 (size_t i, size_t j, size_t depth, size_t pdeth, const int64\_t alpha, const int64\_t *blA, const int64\_t *blB, int64\_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb21 (size_t i, size_t j, size_t depth, size_t pdeth, const int64\_t alpha, const int64\_t *blA, const int64\_t *blB, int64\_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebb11 (size_t i, size_t j, size_t depth, size_t pdeth, const int64\_t alpha, const int64\_t *blA, const int64\_t *blB, int64\_t *C, size_t ldc)`
- `template<enum number_kind K>`  
`void igebp (size_t rows, size_t cols, size_t depth, const int64\_t alpha, const int64\_t *blockA, size_t lda, const int64\_t *blockB, size_t ldb, int64\_t *C, size_t ldc)`

### 17.105.1 Macro Definition Documentation

#### 17.105.1.1 `__FFLASFFPACK_fflas_igemm_igemm_kernels_INL`

```
#define __FFLASFFPACK_fflas_igemm_igemm_kernels_INL
```

## 17.106 igemm\_tools.h File Reference

```
#include "igemm_tools.inl"
```

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::details](#)

## Functions

- `template<size_t k, bool transpose>`  
`void pack\_lhs (int64\_t *XX, const int64\_t *X, size_t ldx, size_t rows, size_t cols)`
- `template<size_t k, bool transpose>`  
`void pack\_rhs (int64\_t *XX, const int64\_t *X, size_t ldx, size_t rows, size_t cols)`
- `void gebp (size_t rows, size_t cols, size_t depth, int64\_t *C, size_t ldc, const int64\_t *blockA, size_t lda, const int64\_t *BlockB, size_t ldb, int64\_t *BlockW)`
- `void BlockingFactor (size_t &m, size_t &n, size_t &k)`

### 17.107 igemm\_tools.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd.h"
```

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::details](#)

## Macros

- `#define __FFLASFFPACK_fflas_igemm_igemm_tools_INL`

## Functions

- `template<size_t k, bool transpose>`  
void [pack\\_rhs](#) ([int64\\_t](#) \*XX, const [int64\\_t](#) \*X, size\_t ldx, size\_t rows, size\_t cols)
- `template<size_t k, bool transpose>`  
void [pack\\_lhs](#) ([int64\\_t](#) \*XX, const [int64\\_t](#) \*X, size\_t ldx, size\_t rows, size\_t cols)
- void [BlockingFactor](#) (size\_t &m, size\_t &n, size\_t &k)

### 17.107.1 Macro Definition Documentation

#### 17.107.1.1 \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_tools_INL
```

## 17.108 fflas\_level1.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- `#define __FFLASFFPACK_fflas_fflas_level1_INL`

### Functions

- `template<class Field >`  
void [freduce](#) (const [Field](#) &F, const size\_t n, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
 $freduce\ x \leftarrow x mod F.$
- `template<class Field >`  
void [freduce](#) (const [Field](#) &F, const size\_t n, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
 $freduce\ x \leftarrow y mod F.$
- `template<class Field, class OtherElement_ptr >`  
void [finit](#) (const [Field](#) &F, const size\_t n, const OtherElement\_ptr Y, const size\_t incY, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
 $finit\ x \leftarrow y mod F.$
- `template<class Field >`  
void [finit](#) (const [Field](#) &F, const size\_t n, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
 $finit\ \text{Initializes } X \text{ in } F\$.$
- `template<class Field, class OtherElement_ptr >`  
void [fconvert](#) (const [Field](#) &F, const size\_t n, OtherElement\_ptr X, const size\_t incX, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY)  
 $fconvert\ x \leftarrow y mod F.$
- `template<class Field >`  
void [fnegin](#) (const [Field](#) &F, const size\_t n, typename [Field::Element\\_ptr](#) X, const size\_t incX)



- $$fnegin\ x \leftarrow -x.$$
  - template<class [Field](#) >  
void [fneg](#) (const [Field](#) &F, const size\_t n, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY, typename [Field::Element\\_ptr](#) X, const size\_t incX)  

$$fneg\ x \leftarrow -y.$$
- template<class [Field](#) >  
void [fzero](#) (const [Field](#) &F, const size\_t n, typename [Field::Element\\_ptr](#) X, const size\_t incX)  

$$fzero : A \leftarrow 0.$$
- template<class [Field](#) , class RandIter >  
void [frand](#) (const [Field](#) &F, RandIter &G, const size\_t n, typename [Field::Element\\_ptr](#) X, const size\_t incX)  

$$frand : A \leftarrow random.$$
- template<class [Field](#) >  
bool [fiszero](#) (const [Field](#) &F, const size\_t n, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX)  

$$fiszero : test\ X = 0.$$
- template<class [Field](#) >  
bool [fequal](#) (const [Field](#) &F, const size\_t n, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY)  

$$fequal : test\ X = Y.$$
- template<class [Field](#) >  
void [fassign](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY, typename [Field::Element\\_ptr](#) X, const size\_t incX)  

$$fassign : x \leftarrow y.$$
- template<class [Field](#) >  
void [fscal](#) (const [Field](#) &F, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) X, const size\_t incX)  

$$fscal\ x \leftarrow \alpha \cdot x.$$
- template<class [Field](#) >  
void [fscal](#) (const [Field](#) &F, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  

$$fscal\ y \leftarrow \alpha \cdot x.$$
- template<class [Field](#) >  
void [faxpy](#) (const [Field](#) &F, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  

$$faxpy : y \leftarrow \alpha \cdot x + y.$$
- template<class [Field](#) >  
void [faxpby](#) (const [Field](#) &F, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  

$$faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$$
- template<class [Field](#) >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY)  

$$fdot: dot\ product\ x^T y.$$
- template<class [Field](#) >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, const ParSeqHelper::Sequential seq)
- template<typename [Field](#) , class Cut , class Param >  
[Field::Element](#) [fdot](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::ConstElement\\_ptr](#) Y, const size\_t incY, const ParSeqHelper::Parallel< Cut, Param > par)
- template<class [Field](#) >  
void [fswap](#) (const [Field](#) &F, const size\_t N, typename [Field::Element\\_ptr](#) X, const size\_t incX, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  

$$fswap: X \leftrightarrow Y.$$

- `template<class Field >`  
`void pfadd (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfsub (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, const size_t numths)`
- `template<class Field >`  
`void pfaddin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void pfsubin (const Field &F, const size_t M, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc, size_t numths)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsub (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void faddin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fsubin (const Field &F, const size_t N, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`
- `template<class Field >`  
`void fadd (const Field &F, const size_t N, typename Field::ConstElement_ptr A, const size_t inca, const typename Field::Element alpha, typename Field::ConstElement_ptr B, const size_t incb, typename Field::Element_ptr C, const size_t incc)`

## 17.108.1 Macro Definition Documentation

### 17.108.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL

```
#define __FFLASFFPACK_fflas_fflas_level1_INL
```

## 17.109 fflas\_level2.inl File Reference

```
#include "givaro/zring.h"
```

## Namespaces

- namespace [FFLAS](#)

## Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_level2\\_INL](#)

## Functions

- `template<class Field >`  
`void fassign (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr A, const size_t lda)`

- $fassign : A \leftarrow B.$
- template<class [Field](#) >  
 void [fzero](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $fzero : A \leftarrow 0.$
- template<class [Field](#) , class RandIter >  
 void [frand](#) (const [Field](#) &F, RandIter &G, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $frand : A \leftarrow random.$
- template<class [Field](#) >  
 bool [fequal](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)  
 $fequal : test A = B.$
- template<class [Field](#) >  
 bool [fiszero](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)  
 $fiszero : test A = 0.$
- template<class [Field](#) >  
 void [fidentity](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) &d)  
 $creates a diagonal matrix$
- template<class [Field](#) >  
 void [fidentity](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $creates a diagonal matrix$
- template<class [Field](#) >  
 void [freduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $freduce A \leftarrow A mod F.$
- template<class [Field](#) >  
 void [freduce](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $freduce A \leftarrow B mod F.$
- template<class [Field](#) , class OtherElement\_ptr >  
 void [finit](#) (const [Field](#) &F, const size\_t m, const size\_t n, const OtherElement\_ptr B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $finit A \leftarrow B mod F.$
- template<class [Field](#) , class OtherElement\_ptr >  
 void [finit](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $finit Initializes A in F\$.$
- template<class [Field](#) , class OtherElement\_ptr >  
 void [fconvert](#) (const [Field](#) &F, const size\_t m, const size\_t n, OtherElement\_ptr A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb)  
 $fconvert A \leftarrow B mod F.$
- template<class [Field](#) >  
 void [fnegin](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $fnegin A \leftarrow -A.$
- template<class [Field](#) >  
 void [fneg](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $fneg A \leftarrow -B.$
- template<class [Field](#) >  
 void [fscaln](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $fscaln A \leftarrow a \cdot A.$
- template<class [Field](#) >  
 void [fscal](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)

- $fscal\ B \leftarrow a \cdot A.$
- template<class [Field](#) >  
void [faxpy](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t ldx, typename [Field::Element\\_ptr](#) Y, const size\_t ldy)  
 $faxpy : y \leftarrow \alpha \cdot x + y.$
  - template<class [Field](#) >  
void [faxpby](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) X, const size\_t ldx, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t ldy)  
 $faxpby : y \leftarrow \alpha \cdot x + \beta \cdot y.$
  - template<class [Field](#) >  
void [fmove](#) (const [Field](#) &F, const size\_t m, const size\_t n, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb)  
 $fmove : A \leftarrow B \text{ and } B \leftarrow 0.$
  - template<class [Field](#) >  
void [fadd](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
 $fadd : \text{matrix addition.}$
  - template<class [Field](#) >  
void [fsub](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
 $fsub : \text{matrix subtraction.}$
  - template<class [Field](#) >  
void [fsubin](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
 $fsubin\ C = C - B$
  - template<class [Field](#) >  
void [fadd](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
 $fadd : \text{matrix addition with scaling.}$
  - template<class [Field](#) >  
void [faddin](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, typename [Field::Element\\_ptr](#) C, const size\_t ldc)  
 $faddin$
  - template<class [Field](#) >  
[Field::Element\\_ptr](#) [fgemv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE TransA, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY)  
 $fprime\ \text{Field GEneral Matrix Vector multiplication.}$
  - template<class [Field](#) >  
void [fger](#) (const [Field](#) &F, const size\_t M, const size\_t N, const typename [Field::Element](#) alpha, typename [Field::ConstElement\\_ptr](#) x, const size\_t incx, typename [Field::ConstElement\\_ptr](#) y, const size\_t incy, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
 $fger : \text{rank one update of a general matrix}$
  - template<class [Field](#) >  
void [ftrsv](#) (const [Field](#) &F, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, int incX)  
 $ftrsv : \text{TRiangular System solve with Vector Computes } X \leftarrow op(A^{-1})X$
  - template<class [Field](#) >  
size\_t [bitsize](#) (const [Field](#) &F, size\_t M, size\_t N, const typename [Field::ConstElement\\_ptr](#) A, size\_t lda)  
 $bitsize : \text{Computes the largest bitsize of the matrix' coefficients.}$

- `template<> size_t bitsize< Givaro::ZRing< Givaro::Integer > > (const Givaro::ZRing< Givaro::Integer > &F, size_t M, size_t N, const Givaro::Integer *A, size_t lda)`
- `template<class Field >`  
`void ftrmv (const Field &F, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr X, int incX)`

*ftrsm: TRIangular Matrix Vector prodcut Computes  $X \leftarrow \text{op}(A)X$*

## 17.109.1 Macro Definition Documentation

### 17.109.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL

```
#define __FFLASFFPACK_fflas_fflas_level2_INL
```

## 17.110 fflas\_level3.inl File Reference

```
#include "fflas_bounds.inl"
#include "fflas_helpers.inl"
#include "fflas-ffpack/paladin/parallel.h"
```

## Namespaces

- namespace `FFLAS`
- namespace `FFLAS::Protected`

## Macros

- `#define __FFLASFFPACK_fflas_fflas_level3_INL`
- `#define __FFLAS__TRSM_READONLY`

## Functions

- `template<class Field >`  
`void MatF2MatD_Triangular (const Field &F, Givaro::DoubleDomain::Element_ptr S, const size_t lds, typename Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)`
- `template<class Field >`  
`void MatF2MatFI_Triangular (const Field &F, Givaro::FloatDomain::Element_ptr S, const size_t lds, typename Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)`
- `template<class Field >`  
`void ftrsm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  
*ftrsm: TRIangular System solve with Matrix.*
- `template<class Field >`  
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)`  
*ftrmm: TRIangular Matrix Multiply.*
- `template<class Field >`  
`void ftrmm (const Field &F, const FFLAS_SIDE Side, const FFLAS_UPLO Uplo, const FFLAS_TRANSPOSE TransA, const FFLAS_DIAG Diag, const size_t M, const size_t N, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`

*ftmrm: TRIangular Matrix Multiply with 3 operands Computes  $C \leftarrow \alpha \text{op}(A)B + \text{beta}C$  or  $C \leftarrow \alpha B \text{op}(A) + \text{beta}C$ .*

- template<class Field >  
 Field::Element\_ptr fsyrk (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc)  
*fsyrk: Symmetric Rank K update*
- template<class Field >  
 Field::Element\_ptr fsyr2k (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc)  
*fsyr2k: Symmetric Rank 2K update*
- template<class Field >  
 Field::Element\_ptr fsyrk (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr D, const size\_t incD, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, const size\_t threshold=\_\_FFLASFFPACK\_FSYRK\_THRESHOLD)  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- template<class Field >  
 Field::Element\_ptr fsyrk (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr D, const size\_t incD, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, const ParSeqHelper::Sequential seq, const size\_t threshold)
- template<class Field , class Cut , class Param >  
 Field::Element\_ptr fsyrk (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t N, const size\_t K, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr D, const size\_t incD, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, const ParSeqHelper::Parallel< Cut, Param > par, const size\_t threshold)
- template<class Field >  
 Field::Element\_ptr fsyrk (const Field &F, const FFLAS\_UPLO UpLo, const FFLAS\_TRANSPOSE trans, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::Element\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr D, const size\_t incD, const std::vector< bool > &two← Block, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, const size\_t threshold=\_\_FFLASFFPACK\_FSYRK\_THRESHOLD)  
*fsyrk: Symmetric Rank K update with diagonal scaling*
- template<typename Field >  
 Field::Element\_ptr fgemm (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc)  
*fgemm: Field GENeral Matrix Multiply.*
- template<typename Field >  
 Field::Element\_ptr fgemm (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, const ParSeqHelper::← Sequential seq)
- template<typename Field , class Cut , class Param >  
 Field::Element\_ptr fgemm (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::ConstElement\_ptr B, const size\_t ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size\_t ldc, const ParSeqHelper::← Parallel< Cut, Param > par)
- template<typename Field >  
 Field::Element\_ptr pfgemm (const Field &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const typename Field::Element alpha, typename

`Field::ConstElement_ptr A, const size_t lda, typename Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, size_t numthreads=0)`

- `template<class Field >`  
`Field::Element * pfgemm_1D_rec (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, size_t seuil)`
- `template<class Field >`  
`Field::Element * pfgemm_2D_rec (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, size_t seuil)`
- `template<class Field >`  
`Field::Element * pfgemm_3D_rec (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, size_t seuil, size_t *x)`
- `template<class Field >`  
`Field::Element_ptr pfgemm_3D_rec2 (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element_ptr A, const size_t lda, const typename Field::Element_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, size_t seuil, size_t *x)`
- `template<class Field >`  
`Field::Element_ptr fsquare (const Field &F, const FFLAS_TRANSPOSE ta, const size_t n, const typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc)`

*fsquare: Squares a matrix.*

## 17.110.1 Macro Definition Documentation

### 17.110.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL

```
#define __FFLASFFPACK_fflas_fflas_level3_INL
```

### 17.110.1.2 \_\_FFLAS\_\_TRSM\_READONLY

```
#define __FFLAS__TRSM_READONLY
```

## 17.111 fflas\_pfgemm.inl File Reference

```
#include "fflas-ffpack/paladin/blockcuts.inl"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/paladin/pfgemm_variants.inl"
```

## Namespaces

- namespace `FFLAS`

## Macros

- `#define __FFLASFFPACK_fflas_pfgemm_INL`
- `#define __FFLASFFPACK_SEQPARTHRESHOLD 220`
- `#define __FFLASFFPACK_DIMKPENALTY 1`

## Functions

- `template<class Field, class ModeTrait, class Strat, class Param>  
std::enable_if<!std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElement↵  
Tag > >::value, typename Field::Element_ptr >::type fgemm (const Field &F, const FFLAS::FFLAS_TRANSPOSE  
ta, const FFLAS::FFLAS_TRANSPOSE tb, const size_t m, const size_t n, const size_t k, const  
typename Field::Element alpha, typename Field::ConstElement_ptr A, const size_t lda, typename  
Field::ConstElement_ptr B, const size_t ldb, const typename Field::Element beta, typename Field::Element_ptr  
C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, ModeTrait, ParSeqHelper::Parallel< Strat,  
Param > > &H)`

### 17.111.1 Macro Definition Documentation

#### 17.111.1.1 \_\_FFLASFFPACK\_fflas\_pfgemm\_INL

```
#define __FFLASFFPACK_fflas_pfgemm_INL
```

#### 17.111.1.2 \_\_FFLASFFPACK\_SEQPARTHRESHOLD

```
#define __FFLASFFPACK_SEQPARTHRESHOLD 220
```

#### 17.111.1.3 \_\_FFLASFFPACK\_DIMKPENALTY

```
#define __FFLASFFPACK_DIMKPENALTY 1
```

### 17.112 fflas\_pftsm.inl File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

## Namespaces

- namespace [FFLAS](#)

## Macros

- `#define __FFLASFFPACK_fflas_pftsm_INL`
- `#define PTRSM_HYBRID_THRESHOLD 256`

## Functions

- `template<class Field, class Cut, class Param>  
Field::Element_ptr ftrsm (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_UPLO  
UpLo, const FFLAS::FFLAS_TRANSPOSE TA, const FFLAS::FFLAS_DIAG Diag, const size_t m, const  
size_t n, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, typename  
Field::Element_ptr B, const size_t ldb, TRSMHelper< StructureHelper::Iterative, ParSeqHelper::Parallel<  
Cut, Param > > &H)`



- template<class [Field](#) , class Cut , class Param >  
[Field::Element\\_ptr](#) ftrsm (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_TRANSPOSE](#) TA, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, TRSMHelper< StructureHelper::Hybrid, ParSeqHelper::Parallel< Cut, Param > > &H)

## 17.112.1 Macro Definition Documentation

### 17.112.1.1 \_\_FFLASFFPACK\_fflas\_pftrsm\_INL

```
#define __FFLASFFPACK_fflas_pftrsm_INL
```

### 17.112.1.2 PTRSM\_HYBRID\_THRESHOLD

```
#define PTRSM_HYBRID_THRESHOLD 256
```

## 17.113 fflas\_simd.h File Reference

```
#include "fflas-ffpack/utils/fflas_intrinsic.h"
#include <iostream>
#include <type_traits>
#include <limits>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include <fflas-ffpack/fflas/fflas_simd/simd_modular.inl>
```

## Data Structures

- struct [support\\_simd](#)< T >
- struct [is\\_simd](#)< T >
- struct [NoSimd](#)< T >
- struct [SimdChooser](#)< T, bool, bool >
- struct [SimdChooser](#)< T, false, b >
- struct [SimdChooser](#)< T, true, false >
- struct [SimdChooser](#)< T, true, true >

## Namespaces

- namespace [FFLAS](#)

## Macros

- #define [SIMD\\_INT](#) 1
- #define [INLINE](#) inline
- #define [CONST](#)
- #define [PURE](#)
- #define [NORML\\_MOD](#)(C, P, NEGP, MIN, MAX, Q, T)
- #define [FLOAT\\_MOD](#)(C, P, INVP, Q)

## Typedefs

- template<class T >  
using [Simd](#) = typename [SimdChooser](#)< T >::value

## 17.113.1 Macro Definition Documentation

### 17.113.1.1 SIMD\_INT

```
#define SIMD_INT 1
```

### 17.113.1.2 INLINE

```
#define INLINE inline
```

### 17.113.1.3 CONST

```
#define CONST
```

### 17.113.1.4 PURE

```
#define PURE
```

### 17.113.1.5 NORML\_MOD

```
#define NORML_MOD(  
    C,  
    P,  
    NEGP,  
    MIN,  
    MAX,  
    Q,  
    T )
```

**Value:**

```
{  
    Q = greater(C, MAX);  
    T = lesser(C, MIN);  
    Q = vand(Q, NEGP);  
    T = vand(T, P);  
    Q = vor(Q, T);  
    C = add(C, Q);  
}
```

### 17.113.1.6 FLOAT\_MOD

```
#define FLOAT_MOD(  
    C,  
    P,  
    INV, P,  
    Q )
```

**Value:**

```
{  
    Q = mul(C, INV);  
    Q = floor(Q);  
}
```

```

    C = fnmadd(C, Q, P);
}

```

## 17.113.2 Typedef Documentation

### 17.113.2.1 Simd

```
using Simd = typename SimdChooser<T>::value
```

## 17.114 simd.dox File Reference

## 17.115 simd128.inl File Reference

```

#include "simd128_float.inl"
#include "simd128_double.inl"

```

## Data Structures

- struct [Simd128fp\\_base](#)
- struct [Simd128i\\_base](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_INL](#)

## Typedefs

- template<class T >  
using [Simd128](#) = [Simd128\\_impl](#)< std::is\_arithmetic< T >::value, std::is\_integral< T >::value, std::is\_↵  
signed< T >::value, sizeof(T)>

## 17.115.1 Macro Definition Documentation

### 17.115.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_INL
```

## 17.115.2 Typedef Documentation

### 17.115.2.1 Simd128

```
using Simd128 = Simd128_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std↵  
::is_signed<T>::value, sizeof(T)>
```

## 17.116 simd128\_double.inl File Reference

## Data Structures

- struct [Simd128\\_impl](#)< true, false, true, 8 >

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_double\\_INL](#)

### 17.116.1 Macro Definition Documentation

#### 17.116.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_double\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL
```

## 17.117 simd128\_float.inl File Reference

### Data Structures

- struct [Simd128\\_impl](#)< true, false, true, 4 >

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_float\\_INL](#)

### 17.117.1 Macro Definition Documentation

#### 17.117.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_float\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL
```

## 17.118 simd128\_int16.inl File Reference

### Data Structures

- struct [Simd128\\_impl](#)< true, true, true, 2 >
- union [Simd128\\_impl](#)< true, true, true, 2 >::Converter
- struct [Simd128\\_impl](#)< true, true, false, 2 >
- union [Simd128\\_impl](#)< true, true, false, 2 >::Converter

## Macros

- [#define \\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd128\\_int16\\_INL](#)

### 17.118.1 Macro Definition Documentation

#### 17.118.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL
```

## 17.119 simd128\_int32.inl File Reference

### Data Structures

- struct [Simd128\\_impl](#)< true, true, true, 4 >
- union [Simd128\\_impl](#)< true, true, true, 4 >::Converter

- struct [Simd128\\_impl< true, true, false, 4 >](#)
- union [Simd128\\_impl< true, true, false, 4 >::Converter](#)

## Macros

- `#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL`

### 17.119.1 Macro Definition Documentation

#### 17.119.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL
```

## 17.120 simd128\_int64.inl File Reference

### Data Structures

- struct [Simd128\\_impl< true, true, true, 8 >](#)
- union [Simd128\\_impl< true, true, true, 8 >::Converter](#)
- struct [Simd128\\_impl< true, true, false, 8 >](#)
- union [Simd128\\_impl< true, true, false, 8 >::Converter](#)

## Macros

- `#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL`
- `#define vect_t Simd128_impl<true,true,true,8>::vect_t`

### 17.120.1 Macro Definition Documentation

#### 17.120.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL
```

#### 17.120.1.2 vect\_t

```
#define vect_t Simd128_impl<true,true,true,8>::vect_t
```

## 17.121 simd256.inl File Reference

```
#include "simd256_float.inl"
#include "simd256_double.inl"
```

### Data Structures

- struct [Simd256fp\\_base](#)
- struct [Simd256i\\_base](#)

## Macros

- `#define __FFLASFFPACK_fflas_ffpack_utils_simd256_INL`

## Typedefs

- `template<class T>`  
`using Simd256 = Simd256_impl< std::is_arithmetic< T >::value, std::is_integral< T >::value, std::is_`  
`signed< T >::value, sizeof(T)>`

### 17.121.1 Macro Definition Documentation

#### 17.121.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_INL
```

### 17.121.2 Typedef Documentation

#### 17.121.2.1 Simd256

```
using Simd256 = Simd256_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std::is_
```

```
::is_signed<T>::value, sizeof(T)>
```

## 17.122 simd256\_double.inl File Reference

### Data Structures

- struct `Simd256_impl< true, false, true, 8 >`

### Macros

- `#define __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL`

### 17.122.1 Macro Definition Documentation

#### 17.122.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_double\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL
```

## 17.123 simd256\_float.inl File Reference

### Data Structures

- struct `Simd256_impl< true, false, true, 4 >`

### Macros

- `#define __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL`

### 17.123.1 Macro Definition Documentation

#### 17.123.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL
```

## 17.124 simd256\_int16.inl File Reference

### Data Structures

- struct [Simd256\\_impl](#)< true, true, true, 2 >
- union [Simd256\\_impl](#)< true, true, true, 2 >::Converter
- struct [Simd256\\_impl](#)< true, true, false, 2 >
- union [Simd256\\_impl](#)< true, true, false, 2 >::Converter

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_int16\\_INL](#)

### 17.124.1 Macro Definition Documentation

#### 17.124.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL
```

## 17.125 simd256\_int32.inl File Reference

### Data Structures

- struct [Simd256\\_impl](#)< true, true, true, 4 >
- union [Simd256\\_impl](#)< true, true, true, 4 >::Converter
- struct [Simd256\\_impl](#)< true, true, false, 4 >
- union [Simd256\\_impl](#)< true, true, false, 4 >::Converter

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_int32\\_INL](#)

### 17.125.1 Macro Definition Documentation

#### 17.125.1.1 \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL
```

## 17.126 simd256\_int64.inl File Reference

### Data Structures

- struct [Simd256\\_impl](#)< true, true, true, 8 >
- union [Simd256\\_impl](#)< true, true, true, 8 >::Converter
- struct [Simd256\\_impl](#)< true, true, false, 8 >
- union [Simd256\\_impl](#)< true, true, false, 8 >::Converter

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_ffpack\\_utils\\_simd256\\_int64\\_INL](#)
- #define [vect\\_t](#) [Simd256\\_impl](#)<true, true, true, 8>::vect\_t

## 17.126.1 Macro Definition Documentation

### 17.126.1.1 `__FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL`

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL
```

### 17.126.1.2 `vect_t`

```
#define vect_t Simd256_impl<true, true, true, 8>::vect_t
```

## 17.127 `simd512.inl` File Reference

```
#include "simd512_float.inl"
#include "simd512_double.inl"
#include "simd512_int64.inl"
```

### Data Structures

- struct [Simd512fp\\_base](#)
- struct [Simd512i\\_base](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_simd512\\_INL](#)

### Typedefs

- template<class T >  
using [Simd512](#) = [Simd512\\_impl](#)< std::is\_arithmetic< T >::value, std::is\_integral< T >::value, std::is\_↵  
signed< T >::value, sizeof(T)>

## 17.127.1 Macro Definition Documentation

### 17.127.1.1 `__FFLASFFPACK_simd512_INL`

```
#define __FFLASFFPACK_simd512_INL
```

## 17.127.2 Typedef Documentation

### 17.127.2.1 `Simd512`

```
using Simd512 = Simd512_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std↵  
::is_signed<T>::value, sizeof(T)>
```

## 17.128 `simd512_double.inl` File Reference

### Data Structures

- struct [Simd512\\_impl](#)< true, false, true, 8 >



## Macros

- `#define __FFLASFFPACK_simd512_double_INL`

### 17.128.1 Macro Definition Documentation

#### 17.128.1.1 \_\_FFLASFFPACK\_simd512\_double\_INL

```
#define __FFLASFFPACK_simd512_double_INL
```

## 17.129 simd512\_float.inl File Reference

### Data Structures

- struct [Simd512\\_impl](#)< true, false, true, 4 >

## Macros

- `#define __FFLASFFPACK_simd512_float_INL`

### 17.129.1 Macro Definition Documentation

#### 17.129.1.1 \_\_FFLASFFPACK\_simd512\_float\_INL

```
#define __FFLASFFPACK_simd512_float_INL
```

## 17.130 simd512\_int32.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd/simd512_int64.inl"
```

### Data Structures

- struct [Simd256\\_impl](#)< true, true, true, 4 >
- union [Simd256\\_impl](#)< true, true, true, 4 >::Converter
- struct [Simd256\\_impl](#)< true, true, false, 4 >
- union [Simd256\\_impl](#)< true, true, false, 4 >::Converter

## Macros

- `#define __FFLASFFPACK_simd512_int32_INL`

### 17.130.1 Macro Definition Documentation

#### 17.130.1.1 \_\_FFLASFFPACK\_simd512\_int32\_INL

```
#define __FFLASFFPACK_simd512_int32_INL
```

## 17.131 simd512\_int64.inl File Reference

### Data Structures

- struct [Simd512\\_impl](#)< true, true, true, 8 >
- union [Simd512\\_impl](#)< true, true, true, 8 >::Converter
- struct [Simd512\\_impl](#)< true, true, false, 8 >
- union [Simd512\\_impl](#)< true, true, false, 8 >::Converter

### Macros

- #define [\\_simd512\\_int64\\_INL](#)
- #define [vect\\_t Simd512\\_impl](#)<true, true, true, 8>::vect\_t

### 17.131.1 Macro Definition Documentation

#### 17.131.1.1 \_simd512\_int64\_INL

```
#define _simd512_int64_INL
```

#### 17.131.1.2 vect\_t

```
#define vect_t Simd512_impl<true, true, true, 8>::vect_t
```

## 17.132 simd\_modular.inl File Reference

### Data Structures

- class [FieldSimd](#)< \_Field >

## 17.133 fflas\_sparse.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/paladin/parallel.h"
#include <recint/recint.h>
#include <givaro/udl.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/field/field-traits.h"
#include "fflas-ffpack/fflas/fflas_bounds.inl"
#include "fflas-ffpack/utils/fflas_memory.h"
#include <type_traits>
#include <vector>
#include <iostream>
#include "fflas-ffpack/fflas/fflas_sparse/sparse_matrix_traits.h"
#include "fflas-ffpack/fflas/fflas_sparse/utils.h"
#include "fflas-ffpack/fflas/fflas_sparse/csr.h"
#include "fflas-ffpack/fflas/fflas_sparse/coo.h"
#include "fflas-ffpack/fflas/fflas_sparse/ell.h"
#include "fflas-ffpack/fflas/fflas_sparse/sell.h"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb.h"
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd.h"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo.h"
```

```
#include "fflas-ffpack/fflas/fflas_sparse.inl"
#include "fflas-ffpack/fflas/fflas_sparse/read_sparse.h"
```

## Data Structures

- struct [HelperFlag](#)
- struct [CsrMat< Field >](#)
- struct [CooMat< Field >](#)
- struct [EllMat< Field >](#)
- struct [SpMat< Field, flag >](#)

## Namespaces

- namespace [MKL\\_CONFIG](#)
- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details](#)

## Macros

- [#define index\\_t uint32\\_t](#)
- [#define ROUND\\_DOWN\(x, s\) \(\(x\) & ~\(\(s\)-1\)\)](#)
- [#define \\_\\_FFLASFFPACK\\_CACHE\\_LINE\\_SIZE 64](#)
- [#define assume\\_aligned\(pout, pin, v\) decltype\(pin\) pout = pin;](#)
- [#define DENSE\\_THRESHOLD 0.5](#)

## Enumerations

- enum class [SparseMatrix\\_t](#) {  
[CSR](#) , [CSR\\_ZO](#) , [CSC](#) , [CSC\\_ZO](#) ,  
[COO](#) , [COO\\_ZO](#) , [ELL](#) , [ELL\\_ZO](#) ,  
[SELL](#) , [SELL\\_ZO](#) , [ELL\\_simd](#) , [ELL\\_simd\\_ZO](#) ,  
[CSR\\_HYB](#) , [HYB\\_ZO](#) }

## Functions

- [template<class Field >](#)  
[void init\\_y](#) (const [Field](#) &F, const size\_t m, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y)
- [template<class Field >](#)  
[void init\\_y](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y, const int ldy)
- [template<class Field , class SM , class FC , class MZO >](#)  
[std::enable\\_if<!\(std::is\\_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloatTag >::value||std::is\\_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value\)>::type fspmv\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- [template<class Field , class SM , class FC , class MZO >](#)  
[std::enable\\_if< std::is\\_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloatTag >::value||std::is\\_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmv\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- [template<class Field , class SM >](#)  
[void fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::GenericTag, NotZOSparseMatrix)
- [template<class Field , class SM >](#)  
[std::enable\\_if<lisSparseMatrixSimdFormat< Field, SM >::value >::type fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::UnparametricTag, NotZOSparseMatrix)

- `template<class Field, class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM`  
`&A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::Unparametric←`  
`Tag, NotZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM`  
`&A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag,`  
`NotZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM &A,`  
`typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, Not←`  
`ZOSparseMatrix)`
- `template<class Field, class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr`  
`y, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if< !isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM`  
`&A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::Unparametric←`  
`Tag, ZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value >::type fspmv (const Field &F, const SM`  
`&A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::Unparametric←`  
`Tag, ZOSparseMatrix)`
- `template<class Field, class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr`  
`y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field, class SM, class FCat, class MZO >`  
`std::enable_if< !(std::is_same< typenameElementTraits< typenameField::Element >::value, Element←`  
`Categories::MachineFloatTag >::value || std::is_same< typenameElementTraits< typenameField::Element`  
`>::value, ElementCategories::MachineIntTag >::value) >::type fspmm_dispatch (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`  
`FCat, MZO)`
- `template<class Field, class SM, class FCat, class MZO >`  
`std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, Element←`  
`Categories::MachineFloatTag >::value || std::is_same< typenameElementTraits< typenameField::Element`  
`>::value, ElementCategories::MachineIntTag >::value >::type fspmm_dispatch (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`  
`FCat, MZO)`
- `template<class Field, class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx,`  
`typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`  
`FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if< !support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`  
`FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`  
`FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if< !support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM`  
`&A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy,`  
`FieldCategories::ModularTag, NotZOSparseMatrix)`

- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloatTag >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FCat, MZO)`
- `template<class Field , class SM , class FCat , class MZO >`  
`std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloatTag >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type pfspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FCat, MZO)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field , class SM >`  
`void pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field , class SM >`  
`std::enable_if<!support_simd< typenameField::Element >::value >::type pfspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int Idx, typename Field::Element_ptr y, int Idy, FieldCategories::UnparametricTag, ZOSparseMatrix)`

- `template<class Field, class SM >`  
`void pfsppmv (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field, class SM >`  
`void pfsppmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::false_type)`
- `template<class Field, class SM >`  
`void pfsppmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::false_type)`
- `template<class Field, class SM >`  
`void pfsppmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::false_type)`
- `template<class Field, class SM >`  
`void pfsppmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag, std::true_type)`
- `template<class Field, class SM >`  
`void pfsppmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, std::true_type)`
- `template<class Field, class SM >`  
`void pfsppmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field, class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)`
- `template<class Field, class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)`

## 17.133.1 Macro Definition Documentation

### 17.133.1.1 index\_t

```
#define index_t uint32_t
```

### 17.133.1.2 ROUND\_DOWN

```
#define ROUND_DOWN(  
    x,  
    s ) ((x) & ~((s)-1))
```

### 17.133.1.3 \_\_FFLASFFPACK\_CACHE\_LINE\_SIZE

```
#define __FFLASFFPACK_CACHE_LINE_SIZE 64
```

### 17.133.1.4 assume\_aligned

```
#define assume_aligned(  
    pout,  
    pin,  
    v ) decltype(pin) pout = pin;
```

## 17.133.1.5 DENSE\_THRESHOLD

```
#define DENSE_THRESHOLD 0.5
```

## 17.134 fflas\_sparse.inl File Reference

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_fflas\\_sparse\\_INL](#)

## Functions

- template<class [Field](#) >  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y)
- template<class [Field](#) >  
void [init\\_y](#) (const [Field](#) &F, const size\_t m, const size\_t n, const typename [Field::Element](#) b, typename [Field::Element\\_ptr](#) y, const int ldy)
- template<class [Field](#) , class SM , class FC , class MZO >  
std::enable\_if<!(std::is\_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloatTag >::value||std::is\_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type [fspmv\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- template<class [Field](#) , class SM , class FC , class MZO >  
std::enable\_if< std::is\_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloatTag >::value||std::is\_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type [fspmv\\_dispatch](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FC fc, MZO mzo)
- template<class [Field](#) , class SM >  
void [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::GenericTag, NotZOSparseMatrix)
- template<class [Field](#) , class SM >  
std::enable\_if<lisSparseMatrixSimdFormat< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class [Field](#) , class SM >  
std::enable\_if< isSparseMatrixSimdFormat< [Field](#), SM >::value &&support\_simd< typenameField::Element >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::UnparametricTag, NotZOSparseMatrix)
- template<class [Field](#) , class SM >  
std::enable\_if<lisSparseMatrixSimdFormat< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class [Field](#) , class SM >  
std::enable\_if< isSparseMatrixSimdFormat< [Field](#), SM >::value &&support\_simd< typenameField::Element >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::ModularTag, NotZOSparseMatrix)
- template<class [Field](#) , class SM >  
void [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::GenericTag, ZOSparseMatrix)
- template<class [Field](#) , class SM >  
std::enable\_if<lisSparseMatrixSimdFormat< [Field](#), SM >::value >::type [fspmv](#) (const [Field](#) &F, const SM &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::UnparametricTag, ZOSparseMatrix)



- `template<class Field, class SM >`  
`std::enable_if< isSparseMatrixSimdFormat< Field, SM >::value &&support_simd< typenameField::Element >::value >::type fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field, class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag, std::true_type)`
- `template<class Field, class SM, class FCat, class MZO >`  
`std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloatTag >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value)>::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field, class SM, class FCat, class MZO >`  
`std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineFloatTag >::value||std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::MachineIntTag >::value >::type fspmm_dispatch (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FCat, MZO)`
- `template<class Field, class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, NotZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, NotZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, NotZOSparseMatrix)`
- `template<class Field, class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag, ZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if< support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field, class SM >`  
`std::enable_if<!support_simd< typenameField::Element >::value >::type fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag, ZOSparseMatrix)`
- `template<class Field, class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::ModularTag, ZOSparseMatrix)`
- `template<class Field, class SM >`  
`void fspmv (const Field &F, const SM &A, typename Field::ConstElement_ptr x, const typename Field::Element &beta, typename Field::Element_ptr y)`
- `template<class Field, class SM >`  
`void fspmm (const Field &F, const SM &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, const typename Field::Element &beta, typename Field::Element_ptr y, int ldy)`



## 17.134.1 Macro Definition Documentation

### 17.134.1.1 \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL

```
#define __FFLASFFPACK_fflas_fflas_sparse_INL
```

## 17.135 coo.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/coo/coo_spm্ম.inl"
```

### Data Structures

- struct [Sparse<\\_Field, SparseMatrix\\_t::COO>](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::COO\\_ZO>](#)

### Namespaces

- namespace [FFLAS](#)

### Functions

- template<class [Field](#), class IndexT>  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::COO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#), class IndexT>  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::COO\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#)>  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::COO > &A)
- template<class [Field](#)>  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::COO\_ZO > &A)

## 17.136 coo\_spm্ম.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_coo\\_spm্ম\\_INL](#)

### Functions

- template<class [Field](#)>  
void [fspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::COO > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, FieldCategories::GenericTag)
- template<class [Field](#)>  
void [fspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::COO > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, FieldCategories::UnparametricTag)

- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::COO_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

## 17.136.1 Macro Definition Documentation

### 17.136.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL
```

## 17.137 coo\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_coo\\_spmv\\_INL](#)

## Functions

- template<class [Field](#) >  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::COO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class [Field](#) >  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::COO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class [Field](#) >  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::COO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const [uint64\\_t](#) kmax)
- template<class [Field](#) >  
void [fspmv\\_one](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::COO\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class [Field](#) >  
void [fspmv\\_mone](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::COO\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class [Field](#) >  
void [fspmv\\_one](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::COO\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class [Field](#) >  
void [fspmv\\_mone](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::COO\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)

### 17.137.1 Macro Definition Documentation

#### 17.137.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL
```

## 17.138 coo\_utils.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_coo\\_utils\\_INL](#)

## Functions

- template<class [Field](#) >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::COO > &A)
- template<class [Field](#) >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::COO\_ZO > &A)
- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::COO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::COO\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)

### 17.138.1 Macro Definition Documentation

### 17.138.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_utils_INL
```

## 17.139 csr.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr/csr_spmv.inl"
```

### Data Structures

- struct [Sparse<\\_Field, SparseMatrix\\_t::CSR>](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::CSR\\_ZO>](#)

### Namespaces

- namespace [FFLAS](#)

### Functions

- template<class [Field](#), class IndexT>  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::CSR > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#), class IndexT>  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::CSR\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#)>  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::CSR > &A)
- template<class [Field](#)>  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::CSR\_ZO > &A)

## 17.140 csr\_pspmm.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_pspmm\\_INL](#)

### Functions

- template<class [Field](#)>  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, FieldCategories::GenericTag)
- template<class [Field](#)>  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, FieldCategories::UnparametricTag)
- template<class [Field](#)>  
void [pfspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x\_, int ldx, typename [Field::Element\\_ptr](#) y\_, int ldy, const [int64\\_t](#) kmax)

- `template<class Field >`  
`void pfpsmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::↵`  
`GenericTag)`
- `template<class Field >`  
`void pfpsmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::↵`  
`GenericTag)`
- `template<class Field >`  
`void pfpsmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::↵`  
`UnparametricTag)`
- `template<class Field >`  
`void pfpsmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::↵`  
`UnparametricTag)`

## 17.140.1 Macro Definition Documentation

### 17.140.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL
```

## 17.141 csr\_pspmv.inl File Reference

```
#include <thread>
```

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL`

### Functions

- `template<class Field >`  
`void pfpsmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfpsmv_task (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const index_t iStart, const index_t iStop,`  
`FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfpsmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfpsmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr`  
`x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfpsmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename`  
`Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`

- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.141.1 Macro Definition Documentation

### 17.141.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL
```

## 17.142 csr\_spmv.inl File Reference

### Namespaces

- namespace `FFLAS`
- namespace `FFLAS::sparse_details_impl`

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, index_t blockSize, typename Field::ConstElement_ptr x_, index_t ldx, typename Field::Element_ptr y_, index_t ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`

- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

## 17.142.1 Macro Definition Documentation

### 17.142.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL
```

## 17.143 csr\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.143.1 Macro Definition Documentation

### 17.143.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL
```

## 17.144 csr\_utils.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::CSR_ZO > &A)`
- `template<class Field >`  
`std::ostream & sparse_print (std::ostream &os, const Sparse< Field, SparseMatrix_t::CSR > &A)`
- `template<class IndexT >`  
`void sparse_init (const Givaro::Modular< Givaro::Integer > &F, Sparse< Givaro::Modular< Givaro::Integer >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT >`  
`void sparse_init (const Givaro::ZRing< Givaro::Integer > &F, Sparse< Givaro::ZRing< Givaro::Integer >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, Givaro::Integer *dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT, size_t RECINT_SIZE>`  
`void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class IndexT, size_t RECINT_SIZE>`  
`void sparse_init (const Givaro::ZRing< RecInt::rmint< RECINT_SIZE > > &F, Sparse< Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Givaro::ZRing< RecInt::rmint< RECINT_SIZE > >::Element_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::CSR_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.145 csr\_hyb.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/csr_hyb/csr_hyb_spmmm.inl"
```



## Data Structures

- struct [Sparse< \\_Field, SparseMatrix\\_t::CSR\\_HYB >](#)

## Namespaces

- namespace [FFLAS](#)

## Functions

- template<class [Field](#) >  
void [sparse\\_delete](#) (const [Sparse< Field, SparseMatrix\\_t::CSR\\_HYB >](#) &A)
- template<class [Field](#) , class [IndexT](#) >  
void [sparse\\_init](#) (const [Field](#) &F, [Sparse< Field, SparseMatrix\\_t::CSR\\_HYB >](#) &A, const [IndexT](#) \*row, const [IndexT](#) \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)

## 17.146 csr\_hyb\_pspmm.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_HYB\\_pspmm\\_INL](#)

### Functions

- template<class [Field](#) >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::CSR\\_HYB >](#) &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::GenericTag](#))
- template<class [Field](#) >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::CSR\\_HYB >](#) &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x, [int](#) ldx, typename [Field::Element\\_ptr](#) y, [int](#) ldy, [FieldCategories::GenericTag](#))
- template<class [Field](#) >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::CSR\\_HYB >](#) &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [FieldCategories::UnparametricTag](#))
- template<class [Field](#) >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::CSR\\_HYB >](#) &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x, [int](#) ldx, typename [Field::Element\\_ptr](#) y, [int](#) ldy, [FieldCategories::UnparametricTag](#))
- template<class [Field](#) >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::CSR\\_HYB >](#) &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, const [int64\\_t](#) kmax)
- template<class [Field](#) >  
void [pfspmm](#) (const [Field](#) &F, const [Sparse< Field, SparseMatrix\\_t::CSR\\_HYB >](#) &A, [size\\_t](#) blockSize, typename [Field::ConstElement\\_ptr](#) x, [int](#) ldx, typename [Field::Element\\_ptr](#) y, [int](#) ldy, const [int64\\_t](#) kmax)

### 17.146.1 Macro Definition Documentation

#### 17.146.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL
```

## 17.147 csr\_hyb\_pspmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL`

### Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`

## 17.147.1 Macro Definition Documentation

### 17.147.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL
```

## 17.148 csr\_hyb\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::CSR_HYB > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`

## 17.148.1 Macro Definition Documentation

### 17.148.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
```

## 17.149 csr\_hyb\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_HYB\\_spmv\\_INL](#)

### Functions

- template<class [Field](#) >  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR\_HYB > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class [Field](#) >  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR\_HYB > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class [Field](#) >  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::CSR\_HYB > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const [uint64\\_t](#) kmax)

## 17.149.1 Macro Definition Documentation

### 17.149.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
```

## 17.150 csr\_hyb\_utils.inl File Reference

### Data Structures

- struct [Info](#)
- struct [Coo](#)< [ValT](#), [IdxT](#) >

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::csr\\_hyb\\_details](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_CSR\\_HYB\\_utils\\_INL](#)

## Functions

- template<class [Field](#) >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::CSR\_HYB > &A)
- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::CSR\_HYB > &A, const IndexT \*row, const IndexT \*col, typename [Field](#)::ConstElement\_ptr dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)

### 17.150.1 Macro Definition Documentation

#### 17.150.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL
```

## 17.151 ell.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell/ell_spmv.inl"
```

## Data Structures

- struct [Sparse](#)< [\\_Field](#), SparseMatrix\_t::ELL >
- struct [Sparse](#)< [\\_Field](#), SparseMatrix\_t::ELL\_ZO >

## Namespaces

- namespace [FFLAS](#)

## Functions

- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL > &A, const IndexT \*row, const IndexT \*col, typename [Field](#)::ConstElement\_ptr dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field](#)::ConstElement\_ptr dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#) >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL > &A)
- template<class [Field](#) >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL\_ZO > &A)

## 17.152 ell\_pspmm.inl File Reference

## Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_pspmm\\_INL](#)

## Functions

- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, const int64_t kmax)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, const int64_t kmax)`
- `template<class Field, class Func >`  
`void pfspmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, Func &&func)`
- `template<class Field, class Func >`  
`void pfspmm_zo (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, Func &&func)`

### 17.152.1 Macro Definition Documentation

#### 17.152.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL
```

## 17.153 ell\_pspmv.inl File Reference

### Namespaces

- namespace `FFLAS`
- namespace `FFLAS::sparse_details_impl`

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL`

### Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`

- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.153.1 Macro Definition Documentation

### 17.153.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL
```

## 17.154 ell\_spm.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_spm_INL`

### Functions

- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, const int64_t kmax)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmm_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

- `template<class Field >`  
`void fspmm_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_one_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_aligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmm_mone_simd_unaligned (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, size_t blockSize, typename Field::ConstElement_ptr x_, int ldx, typename Field::Element_ptr y_, int ldy, FieldCategories::UnparametricTag)`

## 17.154.1 Macro Definition Documentation

### 17.154.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL
```

## 17.155 ell\_spmv.inl File Reference

### Namespaces

- namespace `FFLAS`
- namespace `FFLAS::sparse_details_impl`

### Macros

- `#define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL`

### Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::ELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`

- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::ELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

## 17.155.1 Macro Definition Documentation

### 17.155.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL
```

## 17.156 ell\_utils.inl File Reference

```
#include <vector>
```

### Namespaces

- namespace [FFLAS](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_utils\\_INL](#)

### Functions

- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::ELL_ZO > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::ELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.156.1 Macro Definition Documentation

### 17.156.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_utils_INL
```

## 17.157 ell\_simd.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd/ell_simd_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/ell_simd/ell_simd_spmv.inl"
```



## Data Structures

- struct [Sparse<\\_Field, SparseMatrix\\_t::ELL\\_simd>](#)
- struct [Sparse<\\_Field, SparseMatrix\\_t::ELL\\_simd\\_ZO>](#)

## Namespaces

- namespace [FFLAS](#)

## Functions

- template<class [Field](#), class IndexT>  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#), class IndexT>  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#)>  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A)
- template<class [Field](#)>  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A)

## 17.158 ell\_simd\_pspmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_simd\\_pspmv\\_INL](#)

### Functions

- template<class [Field](#)>  
void [pfspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class [Field](#)>  
void [pfspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class [Field](#)>  
void [pfspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const [uint64\\_t](#) kmax)
- template<class [Field](#)>  
void [pfspmv\\_one](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class [Field](#)>  
void [pfspmv\\_mone](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class [Field](#)>  
void [pfspmv\\_one](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class [Field](#)>  
void [pfspmv\\_mone](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)

## 17.158.1 Macro Definition Documentation

### 17.158.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL
```

## 17.159 ell\_simd\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_simd\\_spmv\\_INL](#)

### Functions

- template<class [Field](#) >  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class [Field](#) >  
void [fspmv\\_simd](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class [Field](#) >  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class [Field](#) >  
void [fspmv\\_simd](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const [uint64\\_t](#) kmax)
- template<class [Field](#) >  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, const [uint64\\_t](#) kmax)
- template<class [Field](#) >  
void [fspmv\\_one](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class [Field](#) >  
void [fspmv\\_mone](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::GenericTag)
- template<class [Field](#) >  
void [fspmv\\_one](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class [Field](#) >  
void [fspmv\\_mone](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class [Field](#) >  
void [fspmv\\_one\\_simd](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)
- template<class [Field](#) >  
void [fspmv\\_mone\\_simd](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x\_, typename [Field::Element\\_ptr](#) y\_, FieldCategories::UnparametricTag)

## 17.159.1 Macro Definition Documentation

**17.159.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL**

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL
```

**17.160 ell\_simd\_utils.inl File Reference****Namespaces**

- namespace [FFLAS](#)

**Macros**

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_ELL\\_simd\\_utils\\_INL](#)

**Functions**

- template<class [Field](#) >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A)
- template<class [Field](#) >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A)
- template<class [Field](#) >  
void [sparse\\_print](#) (const Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A)
- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL\_simd > &A, const IndexT \*row, const IndexT \*col, typename [Field](#)::ConstElement\_ptr dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::ELL\_simd\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field](#)::ConstElement\_ptr dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)

**17.160.1 Macro Definition Documentation****17.160.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL**

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL
```

**17.161 hyb\_zo.h File Reference**

```
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_spmv.inl"
#include "fflas-ffpack/fflas/fflas_sparse/hyb_zo/hyb_zo_spmmm.inl"
```

**Data Structures**

- struct [Sparse< \\_Field, SparseMatrix\\_t::HYB\\_ZO >](#)

**Namespaces**

- namespace [FFLAS](#)

**17.162 hyb\_zo\_pspmm.inl File Reference****Namespaces**

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

## Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL`

## Functions

- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize,`  
`typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, FieldCategories::↵`  
`UnparametricTag)`
- `template<class Field >`  
`void pfspmm (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, size_t blockSize, type-`  
`name Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)`

### 17.162.1 Macro Definition Documentation

#### 17.162.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL
```

## 17.163 hyb\_zo\_pspmv.inl File Reference

### Namespaces

- namespace `FFLAS`
- namespace `FFLAS::sparse_details_impl`

## Macros

- `#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL`

## Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::HYB_ZO > &A, typename`  
`Field::ConstElement_ptr x, typename Field::Element_ptr y, uint64_t kmax)`

### 17.163.1 Macro Definition Documentation

#### 17.163.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL
```

## 17.164 hyb\_zo\_spm্ম.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_HYB\\_ZO\\_spm্ম\\_INL](#)

### Functions

- `template<class Field >`  
void [fspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, FieldCategories::GenericTag)
- `template<class Field >`  
void [fspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, FieldCategories::UnparametricTag)
- `template<class Field >`  
void [fspmm](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, size\_t blockSize, typename [Field::ConstElement\\_ptr](#) x, int ldx, typename [Field::Element\\_ptr](#) y, int ldy, [uint64\\_t](#) kmax)

### 17.164.1 Macro Definition Documentation

#### 17.164.1.1 [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_HYB\\_ZO\\_spm্ম\\_INL](#)

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spm্ম_INL
```

## 17.165 hyb\_zo\_spmmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_HYB\\_ZO\\_spmmv\\_INL](#)

### Functions

- `template<class Field >`  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::GenericTag)
- `template<class Field >`  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, FieldCategories::UnparametricTag)
- `template<class Field >`  
void [fspmv](#) (const [Field](#) &F, const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, typename [Field::ConstElement\\_ptr](#) x, typename [Field::Element\\_ptr](#) y, [uint64\\_t](#) kmax)

### 17.165.1 Macro Definition Documentation

### 17.165.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL
```

## 17.166 hyb\_zo\_utils.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_HYB\\_ZO\\_utils\\_INL](#)

### Functions

- template<class [Field](#) >  
void [sparse\\_delete](#) (const Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A)
- template<class [Field](#) , class IndexT >  
void [sparse\\_init](#) (const [Field](#) &F, Sparse< [Field](#), SparseMatrix\_t::HYB\_ZO > &A, const IndexT \*row, const IndexT \*col, typename [Field::ConstElement\\_ptr](#) dat, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)
- template<typename [\\_Field](#) >  
std::ostream & [operator<<](#) (std::ostream &os, const Sparse< [\\_Field](#), SparseMatrix\_t::HYB\_ZO > &A)

## 17.166.1 Macro Definition Documentation

### 17.166.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL
```

## 17.167 read\_sparse.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <fstream>
#include <string>
#include <cstdlib>
#include <iterator>
```

### Data Structures

- struct [Coo< Field >](#)
- struct [readMyMachineType< Field, T >](#)
- struct [readMyMachineType< Field, mpz\\_t >](#)

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::details\\_spmv](#)

### Macros

- #define [DNS\\_BIN\\_VER](#) 0
- #define [mask\\_t](#) [uint64\\_t](#)

## Functions

- `template<class Field, bool sorted = true, bool read_integer = false>`  
`void readSmsFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnc)`
- `template<class Field >`  
`void readSprFormat (const std::string &path, const Field &f, index_t *&row, index_t *&col, typename Field::Element_ptr &val, index_t &rowdim, index_t &coldim, uint64_t &nnc)`
- `template<class T >`  
`std::enable_if< std::is_integral< T >::value, int > getDataType ()`
- `template<class T >`  
`std::enable_if< std::is_floating_point< T >::value, int > getDataType ()`
- `template<class T >`  
`std::enable_if< std::is_same< T, mpz_t >::value, int > getDataType ()`
- `template<class T >`  
`int getDataType ()`
- `template<class Field >`  
`void readMachineType (const Field &F, typename Field::Element &modulo, typename Field::Element_ptr val, std::ifstream &file, const uint64_t dims, const mask_t data_type, const mask_t field_desc)`
- `template<class Field >`  
`void readDnsFormat (const std::string &path, const Field &F, index_t &rowdim, index_t &coldim, typename Field::Element_ptr &val)`
- `template<class Field >`  
`void writeDnsFormat (const std::string &path, const Field &F, const index_t &rowdim, const index_t &coldim, typename Field::Element_ptr A, index_t ldA)`

### 17.167.1 Macro Definition Documentation

#### 17.167.1.1 DNS\_BIN\_VER

```
#define DNS_BIN_VER 0
```

#### 17.167.1.2 mask\_t

```
#define mask_t uint64_t
```

## 17.168 sell.h File Reference

```
#include "fflas-ffpack/fflas/fflas_sparse/sell/sell_utils.inl"
#include "fflas-ffpack/fflas/fflas_sparse/sell/sell_spmv.inl"
```

## Data Structures

- struct `Sparse<_Field, SparseMatrix_t::SELL >`
- struct `Sparse<_Field, SparseMatrix_t::SELL_ZO >`

## Namespaces

- namespace `FFLAS`

## 17.169 sell\_pspmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_sell\\_pspmv\\_INL](#)

### Functions

- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const int64_t kmax)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void pfspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void pfspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 17.169.1 Macro Definition Documentation

#### 17.169.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_pspmv_INL
```

## 17.170 sell\_spmv.inl File Reference

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sparse\\_details\\_impl](#)

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_fflas\\_sparse\\_sell\\_spmv\\_INL](#)



## Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, const uint64_t kmax)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::GenericTag)`
- `template<class Field >`  
`void fspmv_one_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone_simd (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_one (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`
- `template<class Field >`  
`void fspmv_mone (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x_, typename Field::Element_ptr y_, FieldCategories::UnparametricTag)`

### 17.170.1 Macro Definition Documentation

#### 17.170.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_spmv_INL
```

## 17.171 sell\_utils.inl File Reference

### Data Structures

- struct [Info](#)
- struct [Coo< ValT, IdxT >](#)

### Namespaces

- namespace [FFLAS](#)
- namespace [FFLAS::sell\\_details](#)

## Macros

- `#define __FFLASFFPACK_fflas_sparse_sell_utils_INL`

## Functions

- `template<class Field >`  
`void fspmv (const Field &F, const Sparse< Field, SparseMatrix_t::SELL_ZO > &A, typename Field::ConstElement_ptr x, typename Field::Element_ptr y, FieldCategories::ModularTag)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field >`  
`void sparse_delete (const Sparse< Field, SparseMatrix_t::SELL_ZO > &A)`
- `template<class Field >`  
`void sparse_print (const Sparse< Field, SparseMatrix_t::SELL > &A)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz, uint64_t sigma=0)`
- `template<class Field, class IndexT >`  
`void sparse_init (const Field &F, Sparse< Field, SparseMatrix_t::SELL_ZO > &A, const IndexT *row, const IndexT *col, typename Field::ConstElement_ptr dat, uint64_t rowdim, uint64_t coldim, uint64_t nnz)`

## 17.171.1 Macro Definition Documentation

### 17.171.1.1 \_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_utils_INL
```

## 17.172 sparse\_matrix\_traits.h File Reference

```
#include <type_traits>
```

## Data Structures

- `struct isSparseMatrix< Field, M >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR > >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO > >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL > >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL > >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd > >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_simd_ZO > >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_HYB > >`
- `struct isSparseMatrix< Field, Sparse< Field, SparseMatrix_t::HYB_ZO > >`
- `struct isZOSparseMatrix< F, M >`
- `struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::CSR_ZO > >`
- `struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::COO_ZO > >`
- `struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::ELL_ZO > >`
- `struct isZOSparseMatrix< Field, Sparse< Field, SparseMatrix_t::SELL_ZO > >`

- struct [isZOSparseMatrix](#)< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >
- struct [isSparseMatrixSimdFormat](#)< F, M >
- struct [isSparseMatrixMKLFormat](#)< F, M >
- struct [tfn\\_plus](#)
- struct [tfn\\_mul](#)
- struct [tfn\\_mul\\_eq](#)
- struct [tfn\\_minus](#)
- struct [tfn\\_plus\\_eq](#)
- struct [tfn\\_minus\\_eq](#)
- struct [has\\_plus\\_impl](#)< C >
- struct [has\\_mul\\_impl](#)< C >
- struct [has\\_mul\\_eq\\_impl](#)< C >
- struct [has\\_plus\\_eq\\_impl](#)< C >
- struct [has\\_minus\\_eq\\_impl](#)< C >
- struct [has\\_minus\\_impl](#)< C >
- struct [has\\_operation](#)< T >

## Namespaces

- namespace [FFLAS](#)

## Typedefs

- using [ZOSparseMatrix](#) = std::true\_type
- using [NotZOSparseMatrix](#) = std::false\_type
- using [SimdSparseMatrix](#) = std::true\_type
- using [NoSimdSparseMatrix](#) = std::false\_type
- using [MKLSparseMatrixFormat](#) = std::true\_type
- using [NotMKLSparseMatrixFormat](#) = std::false\_type
- template<class T >  
using [has\\_plus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_plus\_impl< T > >::type
- template<class T >  
using [has\\_minus](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_minus\_↵\_impl< T > >::type
- template<class T >  
using [has\\_equal](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, std::is\_copy\_↵\_assignable< T > >::type
- template<class T >  
using [has\\_plus\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_plus\_↵\_eq\_impl< T > >::type
- template<class T >  
using [has\\_minus\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_↵\_minus\_eq\_impl< T > >::type
- template<class T >  
using [has\\_mul](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_mul\_impl< T > >::type
- template<class T >  
using [has\\_mul\\_eq](#) = typename std::conditional< std::is\_arithmetic< T >::value, std::true\_type, has\_mul\_↵\_eq\_impl< T > >::type

## 17.173 utils.h File Reference

```
#include <algorithm>
#include <numeric>
#include <vector>
```

## Data Structures

- struct [StatsMatrix](#)

## Namespaces

- namespace [FFLAS](#)

## Functions

- template<class It >  
double [computeDeviation](#) (It begin, It end)
- template<class Field >  
StatsMatrix [getStat](#) (const Field &F, const [index\\_t](#) \*row, const [index\\_t](#) \*col, typename [Field::ConstElement\\_ptr](#) val, [uint64\\_t](#) rowdim, [uint64\\_t](#) coldim, [uint64\\_t](#) nnz)

## 17.174 ffpack.doxy File Reference

## 17.175 ffpack.h File Reference

Set of elimination based routines for dense linear algebra.

```
#include "givaro/givpoly1.h"
#include <fflas-ffpack/fflas-ffpack-config.h>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include <list>
#include <vector>
#include <iostream>
#include <algorithm>
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include "ffpack_fgesv.inl"
#include "ffpack_fgetrs.inl"
#include "fflas-ffpack/checkers/checkers_ffpack.inl"
#include "ffpack_pluq.inl"
#include "ffpack_pluq_mp.inl"
#include "ffpack_ppluq.inl"
#include "ffpack_ludivine.inl"
#include "ffpack_ludivine_mp.inl"
#include "ffpack_echelonforms.inl"
#include "ffpack_fsytrf.inl"
#include "ffpack_invert.inl"
#include "ffpack_ftrtr.inl"
#include "ffpack_ftrstr.inl"
#include "ffpack_ftrssyr2k.inl"
#include "ffpack_charpoly_kglu.inl"
#include "ffpack_charpoly_kgfast.inl"
#include "ffpack_charpoly_kgfastgeneralized.inl"
#include "ffpack_charpoly_danilevski.inl"
#include "ffpack_charpoly.inl"
#include "ffpack_frobenius.inl"
#include "ffpack_minpoly.inl"
#include "ffpack_krylovelim.inl"
#include "ffpack_permutation.inl"
#include "ffpack_rankprofiles.inl"
#include "ffpack_det_mp.inl"
#include "ffpack.inl"
```

## Data Structures

- class [CharpolyFailed](#)

## Namespaces

- namespace [FFPACK](#)  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

## Macros

- `#define __FFLASFFPACK_FTRSTR_THRESHOLD 64`
- `#define __FFLASFFPACK_FTRSSYR2K_THRESHOLD 64`

## Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)  
*Conversion of a permutation from LAPACK format to Math format.*
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)  
*Conversion of a permutation from Maths format to LAPACK format.*
- template<class [Field](#) >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P)  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- template<class [Field](#) >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const [FFLAS::ParSeqHelper::Sequential](#) seq)
- template<class [Field](#), class [Cut](#), class [Param](#) >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const [FFLAS::ParSeqHelper::Parallel](#)< [Cut](#), [Param](#) > par)
- template<class [Field](#) >  
void [MonotonicApplyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t R)  
*Apply a R-monotonically increasing permutation P, to the matrix A.*
- template<class [Field](#) >  
void [fgetrs](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename [Field::Element\\_ptr](#) B, const size\_t ldb, int \*info)  
*Solve the system  $AX = B$  or  $XA = B$ .*
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [fgetrs](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename [Field::Element\\_ptr](#) X, const size\_t ldx, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, int \*info)  
*Solve the system  $AX = B$  or  $XA = B$ .*
- template<class [Field](#) >  
size\_t [fgesv](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, int \*info)  
*Square system solver.*

- `template<class Field >`  
`size_t fgesv (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, const size_t NRHS, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)`  
*Rectangular system solver.*
- `template<class Field >`  
`void ftrtri (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG Diag, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FTRTRI_THRESHOLD)`  
*Compute the inverse of a triangular matrix.*
- `template<class Field >`  
`void trinv_left (const Field &F, const size_t N, typename Field::ConstElement_ptr L, const size_t ldl, typename Field::Element_ptr X, const size_t ldx)`
- `template<class Field >`  
`void ftrrm (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda)`  
*Compute the product of two triangular matrices of opposite shape.*
- `template<class Field >`  
`void ftrstr (const Field &F, const FFLAS::FFLAS_SIDE side, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const FFLAS::FFLAS_DIAG diagB, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSTR_THRESHOLD)`  
*Solve a triangular system with a triangular right hand side of the same shape.*
- `template<class Field >`  
`void ftrssyr2k (const Field &F, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diagA, const size_t N, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldx, const size_t ldb, const size_t threshold=__FFLASFFPACK_FTRSSYR2K_THRESHOLD)`  
*Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.*
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field, class Cut, class Param >`  
`bool fsytrf (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`
- `template<class Field >`  
`bool fsytrf_nonunit (const Field &F, const FFLAS::FFLAS_UPLO UpLo, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr D, const size_t incD, const size_t threshold=__FFLASFFPACK_FSYTRF_THRESHOLD)`  
*Triangular factorization of symmetric matrices.*
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`  
*Compute a PLUQ factorization of the given matrix.*
- `template<class Field >`  
`size_t pPLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)`
- `template<class Field >`  
`size_t PLUQ (const Field &F, const FFLAS::FFLAS_DIAG Diag, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q, const FFLAS::ParSeqHelper::Sequential &PShelper, size_t BCThreshold=__FFLASFFPACK_PLUQ_THRESHOLD)`

- template<class [Field](#) , class [Cut](#) , class [Param](#) >  
size\_t [PLUQ](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFLAS::ParSeqHelper::Parallel](#)< [Cut](#), [Param](#) > &PSHelper)
- template<class [Field](#) >  
size\_t [LUdivine](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#), const size\_t cutoff=\_\_FFLASFFPACK\_LUDIVINE↵\_THRESHOLD)  
*Compute the CUP or PLE factorization of the given matrix.*
- template<class [Field](#) >  
size\_t [LUdivine\\_construct](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, typename [Field::Element\\_ptr](#) u, const size\_t incu, size\_t \*P, bool computeX, const FFPACK\_MINPOLY\_TAG MinTag=[FfpackDense](#), const size\_t kg\_mc=0, const size\_t kg\_mb=0, const size\_t kg\_j=0)
- template<class [Field](#) >  
size\_t [ColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform=false, const FFPACK\_LU\_TAG Lu↵Tag=[FfpackSlabRecursive](#))  
*Compute the Column Echelon form of the input matrix in-place.*
- template<class [Field](#) >  
size\_t [pColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class [PSHelper](#) >  
size\_t [ColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const [PSHelper](#) &psH)
- template<class [Field](#) >  
size\_t [RowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, const FFPACK\_LU\_TAG Lu↵Tag=[FfpackSlabRecursive](#))  
*Compute the Row Echelon form of the input matrix in-place.*
- template<class [Field](#) >  
size\_t [pRowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_↵LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class [PSHelper](#) >  
size\_t [RowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const [PSHelper](#) &psH)
- template<class [Field](#) >  
size\_t [ReducedColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, const FFPACK\_↵LU\_TAG LuTag=[FfpackSlabRecursive](#))  
*Compute the Reduced Column Echelon form of the input matrix in-place.*
- template<class [Field](#) >  
size\_t [pReducedColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#) , class [PSHelper](#) >  
size\_t [ReducedColumnEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_↵TAG LuTag, const [PSHelper](#) &psH)
- template<class [Field](#) >  
size\_t [ReducedRowEchelonForm](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#)



A, const size\_t Ida, size\_t \*P, size\_t \*Qt, const bool transform=false, const FFPACK\_LU\_TAG LuTag=FfpackSlabRecursive)

*Compute the Reduced Row Echelon form of the input matrix in-place.*

- template<class Field >  
size\_t pReducedRowEchelonForm (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t Ida, size\_t \*P, size\_t \*Qt, const bool transform=false, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=FfpackTileRecursive)
  - template<class Field , class PSHelper >  
size\_t ReducedRowEchelonForm (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t Ida, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag, const PSHelper &psH)
  - template<class Field >  
size\_t GaussJordan (const Field &F, const size\_t M, const size\_t N, typename Field::Element\_ptr A, const size\_t Ida, const size\_t colbeg, const size\_t rowbeg, const size\_t colsize, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag)
- Gauss-Jordan algorithm computing the Reduced Row echelon form and its transform matrix.*
- template<class Field >  
Field::Element\_ptr Invert (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t Ida, int &>nullity)
- Invert the given matrix in place or computes its nullity if it is singular.*
- template<class Field >  
Field::Element\_ptr Invert (const Field &F, const size\_t M, typename Field::ConstElement\_ptr A, const size\_t Ida, typename Field::Element\_ptr X, const size\_t Idx, int &>nullity)
- Invert the given matrix or computes its nullity if it is singular.*
- template<class Field >  
Field::Element\_ptr Invert2 (const Field &F, const size\_t M, typename Field::Element\_ptr A, const size\_t Ida, typename Field::Element\_ptr X, const size\_t Idx, int &>nullity)
- Invert the given matrix or computes its nullity if it is singular.*
- template<class PolRing >  
std::list< typename PolRing::Element > & CharPoly (const PolRing &R, std::list< typename PolRing::Element > &charp, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t Ida, typename PolRing::Domain\_t::Randlter &G, const FFPACK\_CHARPOLY\_TAG CharpTag=FfpackAuto, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)
- Compute the characteristic polynomial of the matrix A.*
- template<class PolRing >  
PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t Ida, typename PolRing::Domain\_t::Randlter &G, const FFPACK\_CHARPOLY\_TAG CharpTag=FfpackAuto, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)
- Compute the characteristic polynomial of the matrix A.*
- template<class PolRing >  
PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size\_t N, typename PolRing::Domain\_t::Element\_ptr A, const size\_t Ida, const FFPACK\_CHARPOLY\_TAG CharpTag=FfpackAuto, const size\_t degree=\_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD)
- Compute the characteristic polynomial of the matrix A.*
- template<class Field , class Polynomial >  
std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t Ida)
  - template<class Field , class Polynomial >  
int KGFast (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::Element\_ptr A, const size\_t Ida, size\_t \*kg\_mc, size\_t \*kg\_mb, size\_t \*kg\_j)
  - template<class Field , class Polynomial >  
std::list< Polynomial > & KGFast\_generalized (const Field &F, std::list< Polynomial > &charp, const size\_t N, typename Field::Element\_ptr A, const size\_t Ida)
  - template<class Field >  
void fgemv\_kgf (const Field &F, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t Ida, typename Field::ConstElement\_ptr X, const size\_t incX, typename Field::Element\_ptr Y, const size\_t incY, const size\_t kg\_mc, const size\_t kg\_mb, const size\_t kg\_j)



- `template<class Field , class Polynomial , class Randlter >`  
`std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr U, const size_t ldu, Randlter &G)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t Ida)`
- `template<class PolRing >`  
`void RandomKrylovPrecond (const PolRing &PR, std::list< typename PolRing::Element > &completed, Factors, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t Ida, size_t &Nb, typename PolRing::Domain_t::Element_ptr &B, size_t &ldb, typename PolRing::Domain_t::Randlter &g, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`
- `template<class PolRing >`  
`std::list< typename PolRing::Element > & ArithProg (const PolRing &PR, std::list< typename PolRing::Element > &frobeniusForm, const size_t N, typename PolRing::Domain_t::Element_ptr A, const size_t Ida, const size_t degree)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & LUKrylov_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element_ptr A, const size_t Ida, typename Field::Element_ptr X, const size_t ldx)`
- `template<class Field , class Polynomial >`  
`Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t Ida)`  
*Compute the minimal polynomial of the matrix A.*
- `template<class Field , class Polynomial , class Randlter >`  
`Polynomial & MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t Ida, Randlter &G)`  
*Compute the minimal polynomial of the matrix A.*
- `template<class Field , class Polynomial >`  
`Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::ConstElement_ptr v, const size_t incv)`  
*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*
- `template<class Field , class Polynomial >`  
`Polynomial & MatVecMinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::Element_ptr v, const size_t incv, typename Field::Element_ptr K, const size_t ldk, size_t *P)`
- `template<class Field , class Polynomial >`  
`Polynomial & Hybrid_KGF_LUK_MinPoly (const Field &F, Polynomial &minP, const size_t N, typename Field::ConstElement_ptr A, const size_t Ida, typename Field::Element_ptr X, const size_t ldx, size_t *P, const FFPACK_MINPOLY_TAG MinTag=FFPACK::FfpackDense, const size_t kg_mc=0, const size_t kg_mb=0, const size_t kg_j=0)`
- `template<class Field >`  
`size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida)`  
*Computes the rank of the given matrix using a PLUQ factorization.*
- `template<class Field >`  
`size_t pRank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, size_t numthreads=0)`
- `template<class Field , class PSHelper >`  
`size_t Rank (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida, const PSHelper &psH)`
- `template<class Field >`  
`bool IsSingular (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t Ida)`  
*Returns true if the given matrix is singular.*
- `template<class Field >`  
`Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t Ida, size_t *P=NULL, size_t *Q=NULL)`

*Returns the determinant of the given square matrix.*

- `template<class Field >`  
`Field::Element & pDet (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t numthreads=0, size_t *P=NULL, size_t *Q=NULL)`
- `template<class Field , class PSHelper >`  
`Field::Element & Det (const Field &F, typename Field::Element &det, const size_t N, typename Field::Element_ptr A, const size_t lda, const PSHelper &psH, size_t *P=NULL, size_t *Q=NULL)`
- `template<class Field >`  
`Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb)`

*Solves a linear system  $AX = b$  using PLUQ factorization.*

- `template<class Field , class PSHelper >`  
`Field::Element_ptr Solve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, PSHelper &psH)`
- `template<class Field >`  
`Field::Element_ptr pSolve (const Field &F, const size_t M, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr x, const int incx, typename Field::ConstElement_ptr b, const int incb, size_t numthreads=0)`
- `template<class Field >`  
`*void RandomNullSpaceVector (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr X, const size_t incX)`

*Solve  $LX = B$  or  $XL = B$  in place.*

- `template<class Field >`  
`size_t NullSpaceBasis (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr &NS, size_t &ldn, size_t &NSdim)`

*Computes a basis of the Left/Right nullspace of the matrix A.*

- `template<class Field >`  
`size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Computes the row rank profile of A.*

- `template<class Field >`  
`size_t pRowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FpackTileRecursive)`
- `template<class Field , class PSHelper >`  
`size_t RowRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)`
- `template<class Field >`  
`size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag=FpackSlabRecursive)`

*Computes the column rank profile of A.*

- `template<class Field >`  
`size_t pColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, size_t numthreads=0, const FFPACK_LU_TAG LuTag=FpackTileRecursive)`
- `template<class Field , class PSHelper >`  
`size_t ColumnRankProfile (const Field &F, const size_t M, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *&rkprofile, const FFPACK_LU_TAG LuTag, PSHelper &psH)`
- `void RankProfileFromLU (const size_t *P, const size_t N, const size_t R, size_t *&rkprofile, const FFPACK_LU_TAG LuTag)`

*Recovers the column/row rank profile from the permutation of an LU decomposition.*

- `size_t LeadingSubmatrixRankProfiles (const size_t M, const size_t N, const size_t R, const size_t LSm, const size_t LSn, const size_t *P, const size_t *Q, size_t *RRP, size_t *CRP)`

*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*

- template<class [Field](#) >  
size\_t [RowRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)  
*RowRankProfileSubmatrixIndices.*
- template<class [Field](#) >  
size\_t [ColRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)  
*Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.*
- template<class [Field](#) >  
size\_t [RowRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)  
*Computes the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*
- template<class [Field](#) >  
size\_t [ColRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)  
*Compute the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*
- template<class [Field](#) >  
void [getTriangular](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false)  
*Extracts a triangular matrix from a compact storage  $A=L\backslash U$  of rank R.*
- template<class [Field](#) >  
void [getTriangular](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Cleans up a compact storage  $A=L\backslash U$  to reveal a triangular matrix of rank R.*
- template<class [Field](#) >  
void [getEchelonForm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false, const [FFPACK\\_LU\\_TAG](#) LuTag=[FfpackSlabRecursive](#))  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank R obtained by RowEchelonForm or ColumnEchelonForm.*
- template<class [Field](#) >  
void [getEchelonForm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename [Field::Element\\_ptr](#) A, const size\_t lda, const [FFPACK\\_LU\\_TAG](#) LuTag=[FfpackSlabRecursive](#))  
*Cleans up a compact storage  $A=L\backslash U$  obtained by RowEchelonForm or ColumnEchelonForm to reveal an echelon form of rank R.*
- template<class [Field](#) >  
void [getEchelonTransform](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const [FFPACK\\_LU\\_TAG](#) LuTag=[FfpackSlabRecursive](#))  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank R obtained by RowEchelonForm or ColumnEchelonForm.*
- template<class [Field](#) >  
void [getReducedEchelonForm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false, const [FFPACK\\_LU\\_TAG](#) LuTag=[FfpackSlabRecursive](#))  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelonForm with transform = true.*
- template<class [Field](#) >  
void [getReducedEchelonForm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename [Field::Element\\_ptr](#) A, const size\_t lda, const [FFPACK\\_LU\\_TAG](#) LuTag=[FfpackSlabRecursive](#))

*Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `ReducedRowEchelonForm` or `ReducedColumnEchelonForm` with `transform = true`.*

- template<class [Field](#) >  
void [getReducedEchelonTransform](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const [FFPACK\\_LU\\_TAG](#) LuTag=[FpackSlabRecursive](#))  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `RowEchelonForm` or `ColumnEchelonForm`.*
- void [PLUQtoEchelonPermutation](#) (const size\_t N, const size\_t R, const size\_t \*P, size\_t \*outPerm)  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [LQUPtoInverseOfFullRankMinor](#) (const [Field](#) &F, const size\_t rank, typename [Field::Element\\_ptr](#) A\_factors, const size\_t lda, const size\_t \*QtPointer, typename [Field::Element\\_ptr](#) X, const size\_t ldx)  
*LQUPtoInverseOfFullRankMinor.*

## 17.175.1 Detailed Description

Set of elimination based routines for dense linear algebra.

Matrices are supposed over finite prime field of characteristic less than  $2^{26}$ .

## 17.175.2 Macro Definition Documentation

### 17.175.2.1 \_\_FFLASFFPACK\_FTRSTR\_THRESHOLD

```
#define __FFLASFFPACK_FTRSTR_THRESHOLD 64
```

### 17.175.2.2 \_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD

```
#define __FFLASFFPACK_FTRSSYR2K_THRESHOLD 64
```

## 17.176 fpack.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_fpack\\_INL](#)

### Functions

- template<class [Field](#) >  
size\_t [Rank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Computes the rank of the given matrix using a PLUQ factorization.*
- template<class [Field](#) >  
size\_t [pRank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t numthreads=0)
- template<class [Field](#) , class PSHelper >  
size\_t [Rank](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const PSHelper &psH)

- template<class [Field](#) >  
 bool [IsSingular](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Returns true if the given matrix is singular.*
- template<class [Field](#) >  
[Field::Element](#) & [Det](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P=NULL, size\_t \*Q=NULL)  
*Returns the determinant of the given square matrix.*
- template<class [Field](#) >  
[Field::Element](#) & [pDet](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t numthreads=0, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class [Field](#) , class PSHelper >  
[Field::Element](#) & [Det](#) (const [Field](#) &F, typename [Field::Element](#) &det, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const PSHelper &psH, size\_t \*P=NULL, size\_t \*Q=NULL)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [Solve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb)  
*Solves a linear system  $AX = b$  using PLUQ factorization.*
- template<class [Field](#) , class PSHelper >  
[Field::Element\\_ptr](#) [Solve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb, PSHelper &psH)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [pSolve](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) x, const int incx, typename [Field::ConstElement\\_ptr](#) b, const int incb, size\_t numthreads=0)
- template<class [Field](#) >  
 void [RandomNullSpaceVector](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t incX)  
*Solve  $LX = B$  or  $XL = B$  in place.*
- template<class [Field](#) >  
 size\_t [NullSpaceBasis](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &NS, size\_t &ldn, size\_t &NSdim)  
*Computes a basis of the Left/Right nullspace of the matrix A.*
- template<class [Field](#) >  
 void [solveLB](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) L, const size\_t ldl, const size\_t \*Q, typename [Field::Element\\_ptr](#) B, const size\_t ldb)
- template<class [Field](#) >  
 void [solveLB2](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) L, const size\_t ldl, const size\_t \*Q, typename [Field::Element\\_ptr](#) B, const size\_t ldb)

## 17.176.1 Macro Definition Documentation

### 17.176.1.1 \_\_FFLASFFPACK\_ffpack\_INL

```
#define __FFLASFFPACK_ffpack_INL
```

## 17.177 ffpack\_charpoly.inl File Reference

```
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "ffpack_charpoly_mp.inl"
```

## Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

## Macros

- `#define` [\\_\\_FFLASFFPACK\\_charpoly\\_INL](#)

## Functions

- `template<class PolRing >`  
`std::list< typename PolRing::Element > & CharPoly (const PolRing &R, std::list< typename PolRing::Element > &charp, const size_t N, typename PolRing::Domain\_t::Element\_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`  
*Compute the characteristic polynomial of the matrix A.*
- `template<class PolRing >`  
`PolRing::Element & CharPoly (const PolRing &R, typename PolRing::Element &charp, const size_t N, typename PolRing::Domain\_t::Element\_ptr A, const size_t lda, typename PolRing::Domain_t::RandIter &G, const FFPACK_CHARPOLY_TAG CharpTag=FfpackAuto, const size_t degree=__FFLASFFPACK_ARITHPROG_THRESHOLD)`  
*Compute the characteristic polynomial of the matrix A.*
- `template<class Field , class Polynomial , class RandIter >`  
`std::list< Polynomial > & LUKrylov (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element\_ptr A, const size_t lda, typename Field::Element\_ptr U, const size_t ldu, RandIter &G)`
- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & LUKrylov\_KGFast (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element\_ptr A, const size_t lda, typename Field::Element\_ptr X, const size_t ldx)`

## 17.177.1 Macro Definition Documentation

### 17.177.1.1 \_\_FFLASFFPACK\_charpoly\_INL

```
#define __FFLASFFPACK_charpoly_INL
```

## 17.178 ffpack\_charpoly\_danilevski.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- `#define` [\\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_danilveski\\_INL](#)

### Functions

- `template<class Field , class Polynomial >`  
`std::list< Polynomial > & Danilevski (const Field &F, std::list< Polynomial > &charp, const size_t N, typename Field::Element\_ptr A, const size_t lda)`

## 17.178.1 Macro Definition Documentation

## 17.178.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_danilveski_INL
```

## 17.179 ffpack\_charpoly\_kgfast.inl File Reference

## Namespaces

- namespace [FFPACK](#)  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_kgfast\\_INL](#)

## Functions

- template<class [Field](#) , class Polynomial >  
int [KGFast](#) (const [Field](#) &F, std::list< Polynomial > &charp, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*kg\_mc, size\_t \*kg\_mb, size\_t \*kg\_j)
- template<class [Field](#) >  
void [fgemv\\_kgf](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, typename [Field::Element\\_ptr](#) Y, const size\_t incY, const size\_t kg\_mc, const size\_t kg\_mb, const size\_t kg\_j)

## 17.179.1 Macro Definition Documentation

## 17.179.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfast_INL
```

## 17.180 ffpack\_charpoly\_kgfastgeneralized.inl File Reference

```
#include <iostream>
#include "fflas-ffpack/utils/fflas_io.h"
```

## Namespaces

- namespace [FFPACK](#)  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

## Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_kgfastgeneralized\\_INL](#)

## Functions

- template<class [Field](#) >  
[Field::Element\\_ptr buildMatrix](#) (const [Field](#) &F, typename [Field::ConstElement\\_ptr](#) E, typename [Field::ConstElement\\_ptr](#) C, const size\_t lda, const size\_t \*B, const size\_t \*T, const size\_t me, const size\_t mc, const size\_t lambda, const size\_t mu)

- template<class [Field](#) , class Polynomial >  
std::list< Polynomial > & [KGFast\\_generalized](#) (const [Field](#) &F, std::list< Polynomial > &charp, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)

## 17.180.1 Macro Definition Documentation

### 17.180.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL
```

## 17.181 ffpack\_charpoly\_kglu.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite **Field** **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_charpoly\\_kglu\\_INL](#)

### Functions

- template<class [Field](#) >  
size\_t [updateD](#) (const [Field](#) &F, size\_t \*d, size\_t k, std::vector< std::vector< typename [Field::Element](#) > > &minpt)
- template<class [Field](#) >  
size\_t [newD](#) (const [Field](#) &F, size\_t \*d, bool &KeepOn, const size\_t l, const size\_t N, typename [Field::Element\\_ptr](#) X, const size\_t \*Q, std::vector< std::vector< typename [Field::Element](#) > > &minpt)
- template<class [Field](#) , class Polynomial >  
std::list< Polynomial > & [KellerGehrig](#) (const [Field](#) &F, std::list< Polynomial > &charp, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)

## 17.181.1 Macro Definition Documentation

### 17.181.1.1 \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kglu_INL
```

## 17.182 ffpack\_charpoly\_mp.inl File Reference

```
#include <givaro/zring.h>
#include "givaro/givinteger.h"
#include "givaro/givpoly1.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite **Field** **PACK** Set of elimination based routines for dense linear algebra.*



## Macros

- `#define __FFPACK_charpoly_mp_INL`

## Functions

- `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr CharPoly` (const `FFPACK::RNSInteger< FFPACK::rns_double > &F`, typename `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr` charp, const `size_t N`, typename `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr` A, const `size_t Ida`, `Givaro::ZRing< Givaro::Integer >::RandIter &G`, const `FFPACK_CHARPOLY_TAG` CharpTag, `size_t degree`)
- `template<> Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element & CharPoly` (const `Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > > &R`, `Givaro::Poly1Dom< Givaro::ZRing< Givaro::Integer > >::Element &charp`, const `size_t N`, `Givaro::Integer *A`, const `size_t Ida`, `Givaro::ZRing< Givaro::Integer >::RandIter &G`, const `FFPACK_CHARPOLY_TAG` CharpTag, `size_t degree`)

## 17.182.1 Macro Definition Documentation

### 17.182.1.1 \_\_FFPACK\_charpoly\_mp\_INL

```
#define __FFPACK_charpoly_mp_INL
```

## 17.183 ffpack\_det\_mp.inl File Reference

```
#include <givaro/zring.h>
#include "givaro/givinteger.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
```

## Namespaces

- namespace `FFPACK`  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*

## Macros

- `#define __FFPACK_det_mp_INL`

## Functions

- `template<class PSHelper > FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr & Det` (const `FFPACK::RNSInteger< FFPACK::rns_double > &F`, typename `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr` &det, const `size_t N`, typename `FFPACK::RNSInteger< FFPACK::rns_double >::Element_ptr` A, const `size_t Ida`, const `PSHelper &psH`)
- `template<class PSHelper > Givaro::Integer & Det` (const `Givaro::ZRing< Givaro::Integer > &F`, `Givaro::Integer &det`, const `size_t N`, `Givaro::Integer *A`, const `size_t Ida`, const `PSHelper &psH`, `size_t *P`, `size_t *Q`)

## 17.183.1 Macro Definition Documentation

### 17.183.1.1 \_\_FFPACK\_det\_mp\_INL

```
#define __FFPACK_det_mp_INL
```

## 17.184 ffpack\_echelonforms.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_echelon\\_forms\\_INL](#)
- #define [\\_\\_FFLASFFPACK\\_GAUSSJORDAN\\_BASECASE](#) 256

### Functions

- template<class [Field](#) >  
void [getTriangular](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false)  
*Extracts a triangular matrix from a compact storage  $A=L\backslash U$  of rank  $R$ .*
- template<class [Field](#) >  
void [getTriangular](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda)  
*Cleans up a compact storage  $A=L\backslash U$  to reveal a triangular matrix of rank  $R$ .*
- void [PLUQtoEchelonPermutation](#) (const size\_t N, const size\_t R, const size\_t \*P, size\_t \*outPerm)  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- template<class [Field](#) >  
void [getEchelonForm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false, const [FFPACK\\_LU\\_TAG](#) LuTag=[FfpackSlabRecursive](#))  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by [RowEchelonForm](#) or [ColumnEchelonForm](#).*
- template<class [Field](#) >  
void [getEchelonForm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename [Field::Element\\_ptr](#) A, const size\_t lda, const [FFPACK\\_LU\\_TAG](#) LuTag=[FfpackSlabRecursive](#))  
*Cleans up a compact storage  $A=L\backslash U$  obtained by [RowEchelonForm](#) or [ColumnEchelonForm](#) to reveal an echelon form of rank  $R$ .*
- template<class [Field](#) >  
void [getEchelonTransform](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const [FFPACK\\_LU\\_TAG](#) LuTag=[FfpackSlabRecursive](#))  
*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by [RowEchelonForm](#) or [ColumnEchelonForm](#).*
- template<class [Field](#) >  
void [getReducedEchelonForm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const bool OnlyNonZeroVectors=false, const [FFPACK\\_LU\\_TAG](#) LuTag=[FfpackSlabRecursive](#))  
*Extracts a matrix in echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by [ReducedRowEchelonForm](#) or [ReducedColumnEchelonForm](#) with transform = true.*
- template<class [Field](#) >  
void [getReducedEchelonForm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, typename [Field::Element\\_ptr](#) A, const size\_t lda, const [FFPACK\\_LU\\_TAG](#) LuTag=[FfpackSlabRecursive](#))

*Cleans up a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `ReducedRowEchelonForm` or `ReducedColumnEchelonForm` with `transform = true`.*

- template<class [Field](#) >  
void [getReducedEchelonTransform](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) T, const size\_t ldt, const [FFPACK\\_LU\\_TAG](#) LuTag=[FfpackSlabRecursive](#))

*Extracts a transformation matrix to echelon form from a compact storage  $A=L\backslash U$  of rank  $R$  obtained by `RowEchelonForm` or `ColumnEchelonForm`.*

## 17.184.1 Macro Definition Documentation

### 17.184.1.1 \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL

```
#define __FFLASFFPACK_ffpack_echelon_forms_INL
```

### 17.184.1.2 \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 256
```

## 17.185 ffpack\_fgesv.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite **Field** **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_fgesv\\_INL](#)

### Functions

- template<class [Field](#) >  
size\_t [fgesv](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, int \*info)  
*Square system solver.*
- template<class [Field](#) >  
size\_t [fgesv](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, int \*info)  
*Rectangular system solver.*

## 17.185.1 Macro Definition Documentation

### 17.185.1.1 \_\_FFLASFFPACK\_ffpack\_fgesv\_INL

```
#define __FFLASFFPACK_ffpack_fgesv_INL
```

## 17.186 ffpack\_fgetrs.inl File Reference

### Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_fgetrs\\_INL](#)

### Functions

- template<class [Field](#) >  
void [fgetrs](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename [Field::Element\\_ptr](#) B, const size\_t ldb, int \*info)  
*Solve the system  $AX = B$  or  $XA = B$ .*
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [fgetrs](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t \*Q, typename [Field::Element\\_ptr](#) X, const size\_t idx, typename [Field::ConstElement\\_ptr](#) B, const size\_t ldb, int \*info)  
*Solve the system  $A X = B$  or  $X A = B$ .*

## 17.186.1 Macro Definition Documentation

### 17.186.1.1 \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL

```
#define __FFLASFFPACK_ffpack_fgetrs_INL
```

## 17.187 ffpack\_frobenius.inl File Reference

```
#include <givaro/givranditer.h>
```

### Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

- namespace [FFPACK::Protected](#)

### Functions

- template<class [Field](#) >  
void [CompressRows](#) ([Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t ldtmp, const size\_t \*d, const size\_t nb\_blocs)
- template<class [Field](#) >  
void [CompressRowsQK](#) ([Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t ldtmp, const size\_t \*d, const size\_t deg, const size\_t nb\_blocs)
- template<class [Field](#) >  
void [DeCompressRows](#) ([Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t ldtmp, const size\_t \*d, const size\_t nb\_blocs)

- template<class [Field](#) >  
void [DeCompressRowsQK](#) ([Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t ldtmp, const size\_t \*d, const size\_t deg, const size\_t nb\_blocs)
- template<class [Field](#) >  
void [CompressRowsQA](#) ([Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t ldtmp, const size\_t \*d, const size\_t nb\_blocs)
- template<class [Field](#) >  
void [DeCompressRowsQA](#) ([Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) tmp, const size\_t ldtmp, const size\_t \*d, const size\_t nb\_blocs)
- template<class [PolRing](#) >  
void [RandomKrylovPrecond](#) (const [PolRing](#) &PR, std::list< typename [PolRing::Element](#) > &completedFactors, const size\_t N, typename [PolRing::Domain\\_t::Element\\_ptr](#) A, const size\_t lda, size\_t &Nb, typename [PolRing::Domain\\_t::Element\\_ptr](#) &B, size\_t &ldb, typename [PolRing::Domain\\_t::RandIter](#) &g, const size\_t degree=[\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#))
- template<class [PolRing](#) >  
std::list< typename [PolRing::Element](#) > & [ArithProg](#) (const [PolRing](#) &PR, std::list< typename [PolRing::Element](#) > &frobeniusForm, const size\_t N, typename [PolRing::Domain\\_t::Element\\_ptr](#) A, const size\_t lda, const size\_t degree)

## 17.188 ffpack\_fsytrf.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_fsytrf\\_INL](#)

### Functions

- template<class [Field](#) >  
bool [fsytrf\\_BC\\_Crout](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv)
- template<class [Field](#) >  
size\_t [fsytrf\\_BC\\_RL](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv)
- template<class [Field](#) >  
size\_t [fsytrf\\_UP\\_RPM\\_BC\\_RL](#) (const [Field](#) &F, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv, size\_t \*P)
- template<class [Field](#) >  
size\_t [fsytrf\\_LOW\\_RPM\\_BC\\_Crout](#) (const [Field](#) &F, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv, size\_t \*P)
- template<class [Field](#) >  
size\_t [fsytrf\\_UP\\_RPM\\_BC\\_Crout](#) (const [Field](#) &F, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv, size\_t \*P)
- template<class [Field](#) >  
size\_t [fsytrf\\_UP\\_RPM](#) (const [Field](#) &Fi, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) Dinv, const size\_t incDinv, size\_t \*P, size\_t BCThreshold)
- template<class [Field](#) >  
bool [fsytrf\\_nonunit](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const size\_t N, typename

```
Field::Element_ptr A, const size_t lda, typename Field::Element_ptr Dinv, const size_t incDinv,
FFLAS::ParSeqHelper::Sequential seq, size_t threshold)
```

- template<class Field , class Cut , class Param >  
bool **fsytrf\_nonunit** (const Field &F, const FFLAS::FFLAS\_UPLO UpLo, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, typename Field::Element\_ptr Dinv, const size\_t incDinv, FFLAS::ParSeqHelper::Parallel< Cut, Param > par, size\_t threshold)
- template<class Field >  
bool **fsytrf** (const Field &F, const FFLAS::FFLAS\_UPLO UpLo, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const size\_t threshold=\_\_FFLASFFPACK\_FSYTRF\_THRESHOLD)  
*Triangular factorization of symmetric matrices.*
- template<class Field >  
bool **fsytrf** (const Field &F, const FFLAS::FFLAS\_UPLO UpLo, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const FFLAS::ParSeqHelper::Sequential seq, const size\_t threshold=\_\_FFLASFFPACK\_FSYTRF\_THRESHOLD)
- template<class Field , class Cut , class Param >  
bool **fsytrf** (const Field &F, const FFLAS::FFLAS\_UPLO UpLo, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par, const size\_t threshold=\_\_FFLASFFPACK\_FSYTRF\_THRESHOLD)
- template<class Field >  
size\_t **fsytrf\_RPM** (const Field &F, const FFLAS::FFLAS\_UPLO UpLo, const size\_t N, typename Field::Element\_ptr A, const size\_t lda, size\_t \*P, size\_t threshold)
- template<class Field >  
void **getTridiagonal** (const Field &F, const size\_t N, const size\_t R, typename Field::ConstElement\_ptr A, const size\_t lda, size\_t \*P, typename Field::Element\_ptr T, const size\_t ldt)

## 17.188.1 Macro Definition Documentation

### 17.188.1.1 \_\_FFLASFFPACK\_ffpack\_fsytrf\_INL

```
#define __FFLASFFPACK_ffpack_fsytrf_INL
```

## 17.189 ffpack\_ftrssyr2k.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

## Namespaces

- namespace **FFPACK**  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define **\_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL**

## Functions

- template<class Field >  
void **ftrssyr2k** (const Field &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG diagA, const size\_t N, typename Field::ConstElement\_ptr A, const size\_t lda, typename Field::Element\_ptr B, const size\_t ldb, const size\_t threshold=\_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD)  
*Solve a triangular system in a symmetric sum: find B upper/lower triangular such that  $A^T B + B^T A = C$  where C is symmetric.*

## 17.189.1 Macro Definition Documentation

## 17.189.1.1 \_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL

```
#define __FFLASFFPACK_ffpack_ftrssyr2k_INL
```

## 17.190 ffpack\_ftrstr.inl File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
```

## Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ftrstr\\_INL](#)

## Functions

- template<class [Field](#) >  
void [ftrstr](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) diagA, const [FFLAS::FFLAS\\_DIAG](#) diagB, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) B, const size\_t ldb, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FTRSTR\\_THRESHOLD](#))

*Solve a triangular system with a triangular right hand side of the same shape.*

## 17.190.1 Macro Definition Documentation

## 17.190.1.1 \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL

```
#define __FFLASFFPACK_ffpack_ftrstr_INL
```

## 17.191 ffpack\_ftrtr.inl File Reference

## Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1
- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ftrtr\\_INL](#)

## Functions

- template<class [Field](#) >  
void [ftrtri](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_UPLO](#) Uplo, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t threshold=[\\_\\_FFLASFFPACK\\_FTRTRI\\_THRESHOLD](#))  
*Compute the inverse of a triangular matrix.*
- template<class [Field](#) >  
void [ftrtrm](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) side, const [FFLAS::FFLAS\\_DIAG](#) diag, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda)

*Compute the product of two triangular matrices of opposite shape.*

- template<class [Field](#) >  
void [trinv\\_left](#) (const [Field](#) &F, const size\_t N, typename [Field::ConstElement\\_ptr](#) L, const size\_t ldl, typename [Field::Element\\_ptr](#) X, const size\_t ldx)

## 17.191.1 Macro Definition Documentation

### 17.191.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

### 17.191.1.2 \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL

```
#define __FFLASFFPACK_ffpack_ftrtr_INL
```

## 17.192 ffpack\_invert.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite **Field** **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_invert\\_INL](#)

### Functions

- template<class [Field](#) >  
[Field::Element\\_ptr Invert](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, int &nullity)  
*Invert the given matrix in place or computes its nullity if it is singular.*
- template<class [Field](#) >  
[Field::Element\\_ptr Invert](#) (const [Field](#) &F, const size\_t M, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*
- template<class [Field](#) >  
[Field::Element\\_ptr Invert2](#) (const [Field](#) &F, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, int &nullity)  
*Invert the given matrix or computes its nullity if it is singular.*

## 17.192.1 Macro Definition Documentation

### 17.192.1.1 \_\_FFLASFFPACK\_ffpack\_invert\_INL

```
#define __FFLASFFPACK_ffpack_invert_INL
```

## 17.193 ffpack\_krylovelim.inl File Reference

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_krylovelim\\_INL](#)



## 17.193.1 Macro Definition Documentation

### 17.193.1.1 \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL

```
#define __FFLASFFPACK_ffpack_krylovelim_INL
```

## 17.194 ffpack\_ludivine.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_bounds.inl"
```

### Data Structures

- class [callLUdivine\\_small< Element >](#)
- class [callLUdivine\\_small< double >](#)
- class [callLUdivine\\_small< float >](#)

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ludivine\\_INL](#)

### Functions

- template<class [Field](#) >  
size\_t [LUdivine\\_gauss](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)
- template<class [Field](#) >  
size\_t [LUdivine\\_small](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag)
- template<class [Field](#) >  
size\_t [LUdivine](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFPACK::FFPACK\\_LU\\_TAG](#) LuTag, const size\_t cutoff)
- template<class [Field](#) >  
size\_t [LUdivine\\_construct](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, typename [Field::Element\\_ptr](#) u, const size\_t incu, size\_t \*P, bool computeX, const [FFPACK::FFPACK\\_MINPOLY\\_TAG](#) MinTag, const size\_t kg\_mc, const size\_t kg\_mb, const size\_t kg\_j)

## 17.194.1 Macro Definition Documentation

### 17.194.1.1 \_\_FFLASFFPACK\_ffpack\_ludivine\_INL

```
#define __FFLASFFPACK_ffpack_ludivine_INL
```

## 17.195 ffpack\_ludivine\_mp.inl File Reference

```
#include <givaro/modular-integer.h>
#include <givaro/givinteger.h>
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack_ludivine.inl"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFPACK\\_ludivine\\_mp\\_INL](#)

### Functions

- template<> size\_t [LUdivine](#) (const Givaro::Modular< Givaro::Integer > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const [FFLAS::FFLAS\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, typename Givaro::Integer \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const FFPACK::FFPACK\_LU\_TAG LuTag, const size\_t cutoff)

## 17.195.1 Macro Definition Documentation

### 17.195.1.1 \_\_FFPACK\_ludivine\_mp\_INL

```
#define __FFPACK_ludivine_mp_INL
```

## 17.196 ffpack\_minpoly.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFPACK::Protected](#)

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_minpoly\\_INL](#)

### Functions

- template<class [Field](#) , class Polynomial >  
Polynomial & [MinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda)  
*Compute the minimal polynomial of the matrix A.*
- template<class [Field](#) , class Polynomial , class RandIter >  
Polynomial & [MinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, RandIter &G)  
*Compute the minimal polynomial of the matrix A.*

- template<class [Field](#) , class Polynomial >  
Polynomial & [MatVecMinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::ConstElement\\_ptr](#) v, const size\_t incv)  
*Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis  $(v, Av, \dots, A^N v)$ .*
- template<class [Field](#) , class Polynomial >  
Polynomial & [MatVecMinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) v, const size\_t incv, typename [Field::Element\\_ptr](#) K, const size\_t ldk, size\_t \*P)
- template<class [Field](#) , class Polynomial >  
Polynomial & [Hybrid\\_KGF\\_LUK\\_MinPoly](#) (const [Field](#) &F, Polynomial &minP, const size\_t N, typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) X, const size\_t ldx, size\_t \*P, const FFPACK\_MINPOLY\_TAG MinTag=[FFPACK::FfpackDense](#), const size\_t kg\_mc=0, const size\_t kg\_↵ mb=0, const size\_t kg\_j=0)

## 17.196.1 Macro Definition Documentation

### 17.196.1.1 \_\_FFLASFFPACK\_ffpack\_minpoly\_INL

```
#define __FFLASFFPACK_ffpack_minpoly_INL
```

## 17.197 ffpack\_permutation.inl File Reference

```
#include <givaro/zring.h>
#include "fflas-ffpack/fflas/fflas_fassign.h"
```

## Namespaces

- namespace [FFPACK](#)  
*Finite **Field** **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_permutation\\_INL](#)
- #define [FFLASFFPACK\\_PERM\\_BKSIZE](#) 32

## Functions

- template<class [Field](#) >  
void [MonotonicApplyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const size\_t R)  
*Apply a R-monotonically increasing permutation P, to the matrix A.*
- template<class [Field](#) >  
void [MonotonicCompress](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t R, const size\_t maxpiv, const size\_t rowstomove, const std::vector< bool > &ispiv)
- template<class [Field](#) >  
void [MonotonicCompressMorePivots](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, type-name [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t R, const size\_t rowstomove, const size\_t lenP)
- template<class [Field](#) >  
void [MonotonicCompressCycles](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t incA, const size\_t \*MathP, const size\_t lenP)

- `template<class Field >`  
`void MonotonicExpand (const Field &F, const FFLAS::FFLAS_SIDE Side, const size_t M, typename Field::Element_ptr A, const size_t lda, const size_t incA, const size_t *MathP, const size_t R, const size_t maxpiv, const size_t rowstomove, const std::vector< bool > &ispiv)`
- `template<class Field >`  
`void applyP_block (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t M, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P)`
- `template<class Field >`  
`void applyP (const Field &F, const FFLAS::FFLAS_SIDE Side, const FFLAS::FFLAS_TRANSPOSE Trans, const size_t m, const size_t ibeg, const size_t iend, typename Field::Element_ptr A, const size_t lda, const size_t *P, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field >`  
`void doApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field, class Cut, class Param >`  
`void MatrixApplyS (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyS (T *A, const size_t lda, const size_t width, const size_t M2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void doApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr tmp, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `template<class Field >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Sequential seq)`
- `template<class Field, class Cut, class Param >`  
`void MatrixApplyT (const Field &F, typename Field::Element_ptr A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4, const FFLAS::ParSeqHelper::Parallel< Cut, Param > par)`
- `template<class T >`  
`void PermApplyT (T *A, const size_t lda, const size_t width, const size_t N2, const size_t R1, const size_t R2, const size_t R3, const size_t R4)`
- `void LAPACKPerm2MathPerm (size_t *MathP, const size_t *LapackP, const size_t N)`  
*Conversion of a permutation from LAPACK format to Math format.*
- `void MathPerm2LAPACKPerm (size_t *LapackP, const size_t *MathP, const size_t N)`  
*Conversion of a permutation from Maths format to LAPACK format.*
- `void composePermutationsLLL (size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.*
- `void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)`  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $MathP$  as a MathPermutation format.*

- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)  
*Computes MathP1 x Diag (I\_R, P2) where MathP1 is a MathPermutation and P2 a LAPACK permutation and store the result in MathP1 as a MathPermutation format.*
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- template<class [Field](#) >  
void [cyclic\\_shift\\_row\\_col](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) A, size\_t m, size\_t n, size\_t lda)
- template<class [Field](#) >  
void [cyclic\\_shift\\_row](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) A, size\_t m, size\_t n, size\_t lda)
- template<typename T >  
void [cyclic\\_shift\\_row](#) (const RNSIntegerMod< T > &F, typename [T::Element\\_ptr](#) A, size\_t m, size\_t n, size\_t lda)
- template<class [Field](#) >  
void [cyclic\\_shift\\_col](#) (const [Field](#) &F, typename [Field::Element\\_ptr](#) A, size\_t m, size\_t n, size\_t lda)
- template<typename T >  
void [cyclic\\_shift\\_col](#) (const RNSIntegerMod< T > &F, typename [T::Element\\_ptr](#) A, size\_t m, size\_t n, size\_t lda)
- template<class [Field](#) >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P)  
*Computes P1 x Diag (I\_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in P1 as a LAPACK permutation.*
- template<class [Field](#) , class Cut , class Param >  
void [applyP](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_SIDE](#) Side, const [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t m, const size\_t ibeg, const size\_t iend, typename [Field::Element\\_ptr](#) A, const size\_t lda, const size\_t \*P, const [FFLAS::ParSeqHelper::Parallel](#)< Cut, Param > par)

## 17.197.1 Macro Definition Documentation

### 17.197.1.1 \_\_FFLASFFPACK\_ffpack\_permutation\_INL

```
#define __FFLASFFPACK_ffpack_permutation_INL
```

### 17.197.1.2 FFLASFFPACK\_PERM\_BKSIZE

```
#define FFLASFFPACK_PERM_BKSIZE 32
```

## 17.198 ffpack\_pluq.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_pluq\\_INL](#)
- #define [CROUT](#)

### Functions

- template<class [Field](#) >  
size\_t [PLUQ\\_basecaseV3](#) (const [Field](#) &Fi, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Q)

- template<class [Field](#) >  
size\_t [PLUQ\\_basecaseV2](#) (const [Field](#) &Fi, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element](#) \*A, const size\_t lda, size\_t \*P, size\_t \*Q)
- template<class [Field](#) >  
size\_t [PLUQ\\_basecaseCrout](#) (const [Field](#) &Fi, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q)
- template<class [Field](#) >  
size\_t [\\_PLUQ](#) (const [Field](#) &Fi, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, size\_t BCThreshold)
- template<class [Field](#) >  
size\_t [PLUQ](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFLAS::ParSeqHelper::Sequential](#) &PSHelper, size\_t BCThreshold=[\\_\\_FFLASFFPACK\\_PLUQ\\_THRESHOLD](#))
- template<class [Field](#) >  
size\_t [PLUQ](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q)

*Compute a PLUQ factorization of the given matrix.*

## 17.198.1 Macro Definition Documentation

### 17.198.1.1 [\\_\\_FFLASFFPACK\\_ffpack\\_pluq\\_INL](#)

```
#define \_\_FFLASFFPACK\_ffpack\_pluq\_INL
```

### 17.198.1.2 CROUT

```
#define CROUT
```

## 17.199 [ffpack\\_pluq\\_mp.inl](#) File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
#include "fflas-ffpack/field/rns-integer.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "givaro/givinteger.h"
#include "givaro/modular-integer.h"
```

## Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [\\_\\_FFPACK\\_pluq\\_mp\\_INL](#)

## Functions

- template<class Cut, class Param >  
size\_t [PLUQ](#) (const Givaro::Modular< Givaro::Integer > &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename Givaro::Integer \*A, const size\_t lda, size\_t \*P, size\_t \*Q, size\_t BCThreshold, [FFLAS::ParSeqHelper::Parallel](#)< Cut, Param > &PSHelper)

## 17.199.1 Macro Definition Documentation

### 17.199.1.1 \_\_FFPACK\_pluq\_mp\_INL

```
#define __FFPACK_pluq_mp_INL
```

## 17.200 ffpack\_ppluq.inl File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_ppluq\\_INL](#)
- #define [\\_\\_FFLAS\\_\\_TRSM\\_READONLY](#)
- #define [PBASECASE\\_K](#) 256

### Functions

- template<class [Field](#) >  
void [threads\\_fgemm](#) (const size\_t m, const size\_t n, const size\_t r, int nbthreads, size\_t \*W1, size\_t \*W2, size\_t \*W3, size\_t gamma)
- template<class [Field](#) >  
void [threads\\_frsm](#) (const size\_t m, const size\_t n, int nbthreads, size\_t \*t1, size\_t \*t2)
- template<class [Field](#) >  
size\_t [PLUQ](#) (const [Field](#) &Fi, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q, const [FFLAS::ParSeqHelper::Parallel](#)<[FFLAS::CuttingStrategy::Recursive](#), [FFLAS::StrategyParameter::Threads](#) > &PHelper)
- template<class [Field](#) >  
size\_t [pPLUQ](#) (const [Field](#) &F, const [FFLAS::FFLAS\\_DIAG](#) Diag, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*P, size\_t \*Q)

## 17.200.1 Macro Definition Documentation

### 17.200.1.1 \_\_FFLASFFPACK\_ffpack\_ppluq\_INL

```
#define __FFLASFFPACK_ffpack_ppluq_INL
```

### 17.200.1.2 \_\_FFLAS\_\_TRSM\_READONLY

```
#define __FFLAS__TRSM_READONLY
```

### 17.200.1.3 PBASECASE\_K

```
#define PBASECASE_K 256
```

## 17.201 ffpack\_rankprofiles.inl File Reference

### Namespaces

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_ffpack\\_rank\\_profiles\\_INL](#)

### Functions

- template<class [Field](#) >  
size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))  
*Computes the row rank profile of A.*
- template<class [Field](#) >  
size\_t [pRowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#), class PSHelper >  
size\_t [RowRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)
- template<class [Field](#) >  
size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag=[FfpackSlabRecursive](#))  
*Computes the column rank profile of A.*
- template<class [Field](#) >  
size\_t [pColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, size\_t numthreads=0, const FFPACK\_LU\_TAG LuTag=[FfpackTileRecursive](#))
- template<class [Field](#), class PSHelper >  
size\_t [ColumnRankProfile](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag, PSHelper &psH)
- void [RankProfileFromLU](#) (const size\_t \*P, const size\_t N, const size\_t R, size\_t \*rkprofile, const FFPACK\_LU\_TAG LuTag)  
*Recovers the column/row rank profile from the permutation of an LU decomposition.*
- size\_t [LeadingSubmatrixRankProfiles](#) (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP)  
*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*
- template<class [Field](#) >  
size\_t [RowRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)  
*RowRankProfileSubmatrixIndices.*
- template<class [Field](#) >  
size\_t [ColRankProfileSubmatrixIndices](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, size\_t \*&rowindices, size\_t \*&colindices, size\_t &R)  
*Computes the indices of the submatrix  $r \times r$  X of A whose columns correspond to the column rank profile of A.*
- template<class [Field](#) >  
size\_t [RowRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)  
*Computes the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*
- template<class [Field](#) >  
size\_t [ColRankProfileSubmatrix](#) (const [Field](#) &F, const size\_t M, const size\_t N, typename [Field::Element\\_ptr](#) A, const size\_t lda, typename [Field::Element\\_ptr](#) &X, size\_t &R)  
*Compute the  $r \times r$  submatrix X of A, by picking the row rank profile rows of A.*



- `template<class Field >`  
`Field::Element_ptr LQUPtoInverseOfFullRankMinor (const Field &F, const size_t rank, typename`  
`Field::Element_ptr A_factors, const size_t lda, const size_t *QtPointer, typename Field::Element_ptr X,`  
`const size_t ldx)`  
*LQUPtoInverseOfFullRankMinor.*

## 17.201.1 Macro Definition Documentation

### 17.201.1.1 \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL

```
#define __FFLASFFPACK_ffpack_rank_profiles_INL
```

## 17.202 field-traits.h File Reference

Field Traits.

```
#include <type_traits>
#include "fflas-ffpack/field/rns-double-elt.h"
#include "recint/rmint.h"
#include "givaro/modular-general.h"
#include "givaro/zring.h"
```

## Data Structures

- struct [GenericTag](#)  
*generic ring.*
- struct [ModularTag](#)  
*This is a modular field like e.g. Modular<T> or ModularBalanced<T>*
- struct [UnparametricTag](#)  
*If the field uses a representation with infix operators.*
- struct [DefaultTag](#)  
*No specific mode of action: use standard field operations.*
- struct [DefaultBoundedTag](#)  
*Use standard field operations, but keeps track of bounds on input and output.*
- struct [ConvertTo< T >](#)  
*Force conversion to appropriate element type of ElementCategory T.*
- struct [DelayedTag](#)  
*Performs field operations with delayed mod reductions. Ensures result is reduced.*
- struct [LazyTag](#)  
*Performs field operations with delayed mod only when necessary. Result may not be reduced.*
- struct [GenericTag](#)  
*default is generic*
- struct [MachineFloatTag](#)  
*float or double*
- struct [MachineIntTag](#)  
*short, int, long, long long, and unsigned variants*
- struct [FixedPrecIntTag](#)  
*Fixed precision integers above machine precision: Givaro::reclnt.*
- struct [ArbitraryPrecIntTag](#)  
*Arbitrary precision integers: GMP.*
- struct [RNSElementTag](#)

*Representation in a Residue Number System.*

- struct [ElementTraits](#)< [Element](#) >

*ElementTraits.*

- struct [ElementTraits](#)< float >
- struct [ElementTraits](#)< double >
- struct [ElementTraits](#)< int8\_t >
- struct [ElementTraits](#)< int16\_t >
- struct [ElementTraits](#)< int32\_t >
- struct [ElementTraits](#)< int64\_t >
- struct [ElementTraits](#)< uint8\_t >
- struct [ElementTraits](#)< uint16\_t >
- struct [ElementTraits](#)< uint32\_t >
- struct [ElementTraits](#)< uint64\_t >
- struct [ElementTraits](#)< [Givaro::Integer](#) >
- struct [ElementTraits](#)< [ReclInt::rint](#)< K > >
- struct [ElementTraits](#)< [ReclInt::ruint](#)< K > >
- struct [ElementTraits](#)< [ReclInt::rmint](#)< K, MG > >
- struct [ElementTraits](#)< [FFPACK::rns\\_double\\_elt](#) >
- struct [ModeTraits](#)< [Field](#) >

*ModeTraits.*

- struct [ModeTraits](#)< [Givaro::Modular](#)< [Element](#), [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< int8\_t, [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< int16\_t, [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< int32\_t, [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< uint8\_t, [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< uint16\_t, [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< uint32\_t, [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< [Givaro::Integer](#), [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::Modular](#)< [ReclInt::ruint](#)< K >, [Compute](#) > >
- struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [Element](#) > >
- struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< int8\_t > >
- struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< int16\_t > >
- struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< int32\_t > >
- struct [ModeTraits](#)< [Givaro::ModularBalanced](#)< [Givaro::Integer](#) > >
- struct [ModeTraits](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) > >
- struct [ModeTraits](#)< [Givaro::ZRing](#)< float > >
- struct [ModeTraits](#)< [Givaro::ZRing](#)< double > >
- struct [ModeTraits](#)< [Givaro::Montgomery](#)< T > >
- struct [FieldTraits](#)< [Field](#) >

*FieldTrait.*

- struct [FieldTraits](#)< [Givaro::ZRing](#)< [ReclInt::ruint](#)< K > > >
- struct [FieldTraits](#)< [Givaro::Modular](#)< [Element](#) > >
- struct [FieldTraits](#)< [Givaro::ModularBalanced](#)< [Element](#) > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< double > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< float > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< int16\_t > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< uint16\_t > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< int32\_t > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< uint32\_t > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< int64\_t > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< uint64\_t > >
- struct [FieldTraits](#)< [Givaro::ZRing](#)< [Givaro::Integer](#) > >
- struct [FieldTraits](#)< [FFPACK::RNSInteger](#)< T > >
- struct [FieldTraits](#)< [FFPACK::RNSIntegerMod](#)< T > >

- struct [associatedDelayedField< Field >](#)
- struct [associatedDelayedField< const Givaro::Modular< T, X > >](#)
- struct [associatedDelayedField< const Givaro::ModularBalanced< T > >](#)
- struct [associatedDelayedField< const Givaro::ZRing< T > >](#)
- struct [associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >](#)

## Namespaces

- namespace [Reclnt](#)
- namespace [Givaro](#)
- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFLAS](#)
- namespace [FFLAS::FieldCategories](#)  
*Traits and categories will need to be placed in a proper file later.*
- namespace [FFLAS::ModeCategories](#)  
*Specifies the mode of action for an algorithm w.r.t.*
- namespace [FFLAS::ElementCategories](#)

### 17.202.1 Detailed Description

Field Traits.

## 17.203 field.doxy File Reference

### 17.204 rns-double-elt.h File Reference

rns elt structure with double support

```
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/utils/cast.h"
```

## Data Structures

- struct [rns\\_double\\_elt](#)
- struct [rns\\_double\\_elt\\_ptr](#)
- struct [rns\\_double\\_elt\\_cstptr](#)

## Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Functions

- template<> [rns\\_double\\_elt\\_ptr fflas\\_const\\_cast](#) (rns\_double\_elt\_cstptr x)
- template<> [rns\\_double\\_elt\\_cstptr fflas\\_const\\_cast](#) (rns\_double\_elt\_ptr x)

### 17.204.1 Detailed Description

rns elt structure with double support

## 17.205 rns-double-recint.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_field\\_rns\\_double\\_recint\\_INL](#)

## 17.205.1 Macro Definition Documentation

### 17.205.1.1 \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL

```
#define __FFLASFFPACK_field_rns_double_recint_INL
```

## 17.206 rns-double.h File Reference

rns structure with double support

```
#include <iterator>
#include <vector>
#include <givaro/modular-floating.h>
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include "givaro/modular-extended.h"
#include <recint/ruint.h>
#include "fflas-ffpack/config-blas.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/field/rns-double-elt.h"
#include "rns-double.inl"
#include "rns-double-recint.inl"
```

### Data Structures

- struct [rns\\_double](#)
- struct [rns\\_double\\_extended](#)
- class [rnsRandIter](#)< [RNS](#) >

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*
- namespace [FFLAS](#)

### Macros

- #define [ROUND\\_DOWN](#)(x, s) ((x) & ~((s)-1))

## Functions

- template<> void [fflas\\_delete](#) (FFPACK::rns\_double\_elt\_ptr A)
- template<> void [fflas\\_delete](#) (FFPACK::rns\_double\_elt\_cstptr A)

### 17.206.1 Detailed Description

rns structure with double support

### 17.206.2 Macro Definition Documentation

#### 17.206.2.1 ROUND\_DOWN

```
#define ROUND_DOWN(  
    x,  
    s ) ((x) & ~((s)-1))
```

## 17.207 rns-double.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

## Namespaces

- namespace [FFPACK](#)  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*

## Macros

- #define [\\_\\_FFLASFFPACK\\_field\\_rns\\_double\\_INL](#)

### 17.207.1 Macro Definition Documentation

#### 17.207.1.1 \_\_FFLASFFPACK\_field\_rns\_double\_INL

```
#define __FFLASFFPACK_field_rns_double_INL
```

## 17.208 rns-integer-mod.h File Reference

representation of  $\mathbb{Z}/p\mathbb{Z}$  using RNS representation (note: fixed precision)

```
#include <vector>  
#include <cmath>  
#include <recint/recint.h>  
#include <givaro/modular-integer.h>  
#include <givaro/givinteger.h>  
#include <givaro/udl.h>  
#include "givaro/modular-extended.h"  
#include "fflas-ffpack/field/rns-double.h"  
#include "fflas-ffpack/field/rns-integer.h"  
#include "fflas-ffpack/fflas/fflas_level1.inl"  
#include "fflas-ffpack/fflas/fflas_level2.inl"  
#include "fflas-ffpack/fflas/fflas_level3.inl"  
#include "fflas-ffpack/fflas/fflas_enum.h"
```

```
#include "fflas-ffpack/fflas/fflas_fscal_mp.inl"
```

## Data Structures

- class [RNSIntegerMod< RNS >](#)
- class [RNSIntegerMod< RNS >::RandIter](#)

## Namespaces

- namespace [FFPACK](#)  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*
- namespace [FFLAS](#)

## Functions

- template<> [FFPACK::rns\\_double\\_elt\\_ptr fflas\\_new](#) (const [FFPACK::RNSIntegerMod< FFPACK::rns\\_double >](#) &F, const size\_t m, const Alignment align)
- template<> [FFPACK::rns\\_double\\_elt\\_ptr fflas\\_new](#) (const [FFPACK::RNSIntegerMod< FFPACK::rns\\_double >](#) &F, const size\_t m, const size\_t n, const Alignment align)
- template<typename [RNS](#) >  
void [finit\\_rns](#) (const [FFPACK::RNSIntegerMod< RNS >](#) &F, const size\_t m, const size\_t n, size\_t k, const Givaro::Integer \*B, const size\_t ldb, typename [RNS::Element\\_ptr](#) A)
- template<typename [RNS](#) >  
void [finit\\_trans\\_rns](#) (const [FFPACK::RNSIntegerMod< RNS >](#) &F, const size\_t m, const size\_t n, size\_t k, const Givaro::Integer \*B, const size\_t ldb, typename [RNS::Element\\_ptr](#) A)
- template<typename [RNS](#) >  
void [fconvert\\_rns](#) (const [FFPACK::RNSIntegerMod< RNS >](#) &F, const size\_t m, const size\_t n, Givaro::Integer alpha, Givaro::Integer \*B, const size\_t ldb, typename [RNS::ConstElement\\_ptr](#) A)
- template<typename [RNS](#) >  
void [fconvert\\_trans\\_rns](#) (const [FFPACK::RNSIntegerMod< RNS >](#) &F, const size\_t m, const size\_t n, Givaro::Integer alpha, Givaro::Integer \*B, const size\_t ldb, typename [RNS::ConstElement\\_ptr](#) A)

### 17.208.1 Detailed Description

representation of  $\mathbb{Z}/p\mathbb{Z}$  using RNS representation (note: fixed precision)

## 17.209 rns-integer.h File Reference

representation of  $\mathbb{Z}$  using RNS representation (note: fixed precision)

```
#include <givaro/givinteger.h>
#include "fflas-ffpack/field/rns-double.h"
```

## Data Structures

- class [RNSInteger< RNS >](#)
- class [RNSInteger< RNS >::RandIter](#)

## Namespaces

- namespace [FFPACK](#)  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*
- namespace [FFLAS](#)

## Functions

- `template<> FFPACK::rns_double_elt_ptr fflas_new` (const `FFPACK::RNSInteger< FFPACK::rns_double >` &F, const `size_t` m, const `Alignment` align)
- `template<> FFPACK::rns_double_elt_ptr fflas_new` (const `FFPACK::RNSInteger< FFPACK::rns_double >` &F, const `size_t` m, const `size_t` n, const `Alignment` align)
- `template<typename RNS >`  
void `finit_rns` (const `FFPACK::RNSInteger< RNS >` &F, const `size_t` m, const `size_t` n, `size_t` k, const `Givaro::Integer *B`, const `size_t` ldb, `typename FFPACK::RNSInteger< RNS >::Element_ptr` A)
- `template<typename RNS >`  
void `fconvert_rns` (const `FFPACK::RNSInteger< RNS >` &F, const `size_t` m, const `size_t` n, `Givaro::Integer` alpha, `Givaro::Integer *B`, const `size_t` ldb, `typename FFPACK::RNSInteger< RNS >::ConstElement_ptr` A)

### 17.209.1 Detailed Description

representation of  $\mathbb{Z}$  using RNS representation (note: fixed precision)

## 17.210 rns.h File Reference

### Namespaces

- namespace `FFPACK`  
*Finite Field PACK Set of elimination based routines for dense linear algebra.*

### 17.211 rns.inl File Reference

```
#include "rns-double.h"
#include "rns-integer.h"
#include "rns-integer-mod.h"
```

### Macros

- `#define __FFLASFFPACK_field_rns_INL`

#### 17.211.1 Macro Definition Documentation

##### 17.211.1.1 \_\_FFLASFFPACK\_field\_rns\_INL

```
#define __FFLASFFPACK_field_rns_INL
```

### 17.212 interfaces.doxy File Reference

### 17.213 fflas\_c.h File Reference

```
#include <stdbool.h>
#include <stdlib.h>
#include <inttypes.h>
```

### Macros

- `#define FFLAS_COMPILED`

## Enumerations

- enum `FFLAS_C_ORDER` { `FflasRowMajor` = 101 , `FflasColMajor` = 102 , `FflasRowMajor` = 101 , `FflasColMajor` = 102 }
- Storage by row or col ?*
- enum `FFLAS_C_TRANSPOSE` { `FflasNoTrans` = 111 , `FflasTrans` = 112 , `FflasNoTrans` = 111 , `FflasTrans` = 112 }
- Is matrix transposed ?*
- enum `FFLAS_C_UPLO` { `FflasUpper` = 121 , `FflasLower` = 122 , `FflasUpper` = 121 , `FflasLower` = 122 }
- Is triangular matrix's shape upper ?*
- enum `FFLAS_C_DIAG` { `FflasNonUnit` = 131 , `FflasUnit` = 132 , `FflasNonUnit` = 131 , `FflasUnit` = 132 }
- Is the triangular matrix implicitly unit diagonal ?*
- enum `FFLAS_C_SIDE` { `FflasLeft` = 141 , `FflasRight` = 142 , `FflasLeft` = 141 , `FflasRight` = 142 }
- On what side ?*
- enum `FFLAS_C_BASE` { `FflasDouble` = 151 , `FflasFloat` = 152 , `FflasGeneric` = 153 }
- FFLAS\_C\_BASE determines the type of the element representation for Matrix Mult kernel.*

## Functions

- void `freducein_1_modular_double` (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- void `freduce_1_modular_double` (const double F, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void `fnegin_1_modular_double` (const double F, const size\_t n, double \*X, const size\_t incX, bool positive)
- void `fneg_1_modular_double` (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void `fzero_1_modular_double` (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- bool `fiszero_1_modular_double` (const double p, const size\_t n, const double \*X, const size\_t incX, bool positive)
- bool `fequal_1_modular_double` (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void `fassign_1_modular_double` (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void `fscal_1_modular_double` (const double p, const size\_t n, const double alpha, double \*X, const size\_t incX, bool positive)
- void `fsca_1_modular_double` (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void `faxpy_1_modular_double` (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- double `fdot_1_modular_double` (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void `fswap_1_modular_double` (const double p, const size\_t n, double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void `fadd_1_modular_double` (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void `fsub_1_modular_double` (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void `faddin_1_modular_double` (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void `fsubin_1_modular_double` (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void `fassign_2_modular_double` (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t incB, const double \*A, const size\_t incA, bool positive)
- void `fzero_2_modular_double` (const double p, const size\_t m, const size\_t n, double \*A, const size\_t incA, bool positive)
- bool `fequal_2_modular_double` (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, bool positive)



- bool [fiszero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, bool positive)
- void [fidentity\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, const double d, bool positive)
- void [freducein\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- void [freduce\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*A, const size\_t lda, bool positive)
- void [fnegin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- void [fneg\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*A, const size\_t lda, bool positive)
- void [fscal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t lda, bool positive)
- void [fscal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [faxpy\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*X, const size\_t ldX, double \*Y, const size\_t ldY, bool positive)
- void [fmove\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fadd\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [fsub\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [fsubin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [faddin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- double \* [fgemv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, const double \*X, const size\_t incX, const double beta, double \*Y, const size\_t incY, bool positive)
- void [fger\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*x, const size\_t incX, const double \*y, const size\_t incY, double \*A, const size\_t lda, bool positive)
- void [ftrsv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t n, const double \*A, const size\_t lda, double \*X, int incX, bool positive)
- void [ftrsm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [ftrmm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- double \* [fgemm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_TRANSPOSE](#) tB, const size\_t m, const size\_t n, const size\_t k, const double alpha, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, const double beta, double \*C, const size\_t ldc, bool positive)
- double \* [fsquare\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const size\_t n, const double alpha, const double \*A, const size\_t lda, const double beta, double \*C, const size\_t ldc, bool positive)

### 17.213.1 Macro Definition Documentation

**17.213.1.1 FFLAS\_COMPILED**

```
#define FFLAS_COMPILED
```

**17.213.2 Enumeration Type Documentation****17.213.2.1 FFLAS\_C\_ORDER**

```
enum FFLAS_C_ORDER
```

Storage by row or col ?

Enumerator

|               |           |
|---------------|-----------|
| FflasRowMajor | row major |
| FflasColMajor | col major |
| FflasRowMajor |           |
| FflasColMajor |           |

**17.213.2.2 FFLAS\_C\_TRANSPOSE**

```
enum FFLAS_C_TRANSPOSE
```

Is matrix transposed ?

Enumerator

|              |                           |
|--------------|---------------------------|
| FflasNoTrans | Matrix is not transposed. |
| FflasTrans   | Matrix is transposed.     |
| FflasNoTrans |                           |
| FflasTrans   |                           |

**17.213.2.3 FFLAS\_C\_UPLO**

```
enum FFLAS_C_UPLO
```

Is triangular matrix's shape upper ?

Enumerator

|            |                                                                        |
|------------|------------------------------------------------------------------------|
| FflasUpper | Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$ ) |
| FflasLower | Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$ ) |
| FflasUpper |                                                                        |
| FflasLower |                                                                        |

**17.213.2.4 FFLAS\_C\_DIAG**

```
enum FFLAS_C_DIAG
```

Is the triangular matrix implicitly unit diagonal ?

## Enumerator

|              |                                                                   |
|--------------|-------------------------------------------------------------------|
| FflasNonUnit | Triangular matrix has an explicit arbitrary diagonal.             |
| FflasUnit    | Triangular matrix has an implicit unit diagonal ( $T_{i,i} = 1$ ) |
| FflasNonUnit |                                                                   |
| FflasUnit    |                                                                   |

## 17.213.2.5 FFLAS\_C\_SIDE

enum [FFLAS\\_C\\_SIDE](#)

On what side ?

## Enumerator

|            |                                 |
|------------|---------------------------------|
| FflasLeft  | Operator applied on the left.   |
| FflasRight | Operator applied on the righth. |
| FflasLeft  |                                 |
| FflasRight |                                 |

## 17.213.2.6 FFLAS\_C\_BASE

enum [FFLAS\\_C\\_BASE](#)

FFLAS\_C\_BASE determines the type of the element representation for Matrix Mult kernel.  
(deprecated, should not be used)

## Enumerator

|              |                                                                            |
|--------------|----------------------------------------------------------------------------|
| FflasDouble  | to use the double precision BLAS                                           |
| FflasFloat   | to use the single precision BLAS                                           |
| FflasGeneric | for any other domain, that can not be converted to floating point integers |

## 17.213.3 Function Documentation

## 17.213.3.1 freducein\_1\_modular\_double()

```
void freducein_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

## 17.213.3.2 freduce\_1\_modular\_double()

```
void freduce_1_modular_double (
    const double F,
    const size_t n,
    const double * Y,
```

```
    const size_t incY,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.213.3.3 fnegin\_1\_modular\_double()

```
void fnegin_1_modular_double (  
    const double F,  
    const size_t n,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.213.3.4 fneg\_1\_modular\_double()

```
void fneg_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * Y,  
    const size_t incY,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.213.3.5 fzero\_1\_modular\_double()

```
void fzero_1_modular_double (  
    const double p,  
    const size_t n,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.213.3.6 fiszero\_1\_modular\_double()

```
bool fiszero_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.213.3.7 fequal\_1\_modular\_double()

```
bool fequal_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * X,  
    const size_t incX,  
    const double * Y,  
    const size_t incY,  
    bool positive )
```

**17.213.3.8 fassign\_1\_modular\_double()**

```
void fassign_1_modular_double (
    const double p,
    const size_t n,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX,
    bool positive )
```

**17.213.3.9 fscaln\_1\_modular\_double()**

```
void fscaln_1_modular_double (
    const double p,
    const size_t n,
    const double alpha,
    double * X,
    const size_t incX,
    bool positive )
```

**17.213.3.10 fscale\_1\_modular\_double()**

```
void fscale_1_modular_double (
    const double p,
    const size_t n,
    const double alpha,
    const double * X,
    const size_t incX,
    double * Y,
    const size_t incY,
    bool positive )
```

**17.213.3.11 faxpy\_1\_modular\_double()**

```
void faxpy_1_modular_double (
    const double p,
    const size_t n,
    const double alpha,
    const double * X,
    const size_t incX,
    double * Y,
    const size_t incY,
    bool positive )
```

**17.213.3.12 fdot\_1\_modular\_double()**

```
double fdot_1_modular_double (
    const double p,
    const size_t n,
    const double * X,
    const size_t incX,
    const double * Y,
    const size_t incY,
    bool positive )
```

**17.213.3.13 fswap\_1\_modular\_double()**

```
void fswap_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    double * Y,
    const size_t incY,
    bool positive )
```

**17.213.3.14 fadd\_1\_modular\_double()**

```
void fadd_1_modular_double (
    const double p,
    const size_t n,
    const double * A,
    const size_t incA,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

**17.213.3.15 fsub\_1\_modular\_double()**

```
void fsub_1_modular_double (
    const double p,
    const size_t n,
    const double * A,
    const size_t incA,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

**17.213.3.16 faddin\_1\_modular\_double()**

```
void faddin_1_modular_double (
    const double p,
    const size_t n,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

**17.213.3.17 fsubin\_1\_modular\_double()**

```
void fsubin_1_modular_double (
    const double p,
    const size_t n,
    const double * B,
    const size_t incB,
```

```
double * C,  
const size_t incC,  
bool positive )
```

#### 17.213.3.18 fassign\_2\_modular\_double()

```
void fassign_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * B,  
    const size_t ldB,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.213.3.19 fzero\_2\_modular\_double()

```
void fzero_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.213.3.20 fequal\_2\_modular\_double()

```
bool fequal_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * A,  
    const size_t ldA,  
    const double * B,  
    const size_t ldB,  
    bool positive )
```

#### 17.213.3.21 fiszero\_2\_modular\_double()

```
bool fiszero_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.213.3.22 fidentity\_2\_modular\_double()

```
void fidentity_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,
```

```
double * A,  
const size_t ldA,  
const double d,  
bool positive )
```

#### 17.213.3.23 **freducein\_2\_modular\_double()**

```
void freducein_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.213.3.24 **freduce\_2\_modular\_double()**

```
void freduce_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * B,  
    const size_t ldB,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.213.3.25 **fnegin\_2\_modular\_double()**

```
void fnegin_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.213.3.26 **fneg\_2\_modular\_double()**

```
void fneg_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * B,  
    const size_t ldB,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.213.3.27 **fscaln\_2\_modular\_double()**

```
void fscaln_2_modular_double (  
    const double p,  
    const size_t m,
```



```
const size_t n,  
const double alpha,  
double * A,  
const size_t ldA,  
bool positive )
```

#### 17.213.3.28 fscal\_2\_modular\_double()

```
void fscal_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    const double * A,  
    const size_t ldA,  
    double * B,  
    const size_t ldB,  
    bool positive )
```

#### 17.213.3.29 faxpy\_2\_modular\_double()

```
void faxpy_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    const double * X,  
    const size_t ldX,  
    double * Y,  
    const size_t ldY,  
    bool positive )
```

#### 17.213.3.30 fmove\_2\_modular\_double()

```
void fmove_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    double * A,  
    const size_t ldA,  
    double * B,  
    const size_t ldB,  
    bool positive )
```

#### 17.213.3.31 fadd\_2\_modular\_double()

```
void fadd_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * A,  
    const size_t ldA,  
    const double * B,  
    const size_t ldB,  
    double * C,
```

```
    const size_t ldC,
    bool positive )
```

#### 17.213.3.32 fsub\_2\_modular\_double()

```
void fsub_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t ldA,
    const double * B,
    const size_t ldB,
    double * C,
    const size_t ldC,
    bool positive )
```

#### 17.213.3.33 fsubin\_2\_modular\_double()

```
void fsubin_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * B,
    const size_t ldB,
    double * C,
    const size_t ldC,
    bool positive )
```

#### 17.213.3.34 faddin\_2\_modular\_double()

```
void faddin_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * B,
    const size_t ldB,
    double * C,
    const size_t ldC,
    bool positive )
```

#### 17.213.3.35 fgemv\_2\_modular\_double()

```
double * fgemv_2_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE TransA,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t ldA,
    const double * X,
    const size_t incX,
    const double betA,
    double * Y,
```

```
    const size_t incY,  
    bool positive )
```

#### 17.213.3.36 fger\_2\_modular\_double()

```
void fger_2_modular_double (  
    const double p,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    const double * x,  
    const size_t incX,  
    const double * y,  
    const size_t incY,  
    double * A,  
    const size_t ldA,  
    bool positive )
```

#### 17.213.3.37 ftrsv\_2\_modular\_double()

```
void ftrsv_2_modular_double (  
    const double p,  
    const enum FFLAS_C_UPLO Uplo,  
    const enum FFLAS_C_TRANSPOSE TransA,  
    const enum FFLAS_C_DIAG Diag,  
    const size_t n,  
    const double * A,  
    const size_t ldA,  
    double * X,  
    int incX,  
    bool positive )
```

#### 17.213.3.38 ftrsm\_3\_modular\_double()

```
void ftrsm_3_modular_double (  
    const double p,  
    const enum FFLAS_C_SIDE Side,  
    const enum FFLAS_C_UPLO Uplo,  
    const enum FFLAS_C_TRANSPOSE TransA,  
    const enum FFLAS_C_DIAG Diag,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    const double * A,  
    const size_t ldA,  
    double * B,  
    const size_t ldB,  
    bool positive )
```

#### 17.213.3.39 ftrmm\_3\_modular\_double()

```
void ftrmm_3_modular_double (  
    const double p,  
    const enum FFLAS_C_SIDE Side,  
    const enum FFLAS_C_UPLO Uplo,
```

```

const enum FFLAS_C_TRANSPOSE TransA,
const enum FFLAS_C_DIAG Diag,
const size_t m,
const size_t n,
const double alpha,
double * A,
const size_t ldA,
double * B,
const size_t ldB,
bool positive )

```

#### 17.213.3.40 fgemm\_3\_modular\_double()

```

double * fgemm_3_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE tA,
    const enum FFLAS_C_TRANSPOSE tB,
    const size_t m,
    const size_t n,
    const size_t k,
    const double alpha,
    const double * A,
    const size_t ldA,
    const double * B,
    const size_t ldB,
    const double betA,
    double * C,
    const size_t ldC,
    bool positive )

```

#### 17.213.3.41 fsquare\_3\_modular\_double()

```

double * fsquare_3_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE tA,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t ldA,
    const double betA,
    double * C,
    const size_t ldC,
    bool positive )

```

### 17.214 fflas\_L1\_inst.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L1_inst_implem.inl"

```

## Macros

- `#define __FFLAS_L1_INST_C`
- `#define INST_OR_DECL`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

## 17.214.1 Macro Definition Documentation

### 17.214.1.1 \_\_FFLAS\_L1\_INST\_C

```
#define __FFLAS_L1_INST_C
```

### 17.214.1.2 INST\_OR\_DECL

```
#define INST_OR_DECL
```

### 17.214.1.3 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.214.1.4 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.214.1.5 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.214.1.6 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.214.1.7 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.214.1.8 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

**17.214.1.9 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**17.214.1.10 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**17.215 fflas\_L1\_inst.h File Reference**

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L1_inst_implem.inl"
```

**Macros**

- #define [INST\\_OR\\_DECL](#) <>
- #define [FFLAS\\_FIELD](#) Givaro::ModularBalanced
- #define [FFLAS\\_ELT](#) double
- #define [FFLAS\\_ELT](#) float
- #define [FFLAS\\_ELT](#) int64\_t
- #define [FFLAS\\_FIELD](#) Givaro::Modular
- #define [FFLAS\\_ELT](#) double
- #define [FFLAS\\_ELT](#) float
- #define [FFLAS\\_ELT](#) int64\_t

**17.215.1 Macro Definition Documentation****17.215.1.1 INST\_OR\_DECL**

```
#define INST_OR_DECL <>
```

**17.215.1.2 FFLAS\_FIELD [1/2]**

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

**17.215.1.3 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**17.215.1.4 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**17.215.1.5 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**17.215.1.6 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**17.215.1.7 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**17.215.1.8 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**17.215.1.9 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**17.216 fflas\_L1\_inst\_implem.inl File Reference****Namespaces**

- namespace [FFLAS](#)

**Functions**

- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$freduce\ x \leftarrow x mod F.$$
- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$freduce\ x \leftarrow y mod F.$$
- template [INST\\_OR\\_DECL](#) void [finit](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$finit\ x \leftarrow y mod F.$$
- template [INST\\_OR\\_DECL](#) void [fconvert](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX, const [FFLAS\\_ELT](#) \*Y, const size\_t incY)  

$$fconvert\ x \leftarrow y mod F.$$
- template [INST\\_OR\\_DECL](#) void [fnegin](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$fnegin\ x \leftarrow -x.$$
- template [INST\\_OR\\_DECL](#) void [fneg](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$fneg\ x \leftarrow -y.$$
- template [INST\\_OR\\_DECL](#) void [fzero](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$fzero : A \leftarrow 0.$$
- template [INST\\_OR\\_DECL](#) bool [fiszero](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*X, const size\_t incX)  

$$fiszero : test\ X = 0.$$
- template [INST\\_OR\\_DECL](#) bool [fequal](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t n, const [FFLAS\\_ELT](#) \*X, const size\_t incX, const [FFLAS\\_ELT](#) \*Y, const size\_t incY)  

$$fequal : test\ X = Y.$$
- template [INST\\_OR\\_DECL](#) void [fassign](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t N, const [FFLAS\\_ELT](#) \*Y, const size\_t incY, [FFLAS\\_ELT](#) \*X, const size\_t incX)

- fassign* :  $x \leftarrow y$ .
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` alpha, `FFLAS_ELT` \*X, const size\_t incX)
- fscal*  $x \leftarrow \alpha \cdot x$ .
- template `INST_OR_DECL` void `fscal` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t n, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)
- fscal*  $y \leftarrow \alpha \cdot x$ .
- template `INST_OR_DECL` void `faxpy` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)
- faxpy* :  $y \leftarrow \alpha \cdot x + y$ .
- template `INST_OR_DECL` `FFLAS_ELT` `fdot` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` \*Y, const size\_t incY)
- fdot* :  $y \leftarrow \alpha \cdot x + \beta \cdot y$ .
- template `INST_OR_DECL` void `fswap` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, `FFLAS_ELT` \*X, const size\_t incX, `FFLAS_ELT` \*Y, const size\_t incY)
- fswap* :  $X \leftrightarrow Y$ .
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
  - template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
  - template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)
  - template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t N, const `FFLAS_ELT` \*A, const size\_t inca, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t incb, `FFLAS_ELT` \*C, const size\_t incc)

## 17.217 fflas\_L2\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L2_inst_implem.inl"
```

### Macros

- #define `__FFLAS_L2_INST_C`
- #define `INST_OR_DECL`
- #define `FFLAS_FIELD` Givaro::ModularBalanced
- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT` int64\_t
- #define `FFLAS_FIELD` Givaro::Modular
- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT` int64\_t

### 17.217.1 Macro Definition Documentation



### 17.217.1.1 \_\_FFLAS\_L2\_INST\_C

```
#define __FFLAS_L2_INST_C
```

### 17.217.1.2 INST\_OR\_DECL

```
#define INST_OR_DECL
```

### 17.217.1.3 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.217.1.4 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.217.1.5 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.217.1.6 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.217.1.7 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.217.1.8 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 17.217.1.9 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

### 17.217.1.10 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 17.218 fflas\_L2\_inst.h File Reference

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L2_inst_implem.inl"
```

## Macros

- `#define INST_OR_DECL <>`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

## 17.218.1 Macro Definition Documentation

### 17.218.1.1 INST\_OR\_DECL

```
#define INST_OR_DECL <>
```

### 17.218.1.2 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.218.1.3 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.218.1.4 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.218.1.5 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.218.1.6 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.218.1.7 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 17.218.1.8 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

### 17.218.1.9 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 17.219 fflas\_L2\_inst\_implem.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Functions

- template [INST\\_OR\\_DECL](#) void [fassign](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fassign : A \leftarrow B.$
- template [INST\\_OR\\_DECL](#) void [fzero](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fzero : A \leftarrow 0.$
- template [INST\\_OR\\_DECL](#) bool [fequal](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*B, const size\_t ldb)  
 $fequal : test A = B.$
- template [INST\\_OR\\_DECL](#) bool [fiszero](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fiszero : test A = 0.$
- template [INST\\_OR\\_DECL](#) void [fidentity](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) &d)  
 $creates a diagonal matrix$
- template [INST\\_OR\\_DECL](#) void [fidentity](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $creates a diagonal matrix$
- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $freduce A \leftarrow A mod F.$
- template [INST\\_OR\\_DECL](#) void [freduce](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $freduce A \leftarrow B mod F.$
- template [INST\\_OR\\_DECL](#) void [finit](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $finit A \leftarrow B mod F.$
- template [INST\\_OR\\_DECL](#) void [fnegin](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fnegin A \leftarrow -A.$
- template [INST\\_OR\\_DECL](#) void [fneg](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fneg A \leftarrow -B.$
- template [INST\\_OR\\_DECL](#) void [fscaln](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) alpha, [FFLAS\\_ELT](#) \*A, const size\_t lda)  
 $fscaln A \leftarrow a \cdot A.$
- template [INST\\_OR\\_DECL](#) void [fscal](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)  
 $fscal B \leftarrow a \cdot A.$
- template [INST\\_OR\\_DECL](#) void [faxpy](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*X, const size\_t ldX, [FFLAS\\_ELT](#) \*Y, const size\_t ldY)  
 $faxpy : y \leftarrow \alpha \cdot x + y.$
- template [INST\\_OR\\_DECL](#) void [fmove](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const size\_t m, const size\_t n, [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)  
 $fmove : y \leftarrow \alpha \cdot x + \beta \cdot y.$

- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  
*fadd : matrix addition.*
- template `INST_OR_DECL` void `fsub` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  
*fsub : matrix subtraction.*
- template `INST_OR_DECL` void `fsubin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  
*fsubin  $C = C - B$*
- template `INST_OR_DECL` void `fadd` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  
*fadd : matrix addition with scaling.*
- template `INST_OR_DECL` void `faddin` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` \*B, const size\_t ldb, `FFLAS_ELT` \*C, const size\_t ldc)  
*faddin*
- template `INST_OR_DECL` `FFLAS_ELT` \* `fgemv` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_TRANSPOSE` TransA, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*A, const size\_t lda, const `FFLAS_ELT` \*X, const size\_t incX, const `FFLAS_ELT` beta, `FFLAS_ELT` \*Y, const size\_t incY)  
*finite prime `FFLAS_FIELD`<`FFLAS_ELT`> GEneral Matrix Vector multiplication.*
- template `INST_OR_DECL` void `fger` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const size\_t M, const size\_t N, const `FFLAS_ELT` alpha, const `FFLAS_ELT` \*x, const size\_t incx, const `FFLAS_ELT` \*y, const size\_t incy, `FFLAS_ELT` \*A, const size\_t lda)  
*fger: rank one update of a general matrix*
- template `INST_OR_DECL` void `ftsv` (const `FFLAS_FIELD`< `FFLAS_ELT` > &F, const `FFLAS_UPLO` Uplo, const `FFLAS_TRANSPOSE` TransA, const `FFLAS_DIAG` Diag, const size\_t N, const `FFLAS_ELT` \*A, const size\_t lda, `FFLAS_ELT` \*X, int incX)  
*ftsv: TRIangular System solve with Vector Computes  $X \leftarrow \text{op}(A^{-1})X$*

## 17.220 fflas\_L3\_inst.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
#include "fflas_L3_inst_implem.inl"
```

### Macros

- #define `__FFLAS_L3_INST_C`
- #define `INST_OR_DECL`
- #define `FFLAS_FIELD` Givaro::ModularBalanced
- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT` int64\_t
- #define `FFLAS_FIELD` Givaro::Modular
- #define `FFLAS_ELT` double
- #define `FFLAS_ELT` float
- #define `FFLAS_ELT` int64\_t

## 17.220.1 Macro Definition Documentation

### 17.220.1.1 \_\_FFLAS\_L3\_INST\_C

```
#define __FFLAS_L3_INST_C
```

### 17.220.1.2 INST\_OR\_DECL

```
#define INST_OR_DECL
```

### 17.220.1.3 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.220.1.4 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.220.1.5 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.220.1.6 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.220.1.7 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.220.1.8 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 17.220.1.9 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

### 17.220.1.10 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 17.221 fflas\_L3\_inst.h File Reference

```
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/fflas/fflas_helpers.inl"
```

```
#include "fflas_L3_inst_implem.inl"
```

## Macros

- `#define INST_OR_DECL <>`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

## 17.221.1 Macro Definition Documentation

### 17.221.1.1 INST\_OR\_DECL

```
#define INST_OR_DECL <>
```

### 17.221.1.2 FFLAS\_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

### 17.221.1.3 FFLAS\_ELT [1/6]

```
#define FFLAS_ELT double
```

### 17.221.1.4 FFLAS\_ELT [2/6]

```
#define FFLAS_ELT float
```

### 17.221.1.5 FFLAS\_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

### 17.221.1.6 FFLAS\_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

### 17.221.1.7 FFLAS\_ELT [4/6]

```
#define FFLAS_ELT double
```

### 17.221.1.8 FFLAS\_ELT [5/6]

```
#define FFLAS_ELT float
```

## 17.221.1.9 FFLAS\_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

## 17.222 fflas\_L3\_inst\_implem.inl File Reference

## Namespaces

- namespace [FFLAS](#)

## Macros

- #define [\\_\\_FFLAS\\_\\_TRSM\\_READONLY](#)

## Functions

- template [INST\\_OR\\_DECL](#) void [ftrsm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)

*ftrsm: **TR**iangular **S**ystem solve with **M**atrix.*

- template [INST\\_OR\\_DECL](#) void [ftrmm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_SIDE Side, const FFLAS\_UPLO Uplo, const FFLAS\_TRANSPOSE TransA, const FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, [FFLAS\\_ELT](#) \*B, const size\_t ldb)

*ftrmm: **TR**iangular **M**atrix **M**ultiply.*

- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fgemm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, const [FFLAS\\_ELT](#) beta, [FFLAS\\_ELT](#) \*C, const size\_t ldc)

*fgemm: **F**ield **GE**neral **M**atrix **M**ultiply.*

- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fgemm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, const [FFLAS\\_ELT](#) beta, [FFLAS\\_ELT](#) \*C, const size\_t ldc, const ParSeqHelper::Sequential seq)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fgemm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, const [FFLAS\\_ELT](#) beta, [FFLAS\\_ELT](#) \*C, const size\_t ldc, const ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive > par)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fgemm](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_TRANSPOSE ta, const FFLAS\_TRANSPOSE tb, const size\_t m, const size\_t n, const size\_t k, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) \*B, const size\_t ldb, const [FFLAS\\_ELT](#) beta, [FFLAS\\_ELT](#) \*C, const size\_t ldc, const ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > par)
- template [INST\\_OR\\_DECL](#) [FFLAS\\_ELT](#) \* [fsquare](#) (const [FFLAS\\_FIELD](#)< [FFLAS\\_ELT](#) > &F, const FFLAS\_TRANSPOSE ta, const size\_t n, const [FFLAS\\_ELT](#) alpha, const [FFLAS\\_ELT](#) \*A, const size\_t lda, const [FFLAS\\_ELT](#) beta, [FFLAS\\_ELT](#) \*C, const size\_t ldc)

*fsquare: Squares a matrix.*

## 17.222.1 Macro Definition Documentation

## 17.222.1.1 \_\_FFLAS\_\_TRSM\_READONLY

```
#define __FFLAS__TRSM_READONLY
```

## 17.223 fflas\_lvl1.C File Reference

C functions calls for level 1 [FFLAS](#) in fflas-c.h.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro//modular-balanced.h"
#include "givaro//modular.h"
```

### Functions

- void [freducein\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- void [freduce\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fnegin\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- void [fneg\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fzero\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, bool positive)
- bool [fiszero\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, bool positive)
- bool [fequal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void [fassign\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*Y, const size\_t incY, double \*X, const size\_t incX, bool positive)
- void [fscal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, double \*X, const size\_t incX, bool positive)
- void [fscal\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [faxpy\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double alpha, const double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- double [fdot\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, bool positive)
- void [fswap\\_1\\_modular\\_double](#) (const double p, const size\_t n, double \*X, const size\_t incX, double \*Y, const size\_t incY, bool positive)
- void [fadd\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsub\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*A, const size\_t incA, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [faddin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)
- void [fsubin\\_1\\_modular\\_double](#) (const double p, const size\_t n, const double \*B, const size\_t incB, double \*C, const size\_t incC, bool positive)

### 17.223.1 Detailed Description

C functions calls for level 1 [FFLAS](#) in fflas-c.h.

Author

Brice Boyer

See also

[fflas/fflas\\_level1.inl](#)

### 17.223.2 Function Documentation



**17.223.2.1 freducein\_1\_modular\_double()**

```
void freducein_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

**17.223.2.2 freduce\_1\_modular\_double()**

```
void freduce_1_modular_double (
    const double p,
    const size_t n,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX,
    bool positive )
```

**17.223.2.3 fnegin\_1\_modular\_double()**

```
void fnegin_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

**17.223.2.4 fneg\_1\_modular\_double()**

```
void fneg_1_modular_double (
    const double p,
    const size_t n,
    const double * Y,
    const size_t incY,
    double * X,
    const size_t incX,
    bool positive )
```

**17.223.2.5 fzero\_1\_modular\_double()**

```
void fzero_1_modular_double (
    const double p,
    const size_t n,
    double * X,
    const size_t incX,
    bool positive )
```

**17.223.2.6 fiszero\_1\_modular\_double()**

```
bool fiszero_1_modular_double (
    const double p,
    const size_t n,
```

```
    const double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.223.2.7 fequal\_1\_modular\_double()

```
bool fequal_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * X,  
    const size_t incX,  
    const double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.223.2.8 fassign\_1\_modular\_double()

```
void fassign_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * Y,  
    const size_t incY,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.223.2.9 fscaln\_1\_modular\_double()

```
void fscaln_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,  
    double * X,  
    const size_t incX,  
    bool positive )
```

#### 17.223.2.10 fscal\_1\_modular\_double()

```
void fscal_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,  
    const double * X,  
    const size_t incX,  
    double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.223.2.11 faxpy\_1\_modular\_double()

```
void faxpy_1_modular_double (  
    const double p,  
    const size_t n,  
    const double alpha,
```

```
const double * X,  
const size_t incX,  
double * Y,  
const size_t incY,  
bool positive )
```

#### 17.223.2.12 fdot\_1\_modular\_double()

```
double fdot_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * X,  
    const size_t incX,  
    const double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.223.2.13 fswap\_1\_modular\_double()

```
void fswap_1_modular_double (  
    const double p,  
    const size_t n,  
    double * X,  
    const size_t incX,  
    double * Y,  
    const size_t incY,  
    bool positive )
```

#### 17.223.2.14 fadd\_1\_modular\_double()

```
void fadd_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * A,  
    const size_t incA,  
    const double * B,  
    const size_t incB,  
    double * C,  
    const size_t incC,  
    bool positive )
```

#### 17.223.2.15 fsub\_1\_modular\_double()

```
void fsub_1_modular_double (  
    const double p,  
    const size_t n,  
    const double * A,  
    const size_t incA,  
    const double * B,  
    const size_t incB,  
    double * C,  
    const size_t incC,  
    bool positive )
```

**17.223.2.16 faddin\_1\_modular\_double()**

```
void faddin_1_modular_double (
    const double p,
    const size_t n,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

**17.223.2.17 fsubin\_1\_modular\_double()**

```
void fsubin_1_modular_double (
    const double p,
    const size_t n,
    const double * B,
    const size_t incB,
    double * C,
    const size_t incC,
    bool positive )
```

**17.224 fflas\_lvl2.C File Reference**

C functions calls for level 2 [FFLAS](#) in `fflas-c.h`.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro/modular-balanced.h"
#include "givaro/modular.h"
```

**Functions**

- void [fassign\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fzero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- bool [fequal\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, bool positive)
- bool [fiszero\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, bool positive)
- void [fidentity\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, const double d, bool positive)
- void [freducein\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- void [freduce\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fnegin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, bool positive)
- void [fneg\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fscalin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t lda, bool positive)
- void [fscale\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [faxpy\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)

- void [fmove\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, double \*A, const size\_t lda, double \*B, const size\_t ldb, bool positive)
- void [fadd\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [fsub\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*A, const size\_t lda, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [fsubin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- void [faddin\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double \*B, const size\_t ldb, double \*C, const size\_t ldc, bool positive)
- double \* [fgemv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t lda, const double \*X, const size\_t incX, const double beta, double \*Y, const size\_t incY, bool positive)
- void [fger\\_2\\_modular\\_double](#) (const double p, const size\_t m, const size\_t n, const double alpha, const double \*X, const size\_t incX, const double \*Y, const size\_t incY, double \*A, const size\_t lda, bool positive)
- void [ftrsv\\_2\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) TransA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t n, const double \*A, const size\_t lda, double \*X, int incX, bool positive)

### 17.224.1 Detailed Description

C functions calls for level 2 [FFLAS](#) in fflas-c.h.

Author

Brice Boyer

See also

[fflas/fflas\\_level2.inl](#)

### 17.224.2 Function Documentation

#### 17.224.2.1 fassign\_2\_modular\_double()

```
void fassign_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    double * B,
    const size_t ldb,
    bool positive )
```

#### 17.224.2.2 fzero\_2\_modular\_double()

```
void fzero_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t lda,
    bool positive )
```

### 17.224.2.3 fequal\_2\_modular\_double()

```
bool fequal_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    const double * B,
    const size_t ldb,
    bool positive )
```

### 17.224.2.4 fiszero\_2\_modular\_double()

```
bool fiszero_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    bool positive )
```

### 17.224.2.5 fidentity\_2\_modular\_double()

```
void fidentity_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t lda,
    const double d,
    bool positive )
```

### 17.224.2.6 freducein\_2\_modular\_double()

```
void freducein_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t lda,
    bool positive )
```

### 17.224.2.7 freduce\_2\_modular\_double()

```
void freduce_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    double * B,
    const size_t ldb,
    bool positive )
```

**17.224.2.8 fnegin\_2\_modular\_double()**

```
void fnegin_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    double * A,
    const size_t lda,
    bool positive )
```

**17.224.2.9 fneg\_2\_modular\_double()**

```
void fneg_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * A,
    const size_t lda,
    double * B,
    const size_t ldb,
    bool positive )
```

**17.224.2.10 fscaln\_2\_modular\_double()**

```
void fscaln_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    double * A,
    const size_t lda,
    bool positive )
```

**17.224.2.11 fscale\_2\_modular\_double()**

```
void fscale_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    double * B,
    const size_t ldb,
    bool positive )
```

**17.224.2.12 faxpy\_2\_modular\_double()**

```
void faxpy_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
```

```
double * B,  
const size_t ldb,  
bool positive )
```

#### 17.224.2.13 fmove\_2\_modular\_double()

```
void fmove_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    double * A,  
    const size_t lda,  
    double * B,  
    const size_t ldb,  
    bool positive )
```

#### 17.224.2.14 fadd\_2\_modular\_double()

```
void fadd_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * A,  
    const size_t lda,  
    const double * B,  
    const size_t ldb,  
    double * C,  
    const size_t ldc,  
    bool positive )
```

#### 17.224.2.15 fsub\_2\_modular\_double()

```
void fsub_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * A,  
    const size_t lda,  
    const double * B,  
    const size_t ldb,  
    double * C,  
    const size_t ldc,  
    bool positive )
```

#### 17.224.2.16 fsubin\_2\_modular\_double()

```
void fsubin_2_modular_double (   
    const double p,  
    const size_t m,  
    const size_t n,  
    const double * B,  
    const size_t ldb,  
    double * C,  
    const size_t ldc,  
    bool positive )
```



**17.224.2.17 faddin\_2\_modular\_double()**

```
void faddin_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double * B,
    const size_t ldb,
    double * C,
    const size_t ldc,
    bool positive )
```

**17.224.2.18 fgemv\_2\_modular\_double()**

```
double * fgemv_2_modular_double (
    const double p,
    const enum FFLAS_C_TRANSPOSE TransA,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
    const size_t lda,
    const double * X,
    const size_t incX,
    const double beta,
    double * Y,
    const size_t incY,
    bool positive )
```

**17.224.2.19 fger\_2\_modular\_double()**

```
void fger_2_modular_double (
    const double p,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * X,
    const size_t incX,
    const double * Y,
    const size_t incY,
    double * A,
    const size_t lda,
    bool positive )
```

**17.224.2.20 ftrsv\_2\_modular\_double()**

```
void ftrsv_2_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_TRANSPOSE TransA,
    const enum FFLAS_C_DIAG Diag,
    const size_t n,
    const double * A,
    const size_t lda,
```

```
double * X,
int incX,
bool positive )
```

## 17.225 fflas\_lvl3.C File Reference

C functions calls for level 3 [FFLAS](#) in fflas-c.h.

```
#include "fflas-ffpack/interfaces/libs/fflas_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "givaro/modular-balanced.h"
#include "givaro/modular.h"
```

### Functions

- void [ftrsm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, const double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- void [ftrmm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_SIDE](#) Side, const enum [FFLAS\\_C\\_UPLO](#) Uplo, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t m, const size\_t n, const double alpha, double \*A, const size\_t ldA, double \*B, const size\_t ldB, bool positive)
- double \* [fgemm\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const enum [FFLAS\\_C\\_TRANSPOSE](#) tB, const size\_t m, const size\_t n, const size\_t k, const double alpha, const double \*A, const size\_t ldA, const double \*B, const size\_t ldB, const double betA, double \*C, const size\_t ldC, bool positive)
- double \* [fsquare\\_3\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_TRANSPOSE](#) tA, const size\_t n, const double alpha, const double \*A, const size\_t ldA, const double betA, double \*C, const size\_t ldC, bool positive)

### 17.225.1 Detailed Description

C functions calls for level 3 [FFLAS](#) in fflas-c.h.

Author

Brice Boyer

See also

[fflas/fflas\\_level3.inl](#)

### 17.225.2 Function Documentation

#### 17.225.2.1 ftrsm\_3\_modular\_double()

```
void ftrsm_3_modular_double (
    const double p,
    const enum FFLAS\_C\_SIDE Side,
    const enum FFLAS\_C\_UPLO Uplo,
    const enum FFLAS\_C\_TRANSPOSE tA,
    const enum FFLAS\_C\_DIAG Diag,
    const size_t m,
    const size_t n,
    const double alpha,
    const double * A,
```

```
    const size_t ldA,  
    double * B,  
    const size_t ldB,  
    bool positive )
```

#### 17.225.2.2 ftrmm\_3\_modular\_double()

```
void ftrmm_3_modular_double (   
    const double p,  
    const enum FFLAS_C_SIDE Side,  
    const enum FFLAS_C_UPLO Uplo,  
    const enum FFLAS_C_TRANSPOSE tA,  
    const enum FFLAS_C_DIAG Diag,  
    const size_t m,  
    const size_t n,  
    const double alpha,  
    double * A,  
    const size_t ldA,  
    double * B,  
    const size_t ldB,  
    bool positive )
```

#### 17.225.2.3 fgemv\_3\_modular\_double()

```
double * fgemv_3_modular_double (   
    const double p,  
    const enum FFLAS_C_TRANSPOSE tA,  
    const enum FFLAS_C_TRANSPOSE tB,  
    const size_t m,  
    const size_t n,  
    const size_t k,  
    const double alpha,  
    const double * A,  
    const size_t ldA,  
    const double * B,  
    const size_t ldB,  
    const double betaA,  
    double * C,  
    const size_t ldC,  
    bool positive )
```

#### 17.225.2.4 fsquare\_3\_modular\_double()

```
double * fsquare_3_modular_double (   
    const double p,  
    const enum FFLAS_C_TRANSPOSE tA,  
    const size_t n,  
    const double alpha,  
    const double * A,  
    const size_t ldA,  
    const double betaA,  
    double * C,  
    const size_t ldC,  
    bool positive )
```

## 17.226 fflas\_sparse.C File Reference

C functions calls for level 1.5 and 2.5 [FFLAS](#) in fflas-c.h.

### 17.226.1 Detailed Description

C functions calls for level 1.5 and 2.5 [FFLAS](#) in fflas-c.h.

Author

Brice Boyer

See also

[fflas/fflas\\_sparse.h](#)

## 17.227 ffpack.C File Reference

C functions calls for [FFPACK](#) in ffpack-c.h.

```
#include "fflas-ffpack/interfaces/libs/ffpack_c.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "givaro/modular-balanced.h"
#include "givaro/modular.h"
```

### Functions

- void [LAPACKPerm2MathPerm](#) (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)
- void [MathPerm2LAPACKPerm](#) (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)
- void [MatrixApplyS\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyS\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void [MatrixApplyT\\_modular\\_double](#) (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void [PermApplyT\\_double](#) (double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void [composePermutationsLLM](#) (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsLLL](#) (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [composePermutationsMLM](#) (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)
- void [cyclic\\_shift\\_mathPerm](#) (size\_t \*P, const size\_t s)
- void [cyclic\\_shift\\_row\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [cyclic\\_shift\\_col\\_modular\\_double](#) (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void [applyP\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const enum [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const size\_t M, const size\_t ibeg, const size\_t iend, double \*A, const size\_t lda, const size\_t \*P, bool positive)
- void [fgetrsin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*B, const size\_t ldb, int \*info, bool positive)
- double \* [fgetrsv\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*X, const size\_t idx, const double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t [fgesvin\\_modular\\_double](#) (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*B, const size\_t ldb, int \*info, bool positive)

- `size_t fgesv_modular_double` (const double p, const enum [FFLAS::FFLAS\\_SIDE](#) Side, const `size_t` M, const `size_t` N, const `size_t` NRHS, double \*A, const `size_t` lda, double \*X, const `size_t` ldx, const double \*B, const `size_t` ldb, int \*info, bool positive)
- `void ftrtri_modular_double` (const double p, const enum [FFLAS::FFLAS\\_UPLO](#) Uplo, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `void trinv_left_modular_double` (const double p, const `size_t` N, const double \*L, const `size_t` ldl, double \*X, const `size_t` ldx, bool positive)
- `void ftrrm_modular_double` (const double p, const [FFLAS::FFLAS\\_SIDE](#) side, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const `size_t` N, double \*A, const `size_t` lda, bool positive)
- `size_t PLUQ_modular_double` (const double p, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, bool positive)
- `size_t LUdivine_modular_double` (const double p, const enum [FFLAS::FFLAS\\_DIAG](#) Diag, const enum [FFLAS::FFLAS\\_TRANSPOSE](#) Trans, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, const `size_t` cutoff, bool positive)
- `size_t ColumnEchelonForm_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t RowEchelonForm_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t ReducedColumnEchelonForm_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t ReducedRowEchelonForm_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t ColumnEchelonForm_modular_float` (const float p, const `size_t` M, const `size_t` N, float \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t RowEchelonForm_modular_float` (const float p, const `size_t` M, const `size_t` N, float \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t ReducedColumnEchelonForm_modular_float` (const float p, const `size_t` M, const `size_t` N, float \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t ReducedRowEchelonForm_modular_float` (const float p, const `size_t` M, const `size_t` N, float \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t ColumnEchelonForm_modular_int32_t` (const [int32\\_t](#) p, const `size_t` M, const `size_t` N, [int32\\_t](#) \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t RowEchelonForm_modular_int32_t` (const [int32\\_t](#) p, const `size_t` M, const `size_t` N, [int32\\_t](#) \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t ReducedColumnEchelonForm_modular_int32_t` (const [int32\\_t](#) p, const `size_t` M, const `size_t` N, [int32\\_t](#) \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t ReducedRowEchelonForm_modular_int32_t` (const [int32\\_t](#) p, const `size_t` M, const `size_t` N, [int32\\_t](#) \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t pColumnEchelonForm_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t pRowEchelonForm_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t pReducedColumnEchelonForm_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t pReducedRowEchelonForm_modular_double` (const double p, const `size_t` M, const `size_t` N, double \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t pColumnEchelonForm_modular_float` (const float p, const `size_t` M, const `size_t` N, float \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- `size_t pRowEchelonForm_modular_float` (const float p, const `size_t` M, const `size_t` N, float \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)

- `size_t pReducedColumnEchelonForm_modular_float` (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t pReducedRowEchelonForm_modular_float` (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t pColumnEchelonForm_modular_int32_t` (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t pRowEchelonForm_modular_int32_t` (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t pReducedColumnEchelonForm_modular_int32_t` (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t pReducedRowEchelonForm_modular_int32_t` (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `double * Invertin_modular_double` (const double p, const size\_t M, double \*A, const size\_t lda, int \*nullity, bool positive)
- `double * Invert_modular_double` (const double p, const size\_t M, const double \*A, const size\_t lda, double \*X, const size\_t idx, int \*nullity, bool positive)
- `double * Invert2_modular_double` (const double p, const size\_t M, double \*A, const size\_t lda, double \*X, const size\_t idx, int \*nullity, bool positive)
- `size_t KrylovElim_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const size\_t deg, size\_t \*iterates, size\_t \*inviterates, const size\_t maxit, size\_t virt, bool positive)
- `size_t SpecRankProfile_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, const size\_t deg, size\_t \*rankProfile, bool positive)
- `size_t Rank_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, bool positive)
- `bool IsSingular_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, bool positive)
- `double Det_modular_double` (const double p, const size\_t N, double \*A, const size\_t lda, bool positive)
- `double * Solve_modular_double` (const double p, const size\_t M, double \*A, const size\_t lda, double \*x, const int incx, const double \*b, const int incb, bool positive)
- `void solveLB_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const size\_t M, const size\_t N, const size\_t R, double \*L, const size\_t ldl, const size\_t \*Q, double \*B, const size\_t ldb, bool positive)
- `void solveLB2_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const size\_t M, const size\_t N, const size\_t R, double \*L, const size\_t ldl, const size\_t \*Q, double \*B, const size\_t ldb, bool positive)
- `void RandomNullSpaceVector_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*X, const size\_t incX, bool positive)
- `size_t NullSpaceBasis_modular_double` (const double p, const enum `FFLAS::FFLAS_SIDE` Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*\*NS, size\_t \*ldn, size\_t \*NSdim, bool positive)
- `size_t RowRankProfile_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `size_t ColumnRankProfile_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag, bool positive)
- `void RankProfileFromLU` (const size\_t \*P, const size\_t N, const size\_t R, size\_t \*rkprofile, const enum `FFPACK_C_LU_TAG` LuTag)
- `size_t LeadingSubmatrixRankProfiles` (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP)
- `size_t RowRankProfileSubmatrixIndices_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rowindices, size\_t \*\*colindices, size\_t \*R, bool positive)
- `size_t ColRankProfileSubmatrixIndices_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rowindices, size\_t \*\*colindices, size\_t \*R, bool positive)
- `size_t RowRankProfileSubmatrix_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*\*X, size\_t \*R, bool positive)

- `size_t ColRankProfileSubmatrix_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*\*X, size\_t \*R, bool positive)
- void `getTriangular_modular_double` (const double p, const enum FFLAS::FFLAS\_UPLO Uplo, const enum FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const size\_t R, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, bool positive)
- void `getTriangularin_modular_double` (const double p, const enum FFLAS::FFLAS\_UPLO Uplo, const enum FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, bool positive)
- void `getEchelonForm_modular_double` (const double p, const enum FFLAS::FFLAS\_UPLO Uplo, const enum FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- void `getEchelonFormin_modular_double` (const double p, const enum FFLAS::FFLAS\_UPLO Uplo, const enum FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- void `getEchelonTransform_modular_double` (const double p, const enum FFLAS::FFLAS\_UPLO Uplo, const enum FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- void `getReducedEchelonForm_modular_double` (const double p, const enum FFLAS::FFLAS\_UPLO Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- void `getReducedEchelonFormin_modular_double` (const double p, const enum FFLAS::FFLAS\_UPLO Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- void `getReducedEchelonTransform_modular_double` (const double p, const enum FFLAS::FFLAS\_UPLO Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- void `PLUQtoEchelonPermutation` (const size\_t N, const size\_t R, const size\_t \*P, size\_t \*outPerm)

## 17.227.1 Detailed Description

C functions calls for [FFPACK](#) in `ffpack-c.h`.

Author

Brice Boyer

See also

[ffpack/ffpack.h](#)

## 17.227.2 Function Documentation

### 17.227.2.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
    size_t * MathP,
    const size_t * LapackP,
    const size_t N )
```

### 17.227.2.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
    size_t * LapackP,
    const size_t * MathP,
    const size_t N )
```

### 17.227.2.3 MatrixApplyS\_modular\_double()

```
void MatrixApplyS_modular_double (
    const double p,
    double * A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    bool positive )
```

### 17.227.2.4 PermApplyS\_double()

```
void PermApplyS_double (
    double * A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 )
```

### 17.227.2.5 MatrixApplyT\_modular\_double()

```
void MatrixApplyT_modular_double (
    const double p,
    double * A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    bool positive )
```

### 17.227.2.6 PermApplyT\_double()

```
void PermApplyT_double (
    double * A,
    const size_t lda,
    const size_t width,
    const size_t N2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 )
```

### 17.227.2.7 composePermutationsLLM()

```
void composePermutationsLLM (
    size_t * MathP,
```



```
const size_t * P1,  
const size_t * P2,  
const size_t R,  
const size_t N )
```

#### 17.227.2.8 composePermutationsLLL()

```
void composePermutationsLLL (  
    size_t * P1,  
    const size_t * P2,  
    const size_t R,  
    const size_t N )
```

#### 17.227.2.9 composePermutationsMLM()

```
void composePermutationsMLM (  
    size_t * MathP1,  
    const size_t * P2,  
    const size_t R,  
    const size_t N )
```

#### 17.227.2.10 cyclic\_shift\_mathPerm()

```
void cyclic_shift_mathPerm (  
    size_t * P,  
    const size_t s )
```

#### 17.227.2.11 cyclic\_shift\_row\_modular\_double()

```
void cyclic_shift_row_modular_double (  
    const double p,  
    double * A,  
    size_t m,  
    size_t n,  
    size_t lda,  
    bool positive )
```

#### 17.227.2.12 cyclic\_shift\_col\_modular\_double()

```
void cyclic_shift_col_modular_double (  
    const double p,  
    double * A,  
    size_t m,  
    size_t n,  
    size_t lda,  
    bool positive )
```

#### 17.227.2.13 applyP\_modular\_double()

```
void applyP_modular_double (  
    const double p,  
    const enum FFLAS::FFLAS_SIDE Side,  
    const enum FFLAS::FFLAS_TRANSPOSE Trans,
```

```

    const size_t M,
    const size_t ibeg,
    const size_t iend,
    double * A,
    const size_t lda,
    const size_t * P,
    bool positive )

```

#### 17.227.2.14 fgetrsin\_modular\_double()

```

void fgetrsin_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    double * A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    double * B,
    const size_t ldb,
    int * info,
    bool positive )

```

#### 17.227.2.15 fgetrsv\_modular\_double()

```

double * fgetrsv_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    const size_t R,
    double * A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    double * X,
    const size_t ldx,
    const double * B,
    const size_t ldb,
    int * info,
    bool positive )

```

#### 17.227.2.16 fgesvin\_modular\_double()

```

size_t fgesvin_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double * B,
    const size_t ldb,

```

```
int * info,  
bool positive )
```

#### 17.227.2.17 fgesv\_modular\_double()

```
size_t fgesv_modular_double (  
    const double p,  
    const enum FFLAS::FFLAS_SIDE Side,  
    const size_t M,  
    const size_t N,  
    const size_t NRHS,  
    double * A,  
    const size_t lda,  
    double * X,  
    const size_t ldx,  
    const double * B,  
    const size_t ldb,  
    int * info,  
    bool positive )
```

#### 17.227.2.18 ftrtri\_modular\_double()

```
void ftrtri_modular_double (  
    const double p,  
    const enum FFLAS::FFLAS_UPLO Uplo,  
    const enum FFLAS::FFLAS_DIAG Diag,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```

#### 17.227.2.19 trinv\_left\_modular\_double()

```
void trinv_left_modular_double (  
    const double p,  
    const size_t N,  
    const double * L,  
    const size_t ldl,  
    double * X,  
    const size_t ldx,  
    bool positive )
```

#### 17.227.2.20 ftrtrm\_modular\_double()

```
void ftrtrm_modular_double (  
    const double p,  
    const FFLAS::FFLAS_SIDE side,  
    const enum FFLAS::FFLAS_DIAG Diag,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```

**17.227.2.21 PLUQ\_modular\_double()**

```

size_t PLUQ_modular_double (
    const double p,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    bool positive )

```

**17.227.2.22 LUdivine\_modular\_double()**

```

size_t LUdivine_modular_double (
    const double p,
    const enum FFLAS::FFLAS_DIAG Diag,
    const enum FFLAS::FFLAS_TRANSPOSE Trans,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const enum FFPACK_C_LU_TAG LuTag,
    const size_t cutoff,
    bool positive )

```

**17.227.2.23 ColumnEchelonForm\_modular\_double()**

```

size_t ColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.227.2.24 RowEchelonForm\_modular\_double()**

```

size_t RowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.227.2.25 ReducedColumnEchelonForm\_modular\_double()**

```
size_t ReducedColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.227.2.26 ReducedRowEchelonForm\_modular\_double()**

```
size_t ReducedRowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.227.2.27 ColumnEchelonForm\_modular\_float()**

```
size_t ColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.227.2.28 RowEchelonForm\_modular\_float()**

```
size_t RowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
```

```
const enum FFPACK_C_LU_TAG LuTag,
bool positive )
```

#### 17.227.2.29 ReducedColumnEchelonForm\_modular\_float()

```
size_t ReducedColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

#### 17.227.2.30 ReducedRowEchelonForm\_modular\_float()

```
size_t ReducedRowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

#### 17.227.2.31 ColumnEchelonForm\_modular\_int32\_t()

```
size_t ColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

#### 17.227.2.32 RowEchelonForm\_modular\_int32\_t()

```
size_t RowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
```

```
const bool transform,  
const enum FFPACK_C_LU_TAG LuTag,  
bool positive )
```

#### 17.227.2.33 ReducedColumnEchelonForm\_modular\_int32\_t()

```
size_t ReducedColumnEchelonForm_modular_int32_t (  
    const int32_t p,  
    const size_t M,  
    const size_t N,  
    int32_t * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.227.2.34 ReducedRowEchelonForm\_modular\_int32\_t()

```
size_t ReducedRowEchelonForm_modular_int32_t (  
    const int32_t p,  
    const size_t M,  
    const size_t N,  
    int32_t * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.227.2.35 pColumnEchelonForm\_modular\_double()

```
size_t pColumnEchelonForm_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.227.2.36 pRowEchelonForm\_modular\_double()

```
size_t pRowEchelonForm_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,
```

```

size_t * Qt,
const bool transform,
const enum FFPACK_C_LU_TAG LuTag,
bool positive )

```

#### 17.227.2.37 pReducedColumnEchelonForm\_modular\_double()

```

size_t pReducedColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.38 pReducedRowEchelonForm\_modular\_double()

```

size_t pReducedRowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.39 pColumnEchelonForm\_modular\_float()

```

size_t pColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.40 pRowEchelonForm\_modular\_float()

```

size_t pRowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,

```



```
size_t * P,  
size_t * Qt,  
const bool transform,  
const enum FFPACK_C_LU_TAG LuTag,  
bool positive )
```

#### 17.227.2.41 pReducedColumnEchelonForm\_modular\_float()

```
size_t pReducedColumnEchelonForm_modular_float (  
    const float p,  
    const size_t M,  
    const size_t N,  
    float * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.227.2.42 pReducedRowEchelonForm\_modular\_float()

```
size_t pReducedRowEchelonForm_modular_float (  
    const float p,  
    const size_t M,  
    const size_t N,  
    float * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.227.2.43 pColumnEchelonForm\_modular\_int32\_t()

```
size_t pColumnEchelonForm_modular_int32_t (  
    const int32_t p,  
    const size_t M,  
    const size_t N,  
    int32_t * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

#### 17.227.2.44 pRowEchelonForm\_modular\_int32\_t()

```
size_t pRowEchelonForm_modular_int32_t (  
    const int32_t p,  
    const size_t M,  
    const size_t N,  
    int32_t * A,
```

```

    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.45 pReducedColumnEchelonForm\_modular\_int32\_t()

```

size_t pReducedColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.46 pReducedRowEchelonForm\_modular\_int32\_t()

```

size_t pReducedRowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.47 Invertin\_modular\_double()

```

double * Invertin_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    int * nullity,
    bool positive )

```

#### 17.227.2.48 Invert\_modular\_double()

```

double * Invert_modular_double (
    const double p,
    const size_t M,
    const double * A,
    const size_t lda,
    double * X,
    const size_t ldx,

```

```
int * nullity,  
bool positive )
```

#### 17.227.2.49 Invert2\_modular\_double()

```
double * Invert2_modular_double (  
    const double p,  
    const size_t M,  
    double * A,  
    const size_t lda,  
    double * X,  
    const size_t ldx,  
    int * nullity,  
    bool positive )
```

#### 17.227.2.50 KrylovElim\_modular\_double()

```
size_t KrylovElim_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Q,  
    const size_t deg,  
    size_t * iterates,  
    size_t * inviterates,  
    const size_t maxit,  
    size_t virt,  
    bool positive )
```

#### 17.227.2.51 SpecRankProfile\_modular\_double()

```
size_t SpecRankProfile_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    const size_t deg,  
    size_t * rankProfile,  
    bool positive )
```

#### 17.227.2.52 Rank\_modular\_double()

```
size_t Rank_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    bool positive )
```

**17.227.2.53 IsSingular\_modular\_double()**

```
bool IsSingular_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.227.2.54 Det\_modular\_double()**

```
double Det_modular_double (
    const double p,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.227.2.55 Solve\_modular\_double()**

```
double * Solve_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    double * x,
    const int incx,
    const double * b,
    const int incb,
    bool positive )
```

**17.227.2.56 solveLB\_modular\_double()**

```
void solveLB_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    double * L,
    const size_t ldl,
    const size_t * Q,
    double * B,
    const size_t ldb,
    bool positive )
```

**17.227.2.57 solveLB2\_modular\_double()**

```
void solveLB2_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
```

```
double * L,
const size_t ldl,
const size_t * Q,
double * B,
const size_t ldb,
bool positive )
```

#### 17.227.2.58 RandomNullSpaceVector\_modular\_double()

```
void RandomNullSpaceVector_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double * X,
    const size_t incX,
    bool positive )
```

#### 17.227.2.59 NullSpaceBasis\_modular\_double()

```
size_t NullSpaceBasis_modular_double (
    const double p,
    const enum FFLAS::FFLAS_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** NS,
    size_t * ldn,
    size_t * NSdim,
    bool positive )
```

#### 17.227.2.60 RowRankProfile\_modular\_double()

```
size_t RowRankProfile_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

#### 17.227.2.61 ColumnRankProfile\_modular\_double()

```
size_t ColumnRankProfile_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
```

```

    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.62 RankProfileFromLU()

```

void RankProfileFromLU (
    const size_t * P,
    const size_t N,
    const size_t R,
    size_t * rkprofile,
    const enum FFPACK_C_LU_TAG LuTag )

```

#### 17.227.2.63 LeadingSubmatrixRankProfiles()

```

size_t LeadingSubmatrixRankProfiles (
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t LSm,
    const size_t LSn,
    const size_t * P,
    const size_t * Q,
    size_t * RRP,
    size_t * CRP )

```

#### 17.227.2.64 RowRankProfileSubmatrixIndices\_modular\_double()

```

size_t RowRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )

```

#### 17.227.2.65 ColRankProfileSubmatrixIndices\_modular\_double()

```

size_t ColRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )

```

#### 17.227.2.66 RowRankProfileSubmatrix\_modular\_double()

```

size_t RowRankProfileSubmatrix_modular_double (

```

```

    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )

```

#### 17.227.2.67 ColRankProfileSubmatrix\_modular\_double()

```

size_t ColRankProfileSubmatrix_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )

```

#### 17.227.2.68 getTriangular\_modular\_double()

```

void getTriangular_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    bool positive )

```

#### 17.227.2.69 getTriangularin\_modular\_double()

```

void getTriangularin_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    double * A,
    const size_t lda,
    bool positive )

```

#### 17.227.2.70 getEchelonForm\_modular\_double()

```

void getEchelonForm_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,

```

```

const enum FFLAS::FFLAS_DIAG Diag,
const size_t M,
const size_t N,
const size_t R,
const size_t * P,
const double * A,
const size_t lda,
double * T,
const size_t ldt,
const bool OnlyNonZeroVectors,
const enum FFPACK_C_LU_TAG LuTag,
bool positive )

```

#### 17.227.2.71 getEchelonFormin\_modular\_double()

```

void getEchelonFormin_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.72 getEchelonTransform\_modular\_double()

```

void getEchelonTransform_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const enum FFLAS::FFLAS_DIAG Diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.73 getReducedEchelonForm\_modular\_double()

```

void getReducedEchelonForm_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const double * A,

```



```

    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.74 getReducedEchelonFormin\_modular\_double()

```

void getReducedEchelonFormin_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.75 getReducedEchelonTransform\_modular\_double()

```

void getReducedEchelonTransform_modular_double (
    const double p,
    const enum FFLAS::FFLAS_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.227.2.76 PLUQtoEchelonPermutation()

```

void PLUQtoEchelonPermutation (
    const size_t N,
    const size_t R,
    const size_t * P,
    size_t * outPerm )

```

## 17.228 ffpack\_c.h File Reference

```

#include <stdbool.h>
#include <stdlib.h>
#include <inttypes.h>

```

## Macros

- `#define FFPACK_COMPILED`

## Enumerations

- enum `FFLAS_C_ORDER` { `FflasRowMajor` =101 , `FflasColMajor` =102 , `FflasRowMajor` =101 , `FflasColMajor` =102 }
- enum `FFLAS_C_TRANSPOSE` { `FflasNoTrans` = 111 , `FflasTrans` = 112 , `FflasNoTrans` = 111 , `FflasTrans` = 112 }
- enum `FFLAS_C_UPLO` { `FflasUpper` = 121 , `FflasLower` = 122 , `FflasUpper` = 121 , `FflasLower` = 122 }
- enum `FFLAS_C_DIAG` { `FflasNonUnit` = 131 , `FflasUnit` = 132 , `FflasNonUnit` = 131 , `FflasUnit` = 132 }
- enum `FFLAS_C_SIDE` { `FflasLeft` = 141 , `FflasRight` = 142 , `FflasLeft` = 141 , `FflasRight` = 142 }
- enum `FFPACK_C_LU_TAG` { `FfpackSlabRecursive` = 1 , `FfpackTileRecursive` = 2 , `FfpackSingular` = 3 }
- enum `FFPACK_C_CHARPOLY_TAG` {  
`FfpackLUK` =1 , `FfpackKG` =2 , `FfpackHybrid` =3 , `FfpackKGFast` =4 ,  
`FfpackDanilevski` =5 , `FfpackArithProg` =6 , `FfpackKGFastG` =7 }
- enum `FFPACK_C_MINPOLY_TAG` { `FfpackDense` =1 , `FfpackKGF` =2 }

## Functions

- void `LAPACKPerm2MathPerm` (size\_t \*MathP, const size\_t \*LapackP, const size\_t N)
- void `MathPerm2LAPACKPerm` (size\_t \*LapackP, const size\_t \*MathP, const size\_t N)
- void `MatrixApplyS_modular_double` (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void `PermApplyS_double` (double \*A, const size\_t lda, const size\_t width, const size\_t M2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void `MatrixApplyT_modular_double` (const double p, double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4, bool positive)
- void `PermApplyT_double` (double \*A, const size\_t lda, const size\_t width, const size\_t N2, const size\_t R1, const size\_t R2, const size\_t R3, const size\_t R4)
- void `composePermutationsLLM` (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void `composePermutationsLLL` (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)
- void `composePermutationsMLM` (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)
- void `cyclic_shift_mathPerm` (size\_t \*P, const size\_t s)
- void `cyclic_shift_row_modular_double` (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void `cyclic_shift_col_modular_double` (const double p, double \*A, size\_t m, size\_t n, size\_t lda, bool positive)
- void `applyP_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const enum `FFLAS_C_TRANSPOSE` Trans, const size\_t M, const size\_t ibeg, const size\_t iend, double \*A, const size\_t lda, const size\_t \*P, bool positive)
- void `fgetrsin_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*B, const size\_t ldb, int \*info, bool positive)
- double \* `fgetrs_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, double \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, double \*X, const size\_t ldx, const double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t `fgesvin_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*B, const size\_t ldb, int \*info, bool positive)
- size\_t `fgesv_modular_double` (const double p, const enum `FFLAS_C_SIDE` Side, const size\_t M, const size\_t N, const size\_t NRHS, double \*A, const size\_t lda, double \*X, const size\_t ldx, const double \*B, const size\_t ldb, int \*info)
- void `frtri_modular_double` (const double p, const enum `FFLAS_C_UPLO` Uplo, const enum `FFLAS_C_DIAG` Diag, const size\_t N, double \*A, const size\_t lda, bool positive)
- void `trinv_left_modular_double` (const double p, const size\_t N, const double \*L, const size\_t ldl, double \*X, const size\_t ldx, bool positive)

- void [ftrrm\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) diag, const size\_t N, double \*A, const size\_t lda, bool positive)
- size\_t [PLUQ\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, bool positive)
- size\_t [LUdivine\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const enum [FFLAS\\_C\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, const size\_t cutoff, bool positive)
- size\_t [LUdivine\\_small\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const enum [FFLAS\\_C\\_TRANSPOSE](#) trans, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [LUdivine\\_gauss\\_modular\\_double](#) (const double p, const enum [FFLAS\\_C\\_DIAG](#) Diag, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ColumnEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [RowEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ColumnEchelonForm\\_modular\\_float](#) (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [RowEchelonForm\\_modular\\_float](#) (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ColumnEchelonForm\\_modular\\_int32\\_t](#) (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [RowEchelonForm\\_modular\\_int32\\_t](#) (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ReducedColumnEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ReducedRowEchelonForm\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ReducedColumnEchelonForm\\_modular\\_float](#) (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ReducedRowEchelonForm\\_modular\\_float](#) (const float p, const size\_t M, const size\_t N, float \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ReducedColumnEchelonForm\\_modular\\_int32\\_t](#) (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ReducedRowEchelonForm\\_modular\\_int32\\_t](#) (const int32\_t p, const size\_t M, const size\_t N, int32\_t \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const enum [FFPACK\\_C\\_LU\\_TAG](#) LuTag, bool positive)
- size\_t [ReducedRowEchelonForm2\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, bool positive)
- size\_t [REF\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, const size\_t colbeg, const size\_t rowbeg, const size\_t colsize, size\_t \*Qt, size\_t \*P, bool positive)
- double \* [Invertin\\_modular\\_double](#) (const double p, const size\_t M, double \*A, const size\_t lda, int \*nullity, bool positive)
- double \* [Invert\\_modular\\_double](#) (const double p, const size\_t M, const double \*A, const size\_t lda, double \*X, const size\_t idx, int \*nullity, bool positive)
- double \* [Invert2\\_modular\\_double](#) (const double p, const size\_t M, double \*A, const size\_t lda, double \*X, const size\_t idx, int \*nullity, bool positive)
- size\_t [KrylovElim\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const size\_t deg, size\_t \*iterates, size\_t \*inviterates, const size\_t maxit, size\_t virt, bool positive)
- size\_t [SpecRankProfile\\_modular\\_double](#) (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, const size\_t deg, size\_t \*rankProfile, bool positive)

- `size_t Rank_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, bool positive)
- `bool IsSingular_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, bool positive)
- `double Det_modular_double` (const double p, const size\_t N, double \*A, const size\_t lda, bool positive)
- `double * Solve_modular_double` (const double p, const size\_t M, double \*A, const size\_t lda, double \*x, const int incx, const double \*b, const int incb, bool positive)
- `void solveLB_modular_double` (const double p, const enum FFLAS\_C\_SIDE Side, const size\_t M, const size\_t N, const size\_t R, double \*L, const size\_t ldl, const size\_t \*Q, double \*B, const size\_t ldb)
- `void solveLB2_modular_double` (const double p, const enum FFLAS\_C\_SIDE Side, const size\_t M, const size\_t N, const size\_t R, double \*L, const size\_t ldl, const size\_t \*Q, double \*B, const size\_t ldb, bool positive)
- `void RandomNullSpaceVector_modular_double` (const double p, const enum FFLAS\_C\_SIDE Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*X, const size\_t incX, bool positive)
- `size_t NullSpaceBasis_modular_double` (const double p, const enum FFLAS\_C\_SIDE Side, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*\*NS, size\_t \*ldn, size\_t \*NSdim, bool positive)
- `size_t RowRankProfile_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rkprofile, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- `size_t ColumnRankProfile_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rkprofile, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- `void RankProfileFromLU` (const size\_t \*P, const size\_t N, const size\_t R, size\_t \*rkprofile, const enum FFPACK\_C\_LU\_TAG LuTag)
- `size_t LeadingSubmatrixRankProfiles` (const size\_t M, const size\_t N, const size\_t R, const size\_t LSm, const size\_t LSn, const size\_t \*P, const size\_t \*Q, size\_t \*RRP, size\_t \*CRP)
- `size_t RowRankProfileSubmatrixIndices_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rowindices, size\_t \*\*colindices, size\_t \*R, bool positive)
- `size_t ColRankProfileSubmatrixIndices_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, size\_t \*\*rowindices, size\_t \*\*colindices, size\_t \*R, bool positive)
- `size_t RowRankProfileSubmatrix_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*\*X, size\_t \*R, bool positive)
- `size_t ColRankProfileSubmatrix_modular_double` (const double p, const size\_t M, const size\_t N, double \*A, const size\_t lda, double \*\*X, size\_t \*R, bool positive)
- `void getTriangular_modular_double` (const double p, const enum FFLAS\_C\_UPLO Uplo, const enum FFLAS\_C\_DIAG diag, const size\_t M, const size\_t N, const size\_t R, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, bool positive)
- `void getTriangularin_modular_double` (const double p, const enum FFLAS\_C\_UPLO Uplo, const enum FFLAS\_C\_DIAG diag, const size\_t M, const size\_t N, const size\_t R, double \*A, const size\_t lda, bool positive)
- `void getEchelonForm_modular_double` (const double p, const enum FFLAS\_C\_UPLO Uplo, const enum FFLAS\_C\_DIAG diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- `void getEchelonFormin_modular_double` (const double p, const enum FFLAS\_C\_UPLO Uplo, const enum FFLAS\_C\_DIAG diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- `void getEchelonTransform_modular_double` (const double p, const enum FFLAS\_C\_UPLO Uplo, const enum FFLAS\_C\_DIAG diag, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- `void getReducedEchelonForm_modular_double` (const double p, const enum FFLAS\_C\_UPLO Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const bool OnlyNonZeroVectors, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- `void getReducedEchelonFormin_modular_double` (const double p, const enum FFLAS\_C\_UPLO Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, double \*A, const size\_t lda, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- `void getReducedEchelonTransform_modular_double` (const double p, const enum FFLAS\_C\_UPLO Uplo, const size\_t M, const size\_t N, const size\_t R, const size\_t \*P, const size\_t \*Q, const double \*A, const size\_t lda, double \*T, const size\_t ldt, const enum FFPACK\_C\_LU\_TAG LuTag, bool positive)
- `void PLUQtoEchelonPermutation` (const size\_t N, const size\_t R, const size\_t \*P, size\_t \*outPerm)

## 17.228.1 Macro Definition Documentation

### 17.228.1.1 FFPACK\_COMPILED

```
#define FFPACK_COMPILED
```

## 17.228.2 Enumeration Type Documentation

### 17.228.2.1 FFLAS\_C\_ORDER

```
enum FFLAS_C_ORDER
```

Enumerator

|               |           |
|---------------|-----------|
| FflasRowMajor | row major |
| FflasColMajor | col major |
| FflasRowMajor |           |
| FflasColMajor |           |

### 17.228.2.2 FFLAS\_C\_TRANSPOSE

```
enum FFLAS_C_TRANSPOSE
```

Enumerator

|              |                           |
|--------------|---------------------------|
| FflasNoTrans | Matrix is not transposed. |
| FflasTrans   | Matrix is transposed.     |
| FflasNoTrans |                           |
| FflasTrans   |                           |

### 17.228.2.3 FFLAS\_C\_UPLO

```
enum FFLAS_C_UPLO
```

Enumerator

|            |                                                                        |
|------------|------------------------------------------------------------------------|
| FflasUpper | Triangular matrix is Upper triangular (if $i > j$ then $T_{i,j} = 0$ ) |
| FflasLower | Triangular matrix is Lower triangular (if $i < j$ then $T_{i,j} = 0$ ) |
| FflasUpper |                                                                        |
| FflasLower |                                                                        |

### 17.228.2.4 FFLAS\_C\_DIAG

```
enum FFLAS_C_DIAG
```

## Enumerator

---

### Enumerator

|              |                                                                   |
|--------------|-------------------------------------------------------------------|
| FflasNonUnit | Triangular matrix has an explicit arbitrary diagonal.             |
| FflasUnit    | Triangular matrix has an implicit unit diagonal ( $T_{i,i} = 1$ ) |
| FflasNonUnit |                                                                   |
| FflasUnit    |                                                                   |

### 17.228.2.5 FFLAS\_C\_SIDE

enum [FFLAS\\_C\\_SIDE](#)

#### Enumerator

|            |                                 |
|------------|---------------------------------|
| FflasLeft  | Operator applied on the left.   |
| FflasRight | Operator applied on the righth. |
| FflasLeft  |                                 |
| FflasRight |                                 |

### 17.228.2.6 FFPACK\_C\_LU\_TAG

enum [FFPACK\\_C\\_LU\\_TAG](#)

#### Enumerator

|                     |  |
|---------------------|--|
| FfpackSlabRecursive |  |
| FfpackTileRecursive |  |
| FfpackSingular      |  |

### 17.228.2.7 FFPACK\_C\_CHARPOLY\_TAG

enum [FFPACK\\_C\\_CHARPOLY\\_TAG](#)

#### Enumerator

|                  |  |
|------------------|--|
| FfpackLUK        |  |
| FfpackKG         |  |
| FfpackHybrid     |  |
| FfpackKGFast     |  |
| FfpackDanilevski |  |
| FfpackArithProg  |  |
| FfpackKGFastG    |  |

### 17.228.2.8 FFPACK\_C\_MINPOLY\_TAG

enum [FFPACK\\_C\\_MINPOLY\\_TAG](#)

## Enumerator

|            |  |
|------------|--|
| FpackDense |  |
| FpackKGF   |  |

## 17.228.3 Function Documentation

### 17.228.3.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
    size_t * MathP,
    const size_t * LapackP,
    const size_t N )
```

### 17.228.3.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
    size_t * LapackP,
    const size_t * MathP,
    const size_t N )
```

### 17.228.3.3 MatrixApplyS\_modular\_double()

```
void MatrixApplyS_modular_double (
    const double p,
    double * A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4,
    bool positive )
```

### 17.228.3.4 PermApplyS\_double()

```
void PermApplyS_double (
    double * A,
    const size_t lda,
    const size_t width,
    const size_t M2,
    const size_t R1,
    const size_t R2,
    const size_t R3,
    const size_t R4 )
```

### 17.228.3.5 MatrixApplyT\_modular\_double()

```
void MatrixApplyT_modular_double (
    const double p,
    double * A,
```

```
    const size_t lda,  
    const size_t width,  
    const size_t N2,  
    const size_t R1,  
    const size_t R2,  
    const size_t R3,  
    const size_t R4,  
    bool positive )
```

#### 17.228.3.6 PermApplyT\_double()

```
void PermApplyT_double (   
    double * A,  
    const size_t lda,  
    const size_t width,  
    const size_t N2,  
    const size_t R1,  
    const size_t R2,  
    const size_t R3,  
    const size_t R4 )
```

#### 17.228.3.7 composePermutationsLLM()

```
void composePermutationsLLM (   
    size_t * MathP,  
    const size_t * P1,  
    const size_t * P2,  
    const size_t R,  
    const size_t N )
```

#### 17.228.3.8 composePermutationsLLL()

```
void composePermutationsLLL (   
    size_t * P1,  
    const size_t * P2,  
    const size_t R,  
    const size_t N )
```

#### 17.228.3.9 composePermutationsMLM()

```
void composePermutationsMLM (   
    size_t * MathP1,  
    const size_t * P2,  
    const size_t R,  
    const size_t N )
```

#### 17.228.3.10 cyclic\_shift\_mathPerm()

```
void cyclic_shift_mathPerm (   
    size_t * P,  
    const size_t s )
```



**17.228.3.11 cyclic\_shift\_row\_modular\_double()**

```
void cyclic_shift_row_modular_double (
    const double p,
    double * A,
    size_t m,
    size_t n,
    size_t lda,
    bool positive )
```

**17.228.3.12 cyclic\_shift\_col\_modular\_double()**

```
void cyclic_shift_col_modular_double (
    const double p,
    double * A,
    size_t m,
    size_t n,
    size_t lda,
    bool positive )
```

**17.228.3.13 applyP\_modular\_double()**

```
void applyP_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const enum FFLAS_C_TRANSPOSE Trans,
    const size_t M,
    const size_t ibeg,
    const size_t iend,
    double * A,
    const size_t lda,
    const size_t * P,
    bool positive )
```

**17.228.3.14 fgetrsin\_modular\_double()**

```
void fgetrsin_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    double * A,
    const size_t lda,
    const size_t * P,
    const size_t * Q,
    double * B,
    const size_t ldb,
    int * info,
    bool positive )
```

**17.228.3.15 fgetrs\_modular\_double()**

```
double * fgetrs_modular_double (
    const double p,
```

```

const enum FFLAS_C_SIDE Side,
const size_t M,
const size_t N,
const size_t NRHS,
const size_t R,
double * A,
const size_t lda,
const size_t * P,
const size_t * Q,
double * X,
const size_t ldX,
const double * B,
const size_t ldb,
int * info,
bool positive )

```

#### 17.228.3.16 fgesvin\_modular\_double()

```

size_t fgesvin_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double * B,
    const size_t ldb,
    int * info,
    bool positive )

```

#### 17.228.3.17 fgesv\_modular\_double()

```

size_t fgesv_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t NRHS,
    double * A,
    const size_t lda,
    double * X,
    const size_t ldX,
    const double * B,
    const size_t ldb,
    int * info )

```

#### 17.228.3.18 ftrtri\_modular\_double()

```

void ftrtri_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG Diag,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )

```

**17.228.3.19 trinv\_left\_modular\_double()**

```
void trinv_left_modular_double (
    const double p,
    const size_t N,
    const double * L,
    const size_t ldl,
    double * X,
    const size_t ldx,
    bool positive )
```

**17.228.3.20 ftrtrm\_modular\_double()**

```
void ftrtrm_modular_double (
    const double p,
    const enum FFLAS_C_DIAG diag,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

**17.228.3.21 PLUQ\_modular\_double()**

```
size_t PLUQ_modular_double (
    const double p,
    const enum FFLAS_C_DIAG Diag,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    bool positive )
```

**17.228.3.22 LUdivine\_modular\_double()**

```
size_t LUdivine_modular_double (
    const double p,
    const enum FFLAS_C_DIAG Diag,
    const enum FFLAS_C_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const enum FFPACK_C_LU_TAG LuTag,
    const size_t cutoff,
    bool positive )
```

**17.228.3.23 LUdivine\_small\_modular\_double()**

```
size_t LUdivine_small_modular_double (
```

```

    const double p,
    const enum FFLAS_C_DIAG Diag,
    const enum FFLAS_C_TRANSPOSE trans,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.228.3.24 LUdivine\_gauss\_modular\_double()

```

size_t LUdivine_gauss_modular_double (
    const double p,
    const enum FFLAS_C_DIAG Diag,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.228.3.25 ColumnEchelonForm\_modular\_double()

```

size_t ColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.228.3.26 RowEchelonForm\_modular\_double()

```

size_t RowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.228.3.27 ColumnEchelonForm\_modular\_float()

```
size_t ColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

### 17.228.3.28 RowEchelonForm\_modular\_float()

```
size_t RowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

### 17.228.3.29 ColumnEchelonForm\_modular\_int32\_t()

```
size_t ColumnEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

### 17.228.3.30 RowEchelonForm\_modular\_int32\_t()

```
size_t RowEchelonForm_modular_int32_t (
    const int32_t p,
    const size_t M,
    const size_t N,
    int32_t * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.228.3.31 ReducedColumnEchelonForm\_modular\_double()**

```
size_t ReducedColumnEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.228.3.32 ReducedRowEchelonForm\_modular\_double()**

```
size_t ReducedRowEchelonForm_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.228.3.33 ReducedColumnEchelonForm\_modular\_float()**

```
size_t ReducedColumnEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.228.3.34 ReducedRowEchelonForm\_modular\_float()**

```
size_t ReducedRowEchelonForm_modular_float (
    const float p,
    const size_t M,
    const size_t N,
    float * A,
    const size_t lda,
    size_t * P,
    size_t * Qt,
    const bool transform,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.228.3.35 ReducedColumnEchelonForm\_modular\_int32\_t()**

```
size_t ReducedColumnEchelonForm_modular_int32_t (  
    const int32_t p,  
    const size_t M,  
    const size_t N,  
    int32_t * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

**17.228.3.36 ReducedRowEchelonForm\_modular\_int32\_t()**

```
size_t ReducedRowEchelonForm_modular_int32_t (  
    const int32_t p,  
    const size_t M,  
    const size_t N,  
    int32_t * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    const enum FFPACK_C_LU_TAG LuTag,  
    bool positive )
```

**17.228.3.37 ReducedRowEchelonForm2\_modular\_double()**

```
size_t ReducedRowEchelonForm2_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    size_t * P,  
    size_t * Qt,  
    const bool transform,  
    bool positive )
```

**17.228.3.38 REF\_modular\_double()**

```
size_t REF_modular_double (  
    const double p,  
    const size_t M,  
    const size_t N,  
    double * A,  
    const size_t lda,  
    const size_t colbeg,  
    const size_t rowbeg,  
    const size_t colsize,  
    size_t * Qt,  
    size_t * P,  
    bool positive )
```

**17.228.3.39 Invertin\_modular\_double()**

```
double * Invertin_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    int * nullity,
    bool positive )
```

**17.228.3.40 Invert\_modular\_double()**

```
double * Invert_modular_double (
    const double p,
    const size_t M,
    const double * A,
    const size_t lda,
    double * X,
    const size_t ldx,
    int * nullity,
    bool positive )
```

**17.228.3.41 Invert2\_modular\_double()**

```
double * Invert2_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    double * X,
    const size_t ldx,
    int * nullity,
    bool positive )
```

**17.228.3.42 KrylovElim\_modular\_double()**

```
size_t KrylovElim_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t * P,
    size_t * Q,
    const size_t deg,
    size_t * iterates,
    size_t * inviterates,
    const size_t maxit,
    size_t virt,
    bool positive )
```

**17.228.3.43 SpecRankProfile\_modular\_double()**

```
size_t SpecRankProfile_modular_double (
    const double p,
```



```
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    const size_t deg,
    size_t * rankProfile,
    bool positive )
```

#### 17.228.3.44 Rank\_modular\_double()

```
size_t Rank_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

#### 17.228.3.45 IsSingular\_modular\_double()

```
bool IsSingular_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

#### 17.228.3.46 Det\_modular\_double()

```
double Det_modular_double (
    const double p,
    const size_t N,
    double * A,
    const size_t lda,
    bool positive )
```

#### 17.228.3.47 Solve\_modular\_double()

```
double * Solve_modular_double (
    const double p,
    const size_t M,
    double * A,
    const size_t lda,
    double * x,
    const int incx,
    const double * b,
    const int incb,
    bool positive )
```

#### 17.228.3.48 solveLB\_modular\_double()

```
void solveLB_modular_double (
    const double p,
```

```

    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    double * L,
    const size_t ldl,
    const size_t * Q,
    double * B,
    const size_t ldb )

```

#### 17.228.3.49 solveLB2\_modular\_double()

```

void solveLB2_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    const size_t R,
    double * L,
    const size_t ldl,
    const size_t * Q,
    double * B,
    const size_t ldb,
    bool positive )

```

#### 17.228.3.50 RandomNullSpaceVector\_modular\_double()

```

void RandomNullSpaceVector_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double * X,
    const size_t incX,
    bool positive )

```

#### 17.228.3.51 NullSpaceBasis\_modular\_double()

```

size_t NullSpaceBasis_modular_double (
    const double p,
    const enum FFLAS_C_SIDE Side,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** NS,
    size_t * ldn,
    size_t * NSdim,
    bool positive )

```

#### 17.228.3.52 RowRankProfile\_modular\_double()

```

size_t RowRankProfile_modular_double (

```

```

    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.228.3.53 ColumnRankProfile\_modular\_double()

```

size_t ColumnRankProfile_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rkprofile,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

### 17.228.3.54 RankProfileFromLU()

```

void RankProfileFromLU (
    const size_t * P,
    const size_t N,
    const size_t R,
    size_t * rkprofile,
    const enum FFPACK_C_LU_TAG LuTag )

```

### 17.228.3.55 LeadingSubmatrixRankProfiles()

```

size_t LeadingSubmatrixRankProfiles (
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t LSm,
    const size_t LSn,
    const size_t * P,
    const size_t * Q,
    size_t * RRP,
    size_t * CRP )

```

### 17.228.3.56 RowRankProfileSubmatrixIndices\_modular\_double()

```

size_t RowRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )

```

**17.228.3.57 ColRankProfileSubmatrixIndices\_modular\_double()**

```
size_t ColRankProfileSubmatrixIndices_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    size_t ** rowindices,
    size_t ** colindices,
    size_t * R,
    bool positive )
```

**17.228.3.58 RowRankProfileSubmatrix\_modular\_double()**

```
size_t RowRankProfileSubmatrix_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )
```

**17.228.3.59 ColRankProfileSubmatrix\_modular\_double()**

```
size_t ColRankProfileSubmatrix_modular_double (
    const double p,
    const size_t M,
    const size_t N,
    double * A,
    const size_t lda,
    double ** X,
    size_t * R,
    bool positive )
```

**17.228.3.60 getTriangular\_modular\_double()**

```
void getTriangular_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    bool positive )
```

**17.228.3.61 getTriangularin\_modular\_double()**

```
void getTriangularin_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    double * A,
    const size_t lda,
    bool positive )
```

**17.228.3.62 getEchelonForm\_modular\_double()**

```
void getEchelonForm_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.228.3.63 getEchelonFormin\_modular\_double()**

```
void getEchelonFormin_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )
```

**17.228.3.64 getEchelonTransform\_modular\_double()**

```
void getEchelonTransform_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const enum FFLAS_C_DIAG diag,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
```

```

    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.228.3.65 getReducedEchelonForm\_modular\_double()

```

void getReducedEchelonForm_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const bool OnlyNonZeroVectors,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.228.3.66 getReducedEchelonFormin\_modular\_double()

```

void getReducedEchelonFormin_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    double * A,
    const size_t lda,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

#### 17.228.3.67 getReducedEchelonTransform\_modular\_double()

```

void getReducedEchelonTransform_modular_double (
    const double p,
    const enum FFLAS_C_UPLO Uplo,
    const size_t M,
    const size_t N,
    const size_t R,
    const size_t * P,
    const size_t * Q,
    const double * A,
    const size_t lda,
    double * T,
    const size_t ldt,
    const enum FFPACK_C_LU_TAG LuTag,
    bool positive )

```

**17.228.3.68 PLUQtoEchelonPermutation()**

```
void PLUQtoEchelonPermutation (
    const size_t N,
    const size_t R,
    const size_t * P,
    size_t * outPerm )
```

**17.229 ffpack\_inst.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "givaro/modular.h"
#include "givaro/modular-balanced.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "ffpack_inst_implem.inl"
```

**Macros**

- `#define __FFPACK_INST_C`
- `#define FFLAS_COMPILED`
- `#define INST_OR_DECL`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

**17.229.1 Macro Definition Documentation****17.229.1.1 \_\_FFPACK\_INST\_C**

```
#define __FFPACK_INST_C
```

**17.229.1.2 FFLAS\_COMPILED**

```
#define FFLAS_COMPILED
```

**17.229.1.3 INST\_OR\_DECL**

```
#define INST_OR_DECL
```

**17.229.1.4 FFLAS\_FIELD [1/2]**

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

**17.229.1.5 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**17.229.1.6 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**17.229.1.7 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**17.229.1.8 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**17.229.1.9 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**17.229.1.10 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**17.229.1.11 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

## 17.230 ffpack\_inst.h File Reference

```
#include "givaro/modular.h"  
#include "givaro/modular-balanced.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include "ffpack_inst_implem.inl"
```

### Macros

- `#define FFLAS_COMPILED`
- `#define INST_OR_DECL <>`
- `#define FFLAS_FIELD Givaro::ModularBalanced`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`
- `#define FFLAS_FIELD Givaro::Modular`
- `#define FFLAS_ELT double`
- `#define FFLAS_ELT float`
- `#define FFLAS_ELT int64_t`

### 17.230.1 Macro Definition Documentation

**17.230.1.1 FFLAS\_COMPILED**

```
#define FFLAS_COMPILED
```



**17.230.1.2 INST\_OR\_DECL**

```
#define INST_OR_DECL <>
```

**17.230.1.3 FFLAS\_FIELD [1/2]**

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

**17.230.1.4 FFLAS\_ELT [1/6]**

```
#define FFLAS_ELT double
```

**17.230.1.5 FFLAS\_ELT [2/6]**

```
#define FFLAS_ELT float
```

**17.230.1.6 FFLAS\_ELT [3/6]**

```
#define FFLAS_ELT int64_t
```

**17.230.1.7 FFLAS\_FIELD [2/2]**

```
#define FFLAS_FIELD Givaro::Modular
```

**17.230.1.8 FFLAS\_ELT [4/6]**

```
#define FFLAS_ELT double
```

**17.230.1.9 FFLAS\_ELT [5/6]**

```
#define FFLAS_ELT float
```

**17.230.1.10 FFLAS\_ELT [6/6]**

```
#define FFLAS_ELT int64_t
```

**17.231 ffpack\_inst\_implem.inl File Reference****Namespaces**

- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

**Functions**

- void [composePermutationsLLM](#) (size\_t \*MathP, const size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)  
*Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $\text{MathP}$  as a  $\text{MathPermutation}$  format.*
- void [composePermutationsLLL](#) (size\_t \*P1, const size\_t \*P2, const size\_t R, const size\_t N)

Computes  $P1 \times \text{Diag}(I_R, P2)$  where  $P1$  is a LAPACK and  $P2$  a LAPACK permutation and store the result in  $P1$  as a LAPACK permutation.

- void `composePermutationsMLM` (size\_t \*MathP1, const size\_t \*P2, const size\_t R, const size\_t N)

Computes  $\text{MathP1} \times \text{Diag}(I_R, P2)$  where  $\text{MathP1}$  is a MathPermutation and  $P2$  a LAPACK permutation and store the result in  $\text{MathP1}$  as a MathPermutation format.

- void `cyclic_shift_mathPerm` (size\_t \*P, const size\_t s)
- template<typename Base\_t >  
void `cyclic_shift_row_col` (Base\_t \*A, size\_t m, size\_t n, size\_t lda)
- template `INST_OR_DECL` void `cyclic_shift_row` (const FFLAS\_FIELD< FFLAS\_ELT > &F, FFLAS\_ELT \*A, size\_t m, size\_t n, size\_t lda)
- template `INST_OR_DECL` void `cyclic_shift_col` (const FFLAS\_FIELD< FFLAS\_ELT > &F, FFLAS\_ELT \*A, size\_t m, size\_t n, size\_t lda)
- template `INST_OR_DECL` void `applyP` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const FFLAS::FFLAS\_TRANSPOSE Trans, const size\_t M, const size\_t ibeg, const size\_t iend, FFLAS\_ELT \*A, const size\_t lda, const size\_t \*P)
- template `INST_OR_DECL` void `fgetrs` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t R, FFLAS\_ELT \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, FFLAS\_ELT \*B, const size\_t ldb, int \*info)
- template `INST_OR_DECL` FFLAS\_ELT \* `fgetrs` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t NRHS, const size\_t R, FFLAS\_ELT \*A, const size\_t lda, const size\_t \*P, const size\_t \*Q, FFLAS\_ELT \*X, const size\_t ldx, const FFLAS\_ELT \*B, const size\_t ldb, int \*info)
- template `INST_OR_DECL` size\_t `fgesv` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, FFLAS\_ELT \*B, const size\_t ldb, int \*info)
- template `INST_OR_DECL` size\_t `fgesv` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE Side, const size\_t M, const size\_t N, const size\_t NRHS, FFLAS\_ELT \*A, const size\_t lda, FFLAS\_ELT \*X, const size\_t ldx, const FFLAS\_ELT \*B, const size\_t ldb, int \*info)
- template `INST_OR_DECL` void `fttrtri` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_UPLO Uplo, const FFLAS::FFLAS\_DIAG Diag, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, const size\_t threshold)
- template `INST_OR_DECL` void `trinv_left` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t N, const FFLAS\_ELT \*L, const size\_t ldl, FFLAS\_ELT \*X, const size\_t ldx)
- template `INST_OR_DECL` void `fttrrm` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_SIDE side, const FFLAS::FFLAS\_DIAG diag, const size\_t N, FFLAS\_ELT \*A, const size\_t lda)
- template `INST_OR_DECL` size\_t `PLUQ` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, size\_t \*P, size\_t \*Q)
- template `INST_OR_DECL` size\_t `LUdivine` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_DIAG Diag, const FFLAS::FFLAS\_TRANSPOSE trans, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const FFPACK\_LU\_TAG LuTag, const size\_t cutoff)
- template `INST_OR_DECL` size\_t `LUdivine_small` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_DIAG Diag, const FFLAS::FFLAS\_TRANSPOSE trans, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const FFPACK\_LU\_TAG LuTag)
- template `INST_OR_DECL` size\_t `LUdivine_gauss` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const FFLAS::FFLAS\_DIAG Diag, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, size\_t \*P, size\_t \*Q, const FFPACK\_LU\_TAG LuTag)
- template `INST_OR_DECL` size\_t `RowEchelonForm` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag)
- template `INST_OR_DECL` size\_t `ReducedRowEchelonForm` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag)
- template `INST_OR_DECL` size\_t `ColumnEchelonForm` (const FFLAS\_FIELD< FFLAS\_ELT > &F, const size\_t M, const size\_t N, FFLAS\_ELT \*A, const size\_t lda, size\_t \*P, size\_t \*Qt, const bool transform, const FFPACK\_LU\_TAG LuTag)

- template `INST_OR_DECL` `size_t ReducedColumnEchelonForm` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Qt, const bool transform, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, `FFLAS_ELT` \*A, const `size_t` lda, int &nullity)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, const `size_t` ldx, int &nullity)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Invert2` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, const `size_t` ldx, int &nullity)
- template `INST_OR_DECL` `std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > & CharPoly` (const `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R`, `std::list< Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element > &charp`, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_FIELD< FFLAS_ELT >::RandIter &G`, const `FFPACK_CHARPOLY_TAG` CharpTag, const `size_t` degree)
- template `INST_OR_DECL` `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly` (const `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R`, `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element &charp`, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_FIELD< FFLAS_ELT >::RandIter &G`, const `FFPACK_CHARPOLY_TAG` CharpTag, const `size_t` degree)
- template `INST_OR_DECL` `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element & CharPoly` (const `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > > &R`, `Givaro::Poly1Dom< FFLAS_FIELD< FFLAS_ELT > >::Element &charp`, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_CHARPOLY_TAG` CharpTag, const `size_t` degree)
- template `INST_OR_DECL` `std::vector< FFLAS_ELT > & MinPoly` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `std::vector< FFLAS_ELT > &minP`, const `size_t` N, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_FIELD< FFLAS_ELT >::RandIter &G`)
- template `INST_OR_DECL` `std::vector< FFLAS_ELT > & MinPoly` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `std::vector< FFLAS_ELT > &minP`, const `size_t` N, const `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` `std::vector< FFLAS_ELT > & MatVecMinPoly` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `std::vector< FFLAS_ELT > &minP`, const `size_t` N, const `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS_ELT` \*V, const `size_t` incv)
- template `INST_OR_DECL` `size_t KrylovElim` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q, const `size_t` deg, `size_t` \*iterates, `size_t` \*inviterates, const `size_t` maxit, `size_t` virt)
- template `INST_OR_DECL` `size_t SpecRankProfile` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, const `size_t` deg, `size_t` \*rankProfile)
- template `INST_OR_DECL` `size_t Rank` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` bool `IsSingular` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` `FFLAS_ELT` & `Det` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `FFLAS_ELT` &det, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*P, `size_t` \*Q)
- template `INST_OR_DECL` `FFLAS_ELT` & `Det` (const `FFLAS_FIELD< FFLAS_ELT >` &F, `FFLAS_ELT` &det, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, const `FFLAS::ParSeqHelper::Parallel< FFLAS::CuttingStrategy::Recursive, FFLAS::StrategyParameter::Threads >` &parH, `size_t` \*P, `size_t` \*Q)
- template `INST_OR_DECL` `FFLAS_ELT` \* `Solve` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*x, const int incx, const `FFLAS_ELT` \*b, const int incb)
- template `INST_OR_DECL` void `solveLB` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*L, const `size_t` ldl, const `size_t` \*Q, `FFLAS_ELT` \*B, const `size_t` ldb)
- template `INST_OR_DECL` void `solveLB2` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*L, const `size_t` ldl, const `size_t` \*Q, `FFLAS_ELT` \*B, const `size_t` ldb)
- template `INST_OR_DECL` void `RandomNullSpaceVector` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, const `size_t` incX)
- template `INST_OR_DECL` `size_t NullSpaceBasis` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_SIDE` Side, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*NS, `size_t` &ldn, `size_t` &NSdim)

- template `INST_OR_DECL` `size_t` `RowRankProfile` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` `size_t` `ColumnRankProfile` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag)
- void `RankProfileFromLU` (const `size_t` \*P, const `size_t` N, const `size_t` R, `size_t` \*&rkprofile, const `FFPACK_LU_TAG` LuTag)  
*Recovers the column/row rank profile from the permutation of an LU decomposition.*
- `size_t` `LeadingSubmatrixRankProfiles` (const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` LSm, const `size_t` LSn, const `size_t` \*P, const `size_t` \*Q, `size_t` \*RRP, `size_t` \*CRP)  
*Recovers the row and column rank profiles of any leading submatrix from the PLUQ decomposition.*
- template `INST_OR_DECL` `size_t` `RowRankProfileSubmatrixIndices` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rowindices, `size_t` \*&colindices, `size_t` &R)
- template `INST_OR_DECL` `size_t` `ColRankProfileSubmatrixIndices` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `size_t` \*&rowindices, `size_t` \*&colindices, `size_t` &R)
- template `INST_OR_DECL` `size_t` `RowRankProfileSubmatrix` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, `size_t` &R)
- template `INST_OR_DECL` `size_t` `ColRankProfileSubmatrix` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` M, const `size_t` N, `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*X, `size_t` &R)
- template `INST_OR_DECL` void `getTriangular< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const bool OnlyNonZeroVectors)
- template `INST_OR_DECL` void `getTriangular< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, `FFLAS_ELT` \*A, const `size_t` lda)
- template `INST_OR_DECL` void `getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const bool OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `FFLAS::FFLAS_DIAG` diag, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const bool OnlyNonZeroVectors, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonForm< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, `FFLAS_ELT` \*A, const `size_t` lda, const `FFPACK_LU_TAG` LuTag)
- template `INST_OR_DECL` void `getReducedEchelonTransform< FFLAS_FIELD< FFLAS_ELT > >` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `FFLAS::FFLAS_UPLO` Uplo, const `size_t` M, const `size_t` N, const `size_t` R, const `size_t` \*P, const `size_t` \*Q, const `FFLAS_ELT` \*A, const `size_t` lda, `FFLAS_ELT` \*T, const `size_t` ldt, const `FFPACK_LU_TAG` LuTag)
- void `PLUQtoEchelonPermutation` (const `size_t` N, const `size_t` R, const `size_t` \*P, `size_t` \*outPerm)  
*Auxiliary routine: determines the permutation that changes a PLUQ decomposition into a echelon form revealing PLUQ decomposition.*
- template `INST_OR_DECL` `FFLAS_ELT` \* `LQUPtoInverseOfFullRankMinor` (const `FFLAS_FIELD< FFLAS_ELT >` &F, const `size_t` rank, `FFLAS_ELT` \*A\_factors, const `size_t` lda, const `size_t` \*QtPointer, `FFLAS_ELT` \*X, const `size_t` ldx)

## 17.232 blockcuts.inl File Reference

```
#include <fflas-ffpack/fflas/fflas_enum.h>
#include <math.h>
#include <cassert>
```

### Data Structures

- struct [Single](#)
- struct [Row](#)
- struct [Column](#)
- struct [Block](#)
- struct [Recursive](#)
- struct [Fixed](#)
- struct [Threads](#)
- struct [Grain](#)
- struct [TwoD](#)
- struct [TwoDAdaptive](#)
- struct [ThreeD](#)
- struct [ThreeDInPlace](#)
- struct [ThreeDAdaptive](#)
- struct [Parallel](#)< C, P >
- struct [Sequential](#)
- struct [Compose](#)< H1, H2 >
- struct [ForStrategy1D](#)< blocksize\_t, Cut, Param >
- struct [ForStrategy2D](#)< blocksize\_t, Cut, Param >

### Namespaces

- namespace [FFLAS](#)
  - namespace [FFLAS::CuttingStrategy](#)
  - namespace [FFLAS::StrategyParameter](#)
  - namespace [FFLAS::ParSeqHelper](#)
- [ParSeqHelper](#) for both *fgemm* and *ftrsm*.*

### Macros

- #define [\\_\\_FFLASFFPACK\\_fflas\\_blockcuts\\_INL](#)
- #define [\\_\\_FFLASFFPACK\\_MINBLOCKCUTS](#) ((size\_t)256)

### Typedefs

- typedef Row [RNSModulus](#)

### Functions

- template<class Cut = CuttingStrategy::Block, class Strat = StrategyParameter::Threads>  
void [BlockCuts](#) (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t numthreads)
- template<> void [BlockCuts](#)< [CuttingStrategy::Single](#), [StrategyParameter::Threads](#) > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t numthreads)
- template<> void [BlockCuts](#)< [CuttingStrategy::Row](#), [StrategyParameter::Fixed](#) > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t numthreads)
- template<> void [BlockCuts](#)< [CuttingStrategy::Row](#), [StrategyParameter::Grain](#) > (size\_t &RBLOCKSIZE, size\_t &CBLOCKSIZE, const size\_t m, const size\_t n, const size\_t grainsize)

- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t grainsize)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<> void BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads > (size_t &RBLOCKSIZE, size_t &CBLOCKSIZE, const size_t m, const size_t n, const size_t numthreads)`
- `template<class Cut = CuttingStrategy::Block, class Param = StrategyParameter::Threads>  
void BlockCuts (size_t &rowBlockSize, size_t &colBlockSize, size_t &lastRBS, size_t &lastCBS, size_t &changeRBS, size_t &changeCBS, size_t &numRowBlock, size_t &numColBlock, size_t m, size_t n, const size_t numthreads)`

## 17.232.1 Macro Definition Documentation

### 17.232.1.1 \_\_FFLASFFPACK\_fflas\_blockcuts\_INL

```
#define __FFLASFFPACK_fflas_blockcuts_INL
```

### 17.232.1.2 \_\_FFLASFFPACK\_MINBLOCKCUTS

```
#define __FFLASFFPACK_MINBLOCKCUTS ((size_t)256)
```

## 17.233 fflas\_plevel1.h File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

## Namespaces

- namespace [FFLAS](#)

## Functions

- `template<class Field >  
void pfzero (const Field &F, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)`
- `template<class Field, class RandIter >  
void pfrand (const Field &F, RandIter &G, size_t m, size_t n, typename Field::Element_ptr C, size_t BS=0)`
- `template<class Field, class Cut, class Param >  
Field::Element & fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr x, const size_t incx, typename Field::ConstElement_ptr y, const size_t incy, typename Field::Element &d, const ParSeqHelper::Parallel< Cut, Param > par)`
- `template<typename Field, class Cut, class Param >  
Field::Element fdot (const Field &F, const size_t N, typename Field::ConstElement_ptr X, const size_t incX, typename Field::ConstElement_ptr Y, const size_t incY, const ParSeqHelper::Parallel< Cut, Param > par)`

## 17.234 kaapi\_routines.inl File Reference

### Macros

- [#define \\_\\_FFLASFFPACK\\_KAAPI\\_ROUTINES\\_INL](#)

### 17.234.1 Macro Definition Documentation

#### 17.234.1.1 \_\_FFLASFFPACK\_KAAPI\_ROUTINES\_INL

```
#define __FFLASFFPACK_KAAPI_ROUTINES_INL
```

## 17.235 parallel.h File Reference

```
#include "fflas-ffpack/config.h"
#include "fflas-ffpack/paladin/blockcuts.inl"
```

### Macros

- [#define \\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- [#define index\\_t](#) size\_t
- [#define TASK\(M, l\)](#) {l;}
- [#define WAIT](#)
- [#define CHECK\\_DEPENDENCIES](#)
- [#define BARRIER](#)
- [#define PAR\\_BLOCK](#)
- [#define SYNCH\\_GROUP](#)(Args...) {{Args};}
- [#define NUM\\_THREADS](#) 1
- [#define MAX\\_THREADS](#) 1
- [#define READ](#)(Args...)
- [#define WRITE](#)(Args...)
- [#define READWRITE](#)(Args...)
- [#define CONSTREFERENCE](#)(...)
- [#define VALUE](#)(...)
- [#define BEGIN\\_PARALLEL\\_MAIN](#)(Args...) int main(Args) {
- [#define END\\_PARALLEL\\_MAIN](#)(void) return 0; }
- [#define FORBLOCK1D](#)(iter, m, Helper, Args...)
- [#define FOR1D](#)(i, m, Helper, Args...)
- [#define PARFORBLOCK1D](#)(iter, m, Helper, Args...)
- [#define PARFOR1D](#)(iter, m, Helper, Args...)
- [#define FORBLOCK2D](#)(iter, m, n, Helper, Args...)
- [#define FOR2D](#)(i, j, m, n, Helper, Args...)
- [#define PARFORBLOCK2D](#)(iter, m, n, Helper, Args...) [FORBLOCK2D](#)(iter, m, n, Helper, Args)
- [#define PARFOR2D](#)(i, j, m, n, Helper, Args...) [FOR2D](#)(i, j, m, n, Helper, Args)
- [#define COMMA](#) ,
- [#define MODE](#)(...) \_\_VA\_ARGS\_\_
- [#define RETURNPARAM](#)(f, P1, Args...) P1=f(Args)
- [#define NUMARGS](#)(...) [PP\\_NARG](#)(\_\_VA\_ARGS\_\_,[PP\\_RSEQ\\_N](#)())
- [#define PP\\_NARG](#)(...) [PP\\_ARG\\_N](#)(\_\_VA\_ARGS\_\_)
- [#define PP\\_ARG\\_N](#)(\_1, \_2, \_3, \_4, \_5, \_6, \_7, \_8, \_9, \_10, \_11, \_12, \_13, \_14, \_15, \_16, \_17, \_18, \_19, \_20, \_21, \_22, \_23, \_24, \_25, \_26, \_27, \_28, \_29, \_30, \_31, \_32, \_33, \_34, \_35, \_36, \_37, \_38, \_39, \_40, \_41, \_42, \_43, \_44, \_45, \_46, \_47, \_48, \_49, \_50, \_51, \_52, \_53, \_54, \_55, \_56, \_57, \_58, \_59, \_60, \_61, \_62, \_63, N, ...) N

- `#define PP_RSEQ_N()`
- `#define NOSPLIT() FFLAS::ParSeqHelper::Sequential()`
- `#define splitting_0() FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::Threads>()`
- `#define splitting_1(a) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::Threads>(a)`
- `#define splitting_2(a, c) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,c>(a)`
- `#define splitting_3(a, b, c) FFLAS::ParSeqHelper::Parallel<b,c>(a)`
- `#define splitt(_1, _2, _3, NAME, ...) NAME`
- `#define SPLITTER(...) splitt(__VA_ARGS__, splitting_3, splitting_2, splitting_1, splitting_0)(__VA_ARGS__)`

## 17.235.1 Macro Definition Documentation

### 17.235.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.235.1.2 index\_t

```
#define index_t size_t
```

### 17.235.1.3 TASK

```
#define TASK(  
    M,  
    I ) {I;}
```

### 17.235.1.4 WAIT

```
#define WAIT
```

### 17.235.1.5 CHECK\_DEPENDENCIES

```
#define CHECK_DEPENDENCIES
```

### 17.235.1.6 BARRIER

```
#define BARRIER
```

### 17.235.1.7 PAR\_BLOCK

```
#define PAR_BLOCK
```

### 17.235.1.8 SYNCH\_GROUP

```
#define SYNCH_GROUP(  
    Args... ) {{Args}};
```

### 17.235.1.9 NUM\_THREADS

```
#define NUM_THREADS 1
```



**17.235.1.10 MAX\_THREADS**

```
#define MAX_THREADS 1
```

**17.235.1.11 READ**

```
#define READ(  
    Args... )
```

**17.235.1.12 WRITE**

```
#define WRITE(  
    Args... )
```

**17.235.1.13 READWRITE**

```
#define READWRITE(  
    Args... )
```

**17.235.1.14 CONSTREFERENCE**

```
#define CONSTREFERENCE(  
    ... )
```

**17.235.1.15 VALUE**

```
#define VALUE(  
    ... )
```

**17.235.1.16 BEGIN\_PARALLEL\_MAIN**

```
#define BEGIN_PARALLEL_MAIN(  
    Args... ) int main(Args) {
```

**17.235.1.17 END\_PARALLEL\_MAIN**

```
#define END_PARALLEL_MAIN(  
    void ) return 0; }
```

**17.235.1.18 FORBLOCK1D**

```
#define FORBLOCK1D(  
    iter,  
    m,  
    Helper,  
    Args... )
```

**Value:**

```
{ FFLAS::ForStrategy1D<std::remove_const<decltype(m)>::type, typename decltype(Helper)::Cut, typename  
    decltype(Helper)::Param> iter(m, Helper); \  
    for(iter.initialize(); !iter.isTerminated(); ++iter) \  
    {Args;} }
```

**17.235.1.19 FOR1D**

```
#define FOR1D(
    i,
    m,
    Helper,
    Args... )
```

**Value:**

```
FORBLOCK1D(_internal_iterator, m, Helper,
    for(auto i=_internal_iterator.begin(); i!=_internal_iterator.end(); ++i) \
    { Args; })
```

**17.235.1.20 PARFORBLOCK1D**

```
#define PARFORBLOCK1D(
    iter,
    m,
    Helper,
    Args... )
```

**Value:**

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter) \
{ Args; }
```

**17.235.1.21 PARFOR1D**

```
#define PARFOR1D(
    iter,
    m,
    Helper,
    Args... )
```

**Value:**

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter) \
{ Args; }
```

**17.235.1.22 FORBLOCK2D**

```
#define FORBLOCK2D(
    iter,
    m,
    n,
    Helper,
    Args... )
```

**Value:**

```
{ FFLAS::ForStrategy2D<std::remove_const<decltype(m)>::type, typename decltype(Helper)::Cut, typename
    decltype(Helper)::Param> iter(m,n,Helper); \
    for(iter.initialize(); !iter.isTerminated(); ++iter) \
    { Args; } }
```

**17.235.1.23 FOR2D**

```
#define FOR2D(
    i,
    j,
    m,
    n,
    Helper,
    Args... )
```

**Value:**

```
FORBLOCK2D(_internal_iterator, m, n, Helper,
    for(auto i=_internal_iterator.ibegin(); i!=_internal_iterator.iend(); ++i) \
```

```
for(auto j=_internal_iterator.jbegin(); j!=_internal_iterator.jend(); ++j) \
{ Args; }
```

#### 17.235.1.24 PARFORBLOCK2D

```
#define PARFORBLOCK2D(
    iter,
    m,
    n,
    Helper,
    Args... ) FORBLOCK2D(iter, m, n, Helper, Args)
```

#### 17.235.1.25 PARFOR2D

```
#define PARFOR2D(
    i,
    j,
    m,
    n,
    Helper,
    Args... ) FOR2D(i, j, m, n, Helper, Args)
```

#### 17.235.1.26 COMMA

```
#define COMMA ,
```

#### 17.235.1.27 MODE

```
#define MODE(
    ... ) __VA_ARGS__
```

#### 17.235.1.28 RETURNPARAM

```
#define RETURNPARAM(
    f,
    Pl,
    Args... ) Pl=f(Args)
```

#### 17.235.1.29 NUMARGS

```
#define NUMARGS(
    ... ) PP_NARG__(__VA_ARGS__, PP_RSEQ_N())
```

#### 17.235.1.30 PP\_NARG\_

```
#define PP_NARG_(
    ... ) PP_ARG_N(__VA_ARGS__)
```

#### 17.235.1.31 PP\_ARG\_N

```
#define PP_ARG_N(
    _1,
```

\_2,  
\_3,  
\_4,  
\_5,  
\_6,  
\_7,  
\_8,  
\_9,  
\_10,  
\_11,  
\_12,  
\_13,  
\_14,  
\_15,  
\_16,  
\_17,  
\_18,  
\_19,  
\_20,  
\_21,  
\_22,  
\_23,  
\_24,  
\_25,  
\_26,  
\_27,  
\_28,  
\_29,  
\_30,  
\_31,  
\_32,  
\_33,  
\_34,  
\_35,  
\_36,  
\_37,  
\_38,  
\_39,  
\_40,  
\_41,  
\_42,  
\_43,  
\_44,  
\_45,  
\_46,  
\_47,  
\_48,  
\_49,  
\_50,  
\_51,  
\_52,  
\_53,  
\_54,  
\_55,  
\_56,  
\_57,  
\_58,  
\_59,

```

    _60,
    _61,
    _62,
    _63,
    N,
    ... ) N

```

### 17.235.1.32 PP\_RSEQ\_N

```
#define PP_RSEQ_N( )
```

#### Value:

```

63, 62, 61, 60, \
59, 58, 57, 56, 55, 54, 53, 52, 51, 50, \
49, 48, 47, 46, 45, 44, 43, 42, 41, 40, \
39, 38, 37, 36, 35, 34, 33, 32, 31, 30, \
29, 28, 27, 26, 25, 24, 23, 22, 21, 20, \
19, 18, 17, 16, 15, 14, 13, 12, 11, 10, \
9, 8, 7, 6, 5, 4, 3, 2, 1, 0

```

### 17.235.1.33 NOSPLIT

```
#define NOSPLIT( ) FFLAS::ParSeqHelper::Sequential()
```

### 17.235.1.34 splitting\_0

```
#define splitting_0( ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, FFLAS::StrategyParameter::Threading>(a)
```

### 17.235.1.35 splitting\_1

```
#define splitting_1(
    a ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, FFLAS::StrategyParameter::Threading>(a)
```

### 17.235.1.36 splitting\_2

```
#define splitting_2(
    a,
    c ) FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block, c>(a)
```

### 17.235.1.37 splitting\_3

```
#define splitting_3(
    a,
    b,
    c ) FFLAS::ParSeqHelper::Parallel<b, c>(a)
```

### 17.235.1.38 splitt

```
#define splitt(
    _1,
    _2,
    _3,
    NAME,
    ... ) NAME
```

### 17.235.1.39 SPLITTER

```
#define SPLITTER(
    ... ) splitt(__VA_ARGS__, splitting_3, splitting_2, splitting_1, splitting_0) (↔
__VA_ARGS__)
```

## 17.236 pfgemm\_variants.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Functions

- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement\_ptr A, const size_t lda, const typename Field::ConstElement\_ptr B, const size_t ldb,`  
`const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field, Al-`  
`goT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Block, StrategyParameter::Threads > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement\_ptr AA, const size_t lda, const typename Field::ConstElement\_ptr BB, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDAdaptive`  
`> > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement\_ptr AA, const size_t lda, const typename Field::ConstElement\_ptr BB, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoDAdaptive`  
`> > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement\_ptr AA, const size_t lda, const typename Field::ConstElement\_ptr BB, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element *C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::TwoD > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element\_ptr pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE`  
`tb, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement\_ptr A, const size_t lda, const typename Field::ConstElement\_ptr B, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size_t ldc, MMHelper< Field,`  
`AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeD > > &H)`
- `template<class Field, class AlgoT, class FieldTrait >`  
`Field::Element * pfgemm (const Field &F, const FFLAS_TRANSPOSE ta, const FFLAS_TRANSPOSE tb,`  
`const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename`  
`Field::ConstElement\_ptr A, const size_t lda, const typename Field::ConstElement\_ptr B, const size_t`  
`ldb, const typename Field::Element beta, typename Field::Element\_ptr C, const size_t ldc, MMHelper< Field, Al-`  
`goT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::ThreeDInPlace`  
`> > &H)`

## 17.237 pfgemv.inl File Reference

### Namespaces

- namespace [FFLAS](#)

### Functions

- template<class [Field](#) , class AlgoT , class FieldTrait >  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [Field](#), AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Recursive, StrategyParameter::Threads > > &H)
- template<class [Field](#) , class AlgoT , class FieldTrait , class Cut >  
[Field::Element\\_ptr fgemv](#) (const [Field](#) &F, const FFLAS\_TRANSPOSE ta, const size\_t m, const size\_t n, const typename [Field::Element](#) alpha, const typename [Field::ConstElement\\_ptr](#) A, const size\_t lda, const typename [Field::ConstElement\\_ptr](#) X, const size\_t incX, const typename [Field::Element](#) beta, typename [Field::Element\\_ptr](#) Y, const size\_t incY, MMHelper< [Field](#), AlgoT, FieldTrait, ParSeqHelper::Parallel< CuttingStrategy::Row, Cut > > &H)

## 17.238 align-allocator.h File Reference

```
#include "fflas-ffpack/config.h"
```

## 17.239 args-parser.h File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/givinteger.h>
#include <givaro/givprint.h>
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <cstring>
#include <list>
#include <stdlib.h>
```

### Data Structures

- struct [Argument](#)

### Namespaces

- namespace [FFLAS](#)

### Macros

- #define [TYPE\\_BOOL](#) [TYPE\\_NONE](#)
- #define [END\\_OF\\_ARGUMENTS](#) { '\0', '\0', '\0', [TYPE\\_NONE](#), NULL }
- #define [type\\_integer](#) long int

## Enumerations

- enum [ArgumentType](#) {  
[TYPE\\_NONE](#) , [TYPE\\_INT](#) , [TYPE\\_UINT64](#) , [TYPE\\_LONGLONG](#) ,  
[TYPE\\_INTEGER](#) , [TYPE\\_DOUBLE](#) , [TYPE\\_INTLIST](#) , [TYPE\\_STR](#) }

## Functions

- void [parseArguments](#) (int argc, char \*\*argv, [Argument](#) \*args, bool printDefaults=true)
- void [printHelpMessage](#) (const char \*program, [Argument](#) \*args, bool printDefaults=false)
- [Argument](#) \* [findArgument](#) ([Argument](#) \*args, char c)
- int [getListArgs](#) (std::list< int > &outlist, std::string &instring)  
*transforms a string list of ints to a list of int string "12,13,15" is turned into list of ints {12,13,15}*
- std::ostream & [writeCommandString](#) (std::ostream &os, [Argument](#) \*args, const char \*programName=nullptr)  
*writes the values of all arguments, preceded by the programName*

## 17.239.1 Macro Definition Documentation

### 17.239.1.1 TYPE\_BOOL

```
#define TYPE_BOOL TYPE\_NONE
```

### 17.239.1.2 END\_OF\_ARGUMENTS

```
#define END_OF_ARGUMENTS { '\0', "\0", "\0", TYPE\_NONE, NULL }
```

### 17.239.1.3 type\_integer

```
#define type_integer long int
```

## 17.239.2 Enumeration Type Documentation

### 17.239.2.1 ArgumentType

```
enum ArgumentType
```

#### Enumerator

|               |  |
|---------------|--|
| TYPE_NONE     |  |
| TYPE_INT      |  |
| TYPE_UINT64   |  |
| TYPE_LONGLONG |  |
| TYPE_INTEGER  |  |
| TYPE_DOUBLE   |  |
| TYPE_INTLIST  |  |
| TYPE_STR      |  |

## 17.239.3 Function Documentation



**17.239.3.1 printHelpMessage()**

```
void printHelpMessage (
    const char * program,
    Argument * args,
    bool printDefaults = false )
```

**17.239.3.2 findArgument()**

```
Argument * findArgument (
    Argument * args,
    char c )
```

**17.239.3.3 getListArgs()**

```
int getListArgs (
    std::list< int > & outlist,
    std::string & instring )
```

transforms a string list of ints to a list of int string "12,13,15" is turned into list of ints {12,13,15}

**Parameters**

|                 |                      |
|-----------------|----------------------|
| <i>outlist</i>  | list once converted  |
| <i>instring</i> | list to be converted |

**Returns**

status message.

**17.240 bit\_manipulation.h File Reference**

```
#include <givaro/udl.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
```

**Macros**

- #define `__has_builtin(x)` 0

**Functions**

- `int32_t clz (uint64_t val)`
- `int32_t clz (uint32_t val)`
- `int32_t ctz (uint32_t val)`
- `int32_t ctz (uint64_t val)`

**17.240.1 Macro Definition Documentation****17.240.1.1 \_\_has\_builtin**

```
#define __has_builtin(
    x ) 0
```

## 17.240.2 Function Documentation

### 17.240.2.1 `clz()` [1/2]

```
int32_t clz (
    uint64_t val ) [inline]
```

### 17.240.2.2 `clz()` [2/2]

```
int32_t clz (
    uint32_t val ) [inline]
```

### 17.240.2.3 `ctz()` [1/2]

```
int32_t ctz (
    uint32_t val ) [inline]
```

### 17.240.2.4 `ctz()` [2/2]

```
int32_t ctz (
    uint64_t val ) [inline]
```

## 17.241 `cast.h` File Reference

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Functions

- template<class T, class CT = const T>  
T [fflas\\_const\\_cast](#) (CT x)

## 17.242 `debug.h` File Reference

Various utilities for debugging.

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <iostream>
#include <sstream>
#include <cmath>
#include <stdexcept>
```

### Data Structures

- class [Failure](#)  
*A precondition failed.*

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Macros

- #define [FFLASFFPACK\\_check](#)(check)
- #define [FFLASFFPACK\\_abort](#)(msg)

## Functions

- Failure & [failure](#) ()
- template<class T >  
bool [isOdd](#) (const T &a)
- bool [isOdd](#) (const float &a)
- bool [isOdd](#) (const double &a)

### 17.242.1 Detailed Description

Various utilities for debugging.

**Todo** we should put vector printing elsewhere.

### 17.242.2 Macro Definition Documentation

#### 17.242.2.1 FFLASFFPACK\_check

```
#define FFLASFFPACK_check(  
    check )
```

**Value:**

```
if (!(check)) {\n    FFPACK::failure() (__func__, __FILE__, __LINE__, #check); \n    throw std::runtime_error(#check); \n}
```

#### 17.242.2.2 FFLASFFPACK\_abort

```
#define FFLASFFPACK_abort(  
    msg )
```

**Value:**

```
{\n    FFPACK::failure() (__func__, __FILE__, __LINE__, msg); \n    throw std::runtime_error(msg); \n}
```

## 17.243 fflas\_intrinsic.h File Reference

### 17.244 fflas\_io.h File Reference

```
#include <cstring>\n#include <stdio.h>\n#include <stdlib.h>\n#include <fstream>\n#include "fflas-ffpack/fflas/fflas.h"\n#include "fflas_memory.h"
```

## Namespaces

- namespace [FFLAS](#)

## Enumerations

- enum `FFLAS_FORMAT` {  
`FflasAuto` = 0 , `FflasDense` = 1 , `FflasSMS` = 2 , `FflasBinary` = 3 ,  
`FflasMath` = 4 , `FflasMaple` = 5 , `FflasSageMath` = 6 }

## Functions

- template<class `Field` >  
`std::ostream & WriteMatrix` (`std::ostream &c`, const `Field &F`, `size_t m`, `size_t n`, typename `Field::ConstElement_ptr` `A`, `size_t lda`, `FFLAS_FORMAT` `format`, bool `column_major`)  
*WriteMatrix: write a matrix to an output stream.*
- void `preamble` (`std::ifstream &ifs`, `FFLAS_FORMAT` &`format`)
- template<class `Field` >  
`Field::Element_ptr ReadMatrix` (`std::ifstream &ifs`, `Field &F`, `size_t &m`, `size_t &n`, typename `Field::Element_ptr` &`A`, `FFLAS_FORMAT` `format=FflasAuto`)  
*ReadMatrix: read a matrix from an input stream.*
- template<class `Field` >  
`Field::Element_ptr ReadMatrix` (const `std::string &matrix_file`, `Field &F`, `size_t &m`, `size_t &n`, typename `Field::Element_ptr` &`A`, `FFLAS_FORMAT` `format=FflasAuto`)  
*ReadMatrix: read a matrix from a file.*
- template<class `Field` >  
void `WriteMatrix` (`std::string &matrix_file`, const `Field &F`, int `m`, int `n`, typename `Field::ConstElement_ptr` `A`, `size_t lda`, `FFLAS_FORMAT` `format=FflasDense`, bool `column_major=false`)  
*WriteMatrix: write a matrix to a file.*
- `std::ostream & WritePermutation` (`std::ostream &c`, const `size_t *P`, `size_t N`)  
*WritePermutation: write a permutation matrix to an output stream.*

## 17.245 fflas\_memory.h File Reference

```
#include "fflas-ffpack/utils/align-allocator.h"
#include <givaro/givinteger.h>
```

## Namespaces

- namespace `FFLAS`

## Functions

- template<class `Element` >  
bool `alignable` ()
- template<> bool `alignable`< `Givaro::Integer *` > ()
- template<class `Field` >  
`Field::Element_ptr fflas_new` (const `Field &F`, const `size_t m`, const `Alignment` `align=Alignment::DEFAULT`)
- template<class `Field` >  
`Field::Element_ptr fflas_new` (const `Field &F`, const `size_t m`, const `size_t n`, const `Alignment` `align=Alignment::DEFAULT`)
- template<class `Element` >  
`Element * fflas_new` (const `size_t m`, const `Alignment` `align=Alignment::DEFAULT`)
- template<class `Element_ptr` >  
void `fflas_delete` (`Element_ptr A`)
- template<class `Ptr` , class ... `Args`>  
void `fflas_delete` (`Ptr p`, `Args ... args`)
- void `prefetch` (const `int64_t *`)
- void `getTLBSize` (int &`tlb`)

- void [queryCacheSizes](#) (int &l1, int &l2, int &l3)
- int [queryL1CacheSize](#) ()
- int [queryTopLevelCacheSize](#) ()

## 17.246 fflas\_randommatrix.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/fflas/fflas.h"
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include <givaro/givranditer.h>
#include "fflas-ffpack/ffpack/ffpack.h"
```

### Namespaces

- namespace [FFPACK](#)  
*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

### Functions

- template<class [Field](#) , class RandIter >  
[Field::Element\\_ptr NonZeroRandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda, RandIter &G)  
*Random non-zero Matrix.*
- template<class [Field](#) , class RandIter >  
[Field::Element\\_ptr NonZeroRandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda)  
*Random non-zero Matrix.*
- template<class [Field](#) , class RandIter >  
[Field::Element\\_ptr RandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda, RandIter &G)  
*Random Matrix.*
- template<class [Field](#) >  
[Field::Element\\_ptr RandomMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, typename [Field::Element\\_ptr](#) A, size\_t lda)  
*Random Matrix.*
- template<class [Field](#) , class RandIter >  
[Field::Element\\_ptr RandomTriangularMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_DIAG](#) Diag, bool nonsingular, typename [Field::Element\\_ptr](#) A, size\_t lda, RandIter &G)  
*Random Triangular Matrix.*
- template<class [Field](#) >  
[Field::Element\\_ptr RandomTriangularMatrix](#) (const [Field](#) &F, size\_t m, size\_t n, const [FFLAS::FFLAS\\_UPLO](#) UpLo, const [FFLAS::FFLAS\\_DIAG](#) Diag, bool nonsingular, typename [Field::Element\\_ptr](#) A, size\_t lda)  
*Random Triangular Matrix.*
- size\_t [RandInt](#) (size\_t a, size\_t b)
- template<class [Field](#) , class RandIter >  
[Field::Element\\_ptr RandomSymmetricMatrix](#) (const [Field](#) &F, size\_t n, bool nonsingular, typename [Field::Element\\_ptr](#) A, size\_t lda, RandIter &G)  
*Random Symmetric Matrix.*
- template<class [Field](#) , class RandIter >  
[Field::Element\\_ptr RandomMatrixWithRank](#) (const [Field](#) &F, size\_t m, size\_t n, size\_t r, typename [Field::Element\\_ptr](#) A, size\_t lda, RandIter &G)

*Random Matrix with prescribed rank.*

- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRank` (const `Field` &F, size\_t m, size\_t n, size\_t r, typename `Field::Element_ptr` A, size\_t lda)

*Random Matrix with prescribed rank.*

- `size_t * RandomIndexSubset` (size\_t N, size\_t R, size\_t \*P)  
*Pick uniformly at random a sequence of  $R$  distinct elements from the set  $\{0, \dots, N - 1\}$  using Knuth's shuffle.*
- `size_t * RandomPermutation` (size\_t N, size\_t \*P)  
*Pick uniformly at random a permutation of size  $N$  stored in LAPACK format using Knuth's shuffle.*
- `void RandomRankProfileMatrix` (size\_t M, size\_t N, size\_t R, size\_t \*rows, size\_t \*cols)  
*Pick uniformly at random an  $R$ -subpermutation of dimension  $M \times N$  : a matrix with only  $R$  non-zeros equal to one, in a random rook placement.*
- `void swapval` (size\_t k, size\_t N, size\_t \*P, size\_t val)
- `void RandomSymmetricRankProfileMatrix` (size\_t N, size\_t R, size\_t \*rows, size\_t \*cols)  
*Pick uniformly at random a symmetric  $R$ -subpermutation of dimension  $N \times N$  : a symmetric matrix with only  $R$  non-zeros, all equal to one, in a random rook placement.*
- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomMatrixWithRankandRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, RandIter &G)  
*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*
- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRankandRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)  
*Random Matrix with prescribed rank and rank profile matrix Creates an  $m \times n$  matrix with random entries and rank  $r$ .*
- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP, RandIter &G)  
*Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .*
- `template<class Field >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, const size\_t \*RRP, const size\_t \*CRP)  
*Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an  $n \times n$  symmetric matrix with random entries and rank  $r$ .*
- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, RandIter &G)  
*Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithRankandRandomRPM` (const `Field` &F, size\_t M, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda)  
*Random Matrix with prescribed rank, with random rank profile matrix Creates an  $m \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda, RandIter &G)  
*Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*
- `template<class Field >`  
`Field::Element_ptr RandomSymmetricMatrixWithRankandRandomRPM` (const `Field` &F, size\_t N, size\_t R, typename `Field::Element_ptr` A, size\_t lda)  
*Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an  $n \times n$  matrix with random entries, rank  $r$  and with a rank profile matrix chosen uniformly at random.*

- `template<class Field >`  
`Field::Element_ptr RandomMatrixWithDet` (const `Field` &F, `size_t` n, const typename `Field::Element` d, type-  
name `Field::Element_ptr` A, `size_t` lda)  
*Random Matrix with prescribed det.*
- `template<class Field , class RandIter >`  
`Field::Element_ptr RandomMatrixWithDet` (const `Field` &F, `size_t` n, const typename `Field::Element` d, type-  
name `Field::Element_ptr` A, `size_t` lda, `RandIter` &G)  
*Random Matrix with prescribed det.*

## 17.247 flimits.h File Reference

```
#include <climits>
#include <limits>
#include <type_traits>
#include <givaro/givinteger.h>
```

### Data Structures

- `struct limits< unsigned char >`
- `struct limits< signed char >`
- `struct limits< char >`
- `struct limits< unsigned short int >`
- `struct limits< short int >`
- `struct limits< unsigned int >`
- `struct limits< int >`
- `struct limits< unsigned long >`
- `struct limits< long >`
- `struct limits< unsigned long long >`
- `struct limits< long long >`
- `struct limits< float >`
- `struct limits< double >`
- `struct limits< Givaro::Integer >`
- `struct limits< Reclnt::ruint< K > >`
- `struct limits< Reclnt::rint< K > >`

### Functions

- `template<class T , class E >`  
`std::enable_if< std::is_signed< T >::value==std::is_signed< E >::value, bool >::type in_range` (E e)
- `template<class T , class E >`  
`std::enable_if<(std::is_signed< T >::value)&&!std::is_signed< E >::value, bool >::type in_range` (E e)
- `template<class T , class E >`  
`std::enable_if<!std::is_signed< T >::value)&&std::is_signed< E >::value, bool >::type in_range` (E e)

## 17.247.1 Function Documentation

### 17.247.1.1 in\_range() [1/3]

```
std::enable_if< std::is_signed< T >::value==std::is_signed< E >::value, bool >::type in_↔
range (
    E e )
```

**17.247.1.2 in\_range() [2/3]**

```
std::enable_if<(std::is_signed< T >::value)&&!(std::is_signed< E >::value), bool >::type
in_range (
    E e )
```

**17.247.1.3 in\_range() [3/3]**

```
std::enable_if<!(std::is_signed< T >::value)&&(std::is_signed< E >::value), bool >::type
in_range (
    E e )
```

**17.248 Matio.h File Reference**

```
#include <cstring>
#include <stdio.h>
#include <stdlib.h>
#include "fflas_memory.h"
```

**Functions**

- template<class [Field](#) >  
[Field::Element\\_ptr read\\_field](#) (const [Field](#) &F, const char \*mat\_file, size\_t \*tni, size\_t \*tnj)
- template<class [Field](#) >  
std::ostream & [write\\_field](#) (const [Field](#) &F, std::ostream &c, typename [Field::ConstElement\\_ptr](#) E, int n, int m, int id, bool mapleFormat=false, bool column\_major=false)

**17.248.1 Function Documentation****17.248.1.1 read\_field()**

```
Field::Element\_ptr read_field (
    const Field & F,
    const char * mat_file,
    size_t * tni,
    size_t * tnj )
```

**17.248.1.2 write\_field()**

```
std::ostream & write_field (
    const Field & F,
    std::ostream & c,
    typename Field::ConstElement\_ptr E,
    int n,
    int m,
    int id,
    bool mapleFormat = false,
    bool column_major = false )
```

**17.249 test-utils.h File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/debug.h"
```



```
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <givaro/givinteger.h>
#include <givaro/givintprime.h>
#include <givaro/givranditer.h>
#include <random>
#include <functional>
```

## Namespaces

- namespace [FFLAS](#)
- namespace [FFPACK](#)

*Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

## Functions

- [uint64\\_t](#) [getSeed](#) ()
- template<typename [Field](#) >  
Givaro::Integer [maxFieldElt](#) ()
- template<> Givaro::Integer [maxFieldElt](#)< Givaro::ZRing< [Givaro::Integer](#) > > ()
- template<typename [Field](#) >  
[Field](#) \* [chooseField](#) (Givaro::Integer q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- template<> Givaro::ZRing< [int32\\_t](#) > \* [chooseField](#)< Givaro::ZRing< [int32\\_t](#) > > (Givaro::Integer q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- template<> Givaro::ZRing< [int64\\_t](#) > \* [chooseField](#)< Givaro::ZRing< [int64\\_t](#) > > (Givaro::Integer q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- template<> Givaro::ZRing< float > \* [chooseField](#)< Givaro::ZRing< float > > (Givaro::Integer q, [uint64\\_t](#) b, [uint64\\_t](#) seed)
- template<> Givaro::ZRing< double > \* [chooseField](#)< Givaro::ZRing< double > > (Givaro::Integer q, [uint64\\_t](#) b, [uint64\\_t](#) seed)

## 17.250 timer.h File Reference

```
#include <time.h>
#include <givaro/givtimer.h>
```

## Namespaces

- namespace [FFLAS](#)

## Typedefs

- typedef Givaro::Timer [Timer](#)
- typedef Givaro::BaseTimer [BaseTimer](#)
- typedef Givaro::UserTimer [UserTimer](#)
- typedef Givaro::SysTimer [SysTimer](#)

## 17.251 cblas.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

## Macros

- `#define __FFLASFFPACK_CONFIGURATION`
- `#define __FFLASFFPACK_HAVE_CBLAS 1`

## Functions

- `int main ()`

### 17.251.1 Macro Definition Documentation

#### 17.251.1.1 \_\_FFLASFFPACK\_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

#### 17.251.1.2 \_\_FFLASFFPACK\_HAVE\_CBLAS

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

### 17.251.2 Function Documentation

#### 17.251.2.1 main()

```
int main (  
    void )
```

## 17.252 clapack.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

## Macros

- `#define __FFLASFFPACK_CONFIGURATION`
- `#define __FFLASFFPACK_HAVE_LAPACK 1`
- `#define __FFLASFFPACK_HAVE_CLAPACK 1`

## Functions

- `int main ()`

### 17.252.1 Macro Definition Documentation

#### 17.252.1.1 \_\_FFLASFFPACK\_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

#### 17.252.1.2 \_\_FFLASFFPACK\_HAVE\_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

### 17.252.1.3 \_\_FFLASFFPACK\_HAVE\_CLAPACK

```
#define __FFLASFFPACK_HAVE_CLAPACK 1
```

## 17.252.2 Function Documentation

### 17.252.2.1 main()

```
int main (  
    void )
```

## 17.253 cuda.C File Reference

```
#include <stdio.h>  
#include <cuda_runtime.h>  
#include <cusparse.h>
```

### Functions

- int [main](#) ()

## 17.253.1 Function Documentation

### 17.253.1.1 main()

```
int main (  
    void )
```

## 17.254 fblas.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_CONFIGURATION](#)

### Functions

- void [dgemm](#)\_ (const char \*, const char \*, const int \*, const int \*, const int \*, const double \*, const double \*, const int \*, const double \*, const int \*, const double \*, double \*, const int \*)
- int [main](#) ()

## 17.254.1 Macro Definition Documentation

### 17.254.1.1 \_\_FFLASFFPACK\_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

## 17.254.2 Function Documentation

### 17.254.2.1 dgemm\_()

```
void dgemm_ (
    const char * ,
    const char * ,
    const int * ,
    const int * ,
    const int * ,
    const double * ,
    const double * ,
    const int * ,
    const double * ,
    const int * ,
    const double * ,
    double * ,
    const int * )
```

### 17.254.2.2 main()

```
int main (
    void )
```

## 17.255 gmp.C File Reference

```
#include <gmpxx.h>
```

### Functions

- int [main](#) ()

## 17.255.1 Function Documentation

### 17.255.1.1 main()

```
int main (
    void )
```

## 17.256 instrset.h File Reference

```
#include <stdlib.h>
```

### Data Structures

- class [Const\\_int\\_t< n >](#)
- class [Const\\_uint\\_t< n >](#)
- class [Static\\_error\\_check< bool >](#)
- class [Static\\_error\\_check< false >](#)

## Macros

- #define `INSTRSET_H` 125
- #define `INSTRSET` 0
- #define `const_int(n)` (`Const_int_t <n>()`)
- #define `const_uint(n)` (`Const_uint_t <n>()`)

## Typedefs

- typedef signed char `int8_t`
- typedef unsigned char `uint8_t`
- typedef signed short int `int16_t`
- typedef unsigned short int `uint16_t`
- typedef signed int `int32_t`
- typedef unsigned int `uint32_t`
- typedef long long `int64_t`
- typedef unsigned long long `uint64_t`
- typedef `int32_t` `intptr_t`

## Functions

- int `instrset_detect` (void)
- bool `hasFMA3` (void)
- bool `hasFMA4` (void)
- bool `hasXOP` (void)
- bool `hasAVX512ER` (void)

## 17.256.1 Macro Definition Documentation

### 17.256.1.1 INSTRSET\_H

```
#define INSTRSET_H 125
```

### 17.256.1.2 INSTRSET

```
#define INSTRSET 0
```

### 17.256.1.3 const\_int

```
#define const_int(  
    n ) (Const_int_t <n>())
```

### 17.256.1.4 const\_uint

```
#define const_uint(  
    n ) (Const_uint_t <n>())
```

## 17.256.2 Typedef Documentation

**17.256.2.1 int8\_t**

```
typedef signed char int8_t
```

**17.256.2.2 uint8\_t**

```
typedef unsigned char uint8_t
```

**17.256.2.3 int16\_t**

```
typedef signed short int int16_t
```

**17.256.2.4 uint16\_t**

```
typedef unsigned short int uint16_t
```

**17.256.2.5 int32\_t**

```
typedef signed int int32_t
```

**17.256.2.6 uint32\_t**

```
typedef unsigned int uint32_t
```

**17.256.2.7 int64\_t**

```
typedef long long int64_t
```

**17.256.2.8 uint64\_t**

```
typedef unsigned long long uint64_t
```

**17.256.2.9 intptr\_t**

```
typedef int32_t intptr_t
```

**17.256.3 Function Documentation****17.256.3.1 instrset\_detect()**

```
int instrset_detect (  
    void )
```

**17.256.3.2 hasFMA3()**

```
bool hasFMA3 (  
    void )
```

### 17.256.3.3 hasFMA4()

```
bool hasFMA4 (  
    void )
```

### 17.256.3.4 hasXOP()

```
bool hasXOP (  
    void )
```

### 17.256.3.5 hasAVX512ER()

```
bool hasAVX512ER (  
    void )
```

## 17.257 instrset\_detect.cpp File Reference

```
#include "instrset.h"
```

### Functions

- int [instrset\\_detect](#) (void)
- bool [hasFMA3](#) (void)
- bool [hasFMA4](#) (void)
- bool [hasXOP](#) (void)
- bool [hasF16C](#) (void)
- bool [hasAVX512ER](#) (void)

### 17.257.1 Function Documentation

#### 17.257.1.1 instrset\_detect()

```
int instrset_detect (  
    void )
```

#### 17.257.1.2 hasFMA3()

```
bool hasFMA3 (  
    void )
```

#### 17.257.1.3 hasFMA4()

```
bool hasFMA4 (  
    void )
```

#### 17.257.1.4 hasXOP()

```
bool hasXOP (  
    void )
```

### 17.257.1.5 hasF16C()

```
bool hasF16C (
    void )
```

### 17.257.1.6 hasAVX512ER()

```
bool hasAVX512ER (
    void )
```

## 17.258 lapack.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

### Macros

- `#define __FFLASFFPACK_CONFIGURATION`
- `#define __FFLASFFPACK_HAVE_LAPACK 1`

### Functions

- `int main ()`

## 17.258.1 Macro Definition Documentation

### 17.258.1.1 \_\_FFLASFFPACK\_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

### 17.258.1.2 \_\_FFLASFFPACK\_HAVE\_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

## 17.258.2 Function Documentation

### 17.258.2.1 main()

```
int main (
    void )
```

## 17.259 regression-check.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack.h"
```



## Functions

- bool [check1](#) ()
- bool [check2](#) ()
- bool [check3](#) ()
- bool [check4](#) ()
- bool [checkZeroDimCharpoly](#) ()
- bool [checkZeroDimMinPoly](#) ()
- bool [gf2ModularBalanced](#) ()
- int [main](#) ()

### 17.259.1 Function Documentation

#### 17.259.1.1 [check1\(\)](#)

```
bool check1 ( )
```

#### 17.259.1.2 [check2\(\)](#)

```
bool check2 ( )
```

#### 17.259.1.3 [check3\(\)](#)

```
bool check3 ( )
```

#### 17.259.1.4 [check4\(\)](#)

```
bool check4 ( )
```

#### 17.259.1.5 [checkZeroDimCharpoly\(\)](#)

```
bool checkZeroDimCharpoly ( )
```

#### 17.259.1.6 [checkZeroDimMinPoly\(\)](#)

```
bool checkZeroDimMinPoly ( )
```

#### 17.259.1.7 [gf2ModularBalanced\(\)](#)

```
bool gf2ModularBalanced ( )
```

#### 17.259.1.8 [main\(\)](#)

```
int main (
    void )
```

## 17.260 test-charpoly-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

### Macros

- `#define` [ENABLE\\_CHECKER\\_charpoly](#) 1
- `#define` [TIME\\_CHECKER\\_CHARPOLY](#) 1

### Functions

- `template<class Field, class Polynomial >`  
`void printPolynomial (const Field &F, Polynomial &v)`
- `int main (int argc, char **argv)`

## 17.260.1 Macro Definition Documentation

### 17.260.1.1 [ENABLE\\_CHECKER\\_charpoly](#)

```
#define ENABLE_CHECKER_charpoly 1
```

### 17.260.1.2 [TIME\\_CHECKER\\_CHARPOLY](#)

```
#define TIME_CHECKER_CHARPOLY 1
```

## 17.260.2 Function Documentation

### 17.260.2.1 [printPolynomial\(\)](#)

```
void printPolynomial (
    const Field & F,
    Polynomial & v )
```

### 17.260.2.2 [main\(\)](#)

```
int main (
    int argc,
    char ** argv )
```

## 17.261 test-charpoly.C File Reference

```
#include <iostream>
#include <iomanip>
#include "givaro/modular.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

```
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <random>
#include <chrono>
```

## Functions

- template<class [Field](#), class RandIter >  
bool [launch\\_test](#) (const [Field](#) &F, size\_t n, typename [Field::Element](#) \*A, size\_t lda, size\_t nbit, RandIter &G, FFPACK::FFPACK\_CHARPOLY\_TAG CT)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (const Givaro::Integer p, [uint64\\_t](#) bits, size\_t n, std::string file, int variant, size\_t iter, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.261.1 Function Documentation

### 17.261.1.1 [launch\\_test\(\)](#)

```
bool launch_test (
    const Field & F,
    size_t n,
    typename Field::Element * A,
    size_t lda,
    size_t nbit,
    RandIter & G,
    FFPACK::FFPACK_CHARPOLY_TAG CT )
```

### 17.261.1.2 [run\\_with\\_field\(\)](#)

```
bool run_with_field (
    const Givaro::Integer p,
    uint64\_t bits,
    size_t n,
    std::string file,
    int variant,
    size_t iter,
    uint64\_t seed )
```

### 17.261.1.3 [main\(\)](#)

```
int main (
    int argc,
    char ** argv )
```

## 17.262 test-compressQ.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <list>
#include <vector>
```

```
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
```

## Typedefs

- typedef Givaro::Modular< double > [Field](#)

## Functions

- template<class T >  
std::ostream & [printvect](#) (std::ostream &o, vector< T > &vect)
- int [main](#) (int argc, char \*\*argv)

### 17.262.1 Typedef Documentation

#### 17.262.1.1 Field

```
typedef Givaro::Modular<double> Field
```

### 17.262.2 Function Documentation

#### 17.262.2.1 printvect()

```
std::ostream & printvect (
    std::ostream & o,
    vector< T > & vect )
```

[Bug](#) does not belong here

#### 17.262.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.263 test-det-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/checkers/checkers_ffpack.h"
#include "fflas-ffpack/checkers/checkers_ffpack.inl"
```

## Macros

- #define [ENABLE\\_CHECKER\\_Det](#) 1
- #define [TIME\\_CHECKER\\_Det](#) 1

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.263.1 Macro Definition Documentation

#### 17.263.1.1 [ENABLE\\_CHECKER\\_Det](#)

```
#define ENABLE_CHECKER_Det 1
```

#### 17.263.1.2 [TIME\\_CHECKER\\_Det](#)

```
#define TIME_CHECKER_Det 1
```

### 17.263.2 Function Documentation

#### 17.263.2.1 [main\(\)](#)

```
int main (  
    int argc,  
    char ** argv )
```

## 17.264 test-det.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"  
#include <iomanip>  
#include <iostream>  
#include <givaro/modular-balanced.h>  
#include "fflas-ffpack/utils/timer.h"  
#include "fflas-ffpack/ffpack/ffpack.h"  
#include "fflas-ffpack/utils/args-parser.h"  
#include "fflas-ffpack/utils/test-utils.h"
```

## Functions

- template<class [Field](#) , class RandIter >  
 bool [test\\_det](#) ([Field](#) &F, size\_t n, int iter, RandIter &G)
- int [main](#) (int argc, char \*\*argv)

### 17.264.1 Function Documentation

### 17.264.1.1 test\_det()

```
bool test_det (
    Field & F,
    size_t n,
    int iter,
    RandIter & G )
```

**Todo** test with stride

### 17.264.1.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.265 test-echelon.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <iomanip>
#include <givaro/modular-balanced.h>
#include <givaro/udl.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include <random>
#include <chrono>
```

## Macros

- `#define __FFLASFFPACK_SEQUENTIAL`
- `#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25`
- `#define __FFLASFFPACK_PLUQ_THRESHOLD 25`

## Functions

- `template<class Field , class RandIter >`  
`bool test_colechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)`
- `template<class Field , class RandIter >`  
`bool test_rowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)`
- `template<class Field , class RandIter >`  
`bool test_redcoechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)`
- `template<class Field , class RandIter >`  
`bool test_redrowechelon (Field &F, size_t m, size_t n, size_t r, size_t iters, FFPACK::FFPACK_LU_TAG LuTag, RandIter &G, bool par)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.265.1 Macro Definition Documentation

### 17.265.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.265.1.2 \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25
```

### 17.265.1.3 \_\_FFLASFFPACK\_PLUQ\_THRESHOLD

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 25
```

## 17.265.2 Function Documentation

### 17.265.2.1 test\_colechelon()

```
bool test_colechelon (  
    Field & F,  
    size_t m,  
    size_t n,  
    size_t r,  
    size_t iters,  
    FFPACK::FFPACK_LU_TAG LuTag,  
    RandIter & G,  
    bool par )
```

**Todo** check Ida

### 17.265.2.2 test\_rowechelon()

```
bool test_rowechelon (  
    Field & F,  
    size_t m,  
    size_t n,  
    size_t r,  
    size_t iters,  
    FFPACK::FFPACK_LU_TAG LuTag,  
    RandIter & G,  
    bool par )
```

**Todo** check Ida

### 17.265.2.3 test\_redcolechelon()

```
bool test_redcolechelon (  
    Field & F,  
    size_t m,  
    size_t n,
```

```

    size_t r,
    size_t iters,
    FFPACK::FFPACK_LU_TAG LuTag,
    RandIter & G,
    bool par )

```

**Todo** check Ida

#### 17.265.2.4 test\_redrowechelon()

```

bool test_redrowechelon (
    Field & F,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
    FFPACK::FFPACK_LU_TAG LuTag,
    RandIter & G,
    bool par )

```

**Todo** check Ida

#### 17.265.2.5 run\_with\_field()

```

bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
    uint64_t seed )

```

#### 17.265.2.6 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.266 test-fadd.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"

```

## Functions

- template<class [Field](#) >  
bool [test\\_fadd](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)



- `template<class Field >`  
`bool test_faddin (const Field &F, size_t m, size_t k, size_t n, bool timing, uint64_t seed)`
- `template<class Field >`  
`bool test_fsub (const Field &F, size_t m, size_t k, size_t n, bool timing, uint64_t seed)`
- `template<class Field >`  
`bool test_fsubin (const Field &F, size_t m, size_t k, size_t n, bool timing, uint64_t seed)`
- `int main (int ac, char **av)`

## 17.266.1 Function Documentation

### 17.266.1.1 test\_fadd()

```
bool test_fadd (  
    const Field & F,  
    size_t m,  
    size_t k,  
    size_t n,  
    bool timing,  
    uint64_t seed )
```

### 17.266.1.2 test\_faddin()

```
bool test_faddin (  
    const Field & F,  
    size_t m,  
    size_t k,  
    size_t n,  
    bool timing,  
    uint64_t seed )
```

### 17.266.1.3 test\_fsub()

```
bool test_fsub (  
    const Field & F,  
    size_t m,  
    size_t k,  
    size_t n,  
    bool timing,  
    uint64_t seed )
```

### 17.266.1.4 test\_fsubin()

```
bool test_fsubin (  
    const Field & F,  
    size_t m,  
    size_t k,  
    size_t n,  
    bool timing,  
    uint64_t seed )
```

**17.266.1.5 main()**

```
int main (
    int ac,
    char ** av )
```

**17.267 test-fdot.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/paladin/parallel.h"
#include "fflas-ffpack/paladin/fflas_plevel1.h"
#include <givaro/zring.h>
#include <givaro/modular.h>
#include <random>
#include <chrono>
```

**Macros**

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

**Functions**

- template<typename [Field](#) >  
bool [check\\_fdot](#) (const [Field](#) &F, size\_t n, typename [Field::ConstElement\\_ptr](#) a, size\_t inca, typename [Field::ConstElement\\_ptr](#) b, size\_t incb)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t BS, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- bool [run\\_with\\_Integer](#) (size\_t BS, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

**17.267.1 Macro Definition Documentation****17.267.1.1 ENABLE\_ALL\_CHECKINGS**

```
#define ENABLE_ALL_CHECKINGS 1
```

**17.267.2 Function Documentation****17.267.2.1 check\_fdot()**

```
bool check_fdot (
    const Field & F,
    size_t n,
    typename Field::ConstElement\_ptr a,
    size_t inca,
    typename Field::ConstElement\_ptr b,
    size_t incb )
```

### 17.267.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t BS,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.267.2.3 run\_with\_Integer()

```
bool run_with_Integer (
    size_t BS,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.267.2.4 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.268 test-fgemm-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utis/args-parser.h"
#include "fflas-ffpack/utis/test-utis.h"
```

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- template<class [Field](#) , class RandIter >  
bool [launch\\_MM\\_dispatch](#) (const [Field](#) &F, const int mm, const int nn, const int kk, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size\_t iters, RandIter &G)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, [uint64\\_t](#) b, int m, int n, int k, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.268.1 Macro Definition Documentation

### 17.268.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.268.2 Function Documentation

**17.268.2.1 launch\_MM\_dispatch()**

```
bool launch_MM_dispatch (
    const Field & F,
    const int mm,
    const int nn,
    const int kk,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t iters,
    RandIter & G )
```

**Bug** test for ldX equal

**Bug**

**Bug** test for transpo

**Bug**

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

**17.268.2.2 run\_with\_field()**

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    int m,
    int n,
    int k,
    size_t iters,
    uint64_t seed )
```

**17.268.2.3 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.269 test-fgemm.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <random>
```

**Macros**

- #define `ENABLE_CHECKER_fgemm` 1

## Functions

- `template<class Field >`  
`bool check_MM (const Field &F, const typename Field::Element_ptr Cd, enum FFLAS_TRANSPOSE &ta, enum FFLAS_TRANSPOSE &tb, const size_t m, const size_t n, const size_t k, const typename Field::Element &alpha, const typename Field::Element_ptr A, size_t lda, const typename Field::Element_ptr B, size_t ldb, const typename Field::Element &beta, const typename Field::Element_ptr C, size_t ldc)`
- `template<class Field, class RandIter >`  
`bool launch_MM (const Field &F, const size_t m, const size_t n, const size_t k, const typename Field::Element alpha, const typename Field::Element beta, const size_t ldc, const size_t lda, enum FFLAS_TRANSPOSE ta, const size_t ldb, enum FFLAS_TRANSPOSE tb, size_t iters, int nbw, bool par, RandIter &G)`
- `template<class Field, class RandIter >`  
`bool launch_MM_dispatch (const Field &F, const int mm, const int nn, const int kk, const typename Field::Element alpha, const typename Field::Element beta, const size_t iters, const int nbw, const bool par, RandIter &G)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, int m, int n, int k, int nbw, size_t iters, bool par, size_t seed)`
- `int main (int argc, char **argv)`

### 17.269.1 Macro Definition Documentation

#### 17.269.1.1 ENABLE\_CHECKER\_fgemm

```
#define ENABLE_CHECKER_fgemm 1
```

### 17.269.2 Function Documentation

#### 17.269.2.1 check\_MM()

```
bool check_MM (
    const Field & F,
    const typename Field::Element_ptr Cd,
    enum FFLAS_TRANSPOSE & ta,
    enum FFLAS_TRANSPOSE & tb,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element_ptr A,
    size_t lda,
    const typename Field::Element_ptr B,
    size_t ldb,
    const typename Field::Element & beta,
    const typename Field::Element_ptr C,
    size_t ldc )
```

#### 17.269.2.2 launch\_MM()

```
bool launch_MM (
    const Field & F,
    const size_t m,
    const size_t n,
    const size_t k,
    const typename Field::Element alpha,
```

```

const typename Field::Element beta,
const size_t ldc,
const size_t lda,
enum FFLAS_TRANSPOSE ta,
const size_t ldb,
enum FFLAS_TRANSPOSE tb,
size_t iters,
int nbw,
bool par,
RandIter & G )

```

### 17.269.2.3 launch\_MM\_dispatch()

```

bool launch_MM_dispatch (
    const Field & F,
    const int mm,
    const int nn,
    const int kk,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t iters,
    const int nbw,
    const bool par,
    RandIter & G )

```

**Bug** test for ldX equal

**Bug**

**Bug** test for transpo

**Bug**

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

### 17.269.2.4 run\_with\_field()

```

bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    int m,
    int n,
    int k,
    int nbw,
    size_t iters,
    bool par,
    size_t seed )

```

### 17.269.2.5 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.270 test-fgemv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

### Functions

- template<class [Field](#) >  
bool [check\\_MV](#) (const [Field](#) &F, const typename [Field::Element\\_ptr](#) Cd, enum [FFLAS\\_TRANSPOSE](#) &ta, const size\_t m, const size\_t k, const typename [Field::Element](#) &alpha, const typename [Field::Element\\_ptr](#) A, size\_t lda, const typename [Field::Element\\_ptr](#) X, size\_t incX, const typename [Field::Element](#) &beta, const typename [Field::Element\\_ptr](#) Y, size\_t incY)
- template<class [Field](#) , class RandIter >  
bool [launch\\_MV](#) (const [Field](#) &F, const size\_t m, const size\_t k, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size\_t lda, enum [FFLAS\\_TRANSPOSE](#) ta, const size\_t incX, const size\_t incY, size\_t iters, bool par, RandIter &G)
- template<class [Field](#) , class RandIter >  
bool [launch\\_MV\\_dispatch](#) (const [Field](#) &F, const int mm, const int kk, const typename [Field::Element](#) alpha, const typename [Field::Element](#) beta, const size\_t iters, const bool par, RandIter &G)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, [uint64\\_t](#) b, int m, int k, size\_t iters, bool par, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

### 17.270.1 Function Documentation

#### 17.270.1.1 [check\\_MV\(\)](#)

```
bool check_MV (
    const Field & F,
    const typename Field::Element\_ptr Cd,
    enum FFLAS\_TRANSPOSE & ta,
    const size_t m,
    const size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element\_ptr A,
    size_t lda,
    const typename Field::Element\_ptr X,
    size_t incX,
    const typename Field::Element & beta,
    const typename Field::Element\_ptr Y,
    size_t incY )
```

#### 17.270.1.2 [launch\\_MV\(\)](#)

```
bool launch_MV (
    const Field & F,
    const size_t m,
```

```

    const size_t k,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t lda,
    enum FFLAS_TRANSPOSE ta,
    const size_t incX,
    const size_t incY,
    size_t iters,
    bool par,
    RandIter & G )

```

### 17.270.1.3 launch\_MV\_dispatch()

```

bool launch_MV_dispatch (
    const Field & F,
    const int mm,
    const int kk,
    const typename Field::Element alpha,
    const typename Field::Element beta,
    const size_t iters,
    const bool par,
    RandIter & G )

```

### 17.270.1.4 run\_with\_field()

```

bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    int m,
    int k,
    size_t iters,
    bool par,
    uint64_t seed )

```

### 17.270.1.5 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.271 test-fger.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <givaro/modular-integral.h>
#include <givaro/modular-balanced.h>
#include <givaro/givintprime.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"

```



## Macros

- `#define TIME 1`

## Functions

- `template<class Field >`  
`bool check_fger (const Field &F, const typename Field::Element_ptr Cd, const size_t m, const size_t n, const`  
`typename Field::Element &alpha, const typename Field::Element_ptr x, const size_t incx, const typename`  
`Field::Element_ptr y, const size_t incy, const typename Field::Element_ptr C, const size_t ldc)`
- `template<class Field, class Randlter >`  
`bool launch_fger (const Field &F, const size_t m, const size_t n, const typename Field::Element alpha, const`  
`size_t ldc, const size_t inca, const size_t incb, size_t iters, Randlter &G)`
- `template<class Field, class Randlter >`  
`bool launch_fger_dispatch (const Field &F, const size_t nn, const typename Field::Element alpha, const`  
`size_t iters, Randlter &G)`
- `template<class Field >`  
`bool run_with_field (int64_t q, uint64_t b, size_t n, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.271.1 Macro Definition Documentation

### 17.271.1.1 TIME

```
#define TIME 1
```

## 17.271.2 Function Documentation

### 17.271.2.1 check\_fger()

```
bool check_fger (
    const Field & F,
    const typename Field::Element_ptr Cd,
    const size_t m,
    const size_t n,
    const typename Field::Element & alpha,
    const typename Field::Element_ptr x,
    const size_t incx,
    const typename Field::Element_ptr y,
    const size_t incy,
    const typename Field::Element_ptr C,
    const size_t ldc )
```

### 17.271.2.2 launch\_fger()

```
bool launch_fger (
    const Field & F,
    const size_t m,
    const size_t n,
    const typename Field::Element alpha,
    const size_t ldc,
    const size_t inca,
    const size_t incb,
```

```
size_t iters,
RandIter & G )
```

### 17.271.2.3 launch\_fger\_dispatch()

```
bool launch_fger_dispatch (
    const Field & F,
    const size_t nn,
    const typename Field::Element alpha,
    const size_t iters,
    RandIter & G )
```

**Bug** test for incx equal

**Bug**

**Bug** test for transpo

**Bug**

**Todo** does nbw actually do nbw recursive calls and then call blas (check ?) ?

### 17.271.2.4 run\_with\_field()

```
bool run_with_field (
    int64_t q,
    uint64_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.271.2.5 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.272 test-fgesv.C File Reference

```
#include <iostream>
#include <iomanip>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

## Functions

- template<class Field, class RandIter >  
bool **test\_square\_fgesv** (Field &F, FFLAS\_SIDE side, string fileA, string fileB, size\_t m, size\_t k, size\_t r, RandIter &G)
- template<class Field, class RandIter >  
bool **test\_rect\_fgesv** (Field &F, FFLAS\_SIDE side, string fileA, string fileB, size\_t m, size\_t n, size\_t k, size\_t r, RandIter &G)

- `template<class Field >`  
  `bool run_with_field` (Givaro::Integer *q*, `uint64_t` *b*, `size_t` *m*, `size_t` *n*, `size_t` *k*, `size_t` *r*, `size_t` *iters*, `string` *fileA*, `string` *fileB*, `uint64_t` &*seed*)
- `int main` (`int` *argc*, `char **`*argv*)

## 17.272.1 Function Documentation

### 17.272.1.1 test\_square\_fgesv()

```
bool test_square_fgesv (  
    Field & F,  
    FFLAS_SIDE side,  
    string fileA,  
    string fileB,  
    size_t m,  
    size_t k,  
    size_t r,  
    RandIter & G )
```

### 17.272.1.2 test\_rect\_fgesv()

```
bool test_rect_fgesv (  
    Field & F,  
    FFLAS_SIDE side,  
    string fileA,  
    string fileB,  
    size_t m,  
    size_t n,  
    size_t k,  
    size_t r,  
    RandIter & G )
```

### 17.272.1.3 run\_with\_field()

```
bool run_with_field (  
    Givaro::Integer q,  
    uint64_t b,  
    size_t m,  
    size_t n,  
    size_t k,  
    size_t r,  
    size_t iters,  
    string fileA,  
    string fileB,  
    uint64_t & seed )
```

### 17.272.1.4 main()

```
int main (  
    int argc,  
    char ** argv )
```

## 17.273 test-finit.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
#include <random>
#include <chrono>
```

### Functions

- template<class [Field](#) >  
bool [test\\_freduce](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t m, size\_t k, size\_t n, size\_t iters, bool timing, [uint64\\_t](#) seed)
- int [main](#) (int ac, char \*\*av)

### 17.273.1 Function Documentation

#### 17.273.1.1 test\_freduce()

```
bool test_freduce (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64\_t seed )
```

#### 17.273.1.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t k,
    size_t n,
    size_t iters,
    bool timing,
    uint64\_t seed )
```

#### 17.273.1.3 main()

```
int main (
    int ac,
    char ** av )
```

## 17.274 test-fscal.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <typeinfo>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "assert.h"
```

### Functions

- template<class [Field](#), class RandIter >  
bool [test\\_fscal](#) (const [Field](#) &F, const typename [Field::Element](#) &alpha, size\_t m, size\_t k, size\_t n, bool timing, RandIter &G)
- template<class [Field](#) >  
bool [test\\_fscal](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- template<class [Field](#), class RandIter >  
bool [test\\_fscalin](#) (const [Field](#) &F, const typename [Field::Element](#) &alpha, size\_t m, size\_t k, size\_t n, bool timing, RandIter &G)
- template<class [Field](#) >  
bool [test\\_fscalin](#) (const [Field](#) &F, size\_t m, size\_t k, size\_t n, bool timing, [uint64\\_t](#) seed)
- int [main](#) (int ac, char \*\*av)

### 17.274.1 Function Documentation

#### 17.274.1.1 test\_fscal() [1/2]

```
bool test_fscal (
    const Field & F,
    const typename Field::Element & alpha,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    RandIter & G )
```

#### 17.274.1.2 test\_fscal() [2/2]

```
bool test_fscal (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64\_t seed )
```

#### 17.274.1.3 test\_fscalin() [1/2]

```
bool test_fscalin (
    const Field & F,
    const typename Field::Element & alpha,
    size_t m,
```

```

    size_t k,
    size_t n,
    bool timing,
    RandIter & G )

```

#### 17.274.1.4 test\_fscalin() [2/2]

```

bool test_fscalin (
    const Field & F,
    size_t m,
    size_t k,
    size_t n,
    bool timing,
    uint64_t seed )

```

#### 17.274.1.5 main()

```

int main (
    int ac,
    char ** av )

```

## 17.275 test-fsyr2k.C File Reference

```

#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>

```

## Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- template<typename [Field](#) , class RandIter >  
bool [check\\_fsyr2k](#) (const [Field](#) &F, size\_t n, size\_t k, const typename [Field::Element](#) &alpha, const typename [Field::Element](#) &beta, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_TRANSPOSE](#) trans, RandIter &Rand)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t k, int a, int c, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.275.1 Macro Definition Documentation

### 17.275.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.275.2 Function Documentation

### 17.275.2.1 check\_fsyrr2k()

```
bool check_fsyrr2k (
    const Field & F,
    size_t n,
    size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element & beta,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
    RandIter & Rand )
```

### 17.275.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t k,
    int a,
    int c,
    size_t iters,
    uint64_t seed )
```

### 17.275.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.276 test-fsyrrk.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

## Macros

- #define `ENABLE_ALL_CHECKINGS` 1

## Functions

- template<typename `Field` , class RandIter >  
 bool `check_fsyrrk` (const `Field` &F, size\_t n, size\_t k, const typename `Field::Element` &alpha, const typename `Field::Element` &beta, `FFLAS::FFLAS_UPLO` uplo, `FFLAS::FFLAS_TRANSPOSE` trans, RandIter &Rand)

- `template<typename Field, class RandIter >`  
`bool check_fsyrk_diag (const Field &F, size_t n, size_t k, const typename Field::Element &alpha, const`  
`typename Field::Element &beta, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_TRANSPOSE trans, RandIter`  
`&Rand)`
- `template<typename Field, class RandIter >`  
`bool check_fsyrk_bkdiag (const Field &F, size_t n, size_t k, const typename Field::Element &alpha, const`  
`typename Field::Element &beta, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_TRANSPOSE trans, RandIter`  
`&Rand)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t k, int a, int c, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.276.1 Macro Definition Documentation

### 17.276.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.276.2 Function Documentation

### 17.276.2.1 check\_fsyrk()

```
bool check_fsyrk (
    const Field & F,
    size_t n,
    size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element & beta,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
    RandIter & Rand )
```

### 17.276.2.2 check\_fsyrk\_diag()

```
bool check_fsyrk_diag (
    const Field & F,
    size_t n,
    size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element & beta,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
    RandIter & Rand )
```

### 17.276.2.3 check\_fsyrk\_bkdiag()

```
bool check_fsyrk_bkdiag (
    const Field & F,
    size_t n,
    size_t k,
    const typename Field::Element & alpha,
    const typename Field::Element & beta,
    FFLAS::FFLAS_UPLO uplo,
```



```

    FFLAS::FFLAS_TRANSPOSE trans,
    RandIter & Rand )

```

#### 17.276.2.4 run\_with\_field()

```

bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t k,
    int a,
    int c,
    size_t iters,
    uint64_t seed )

```

#### 17.276.2.5 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.277 test-fsytrf.C File Reference

```

#include <iostream>
#include <iterator>
#include <vector>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include <iomanip>
#include <random>
#include <chrono>
#include <givaro/modular.h>
#include "fflas-ffpack/ffpack/ffpack.h"

```

## Functions

- template<typename T >  
std::ostream & operator<< (std::ostream &os, const std::vector< T > &x)
- template<class Field , class RandIter >  
bool test\_RPM\_fsytrf (Field &F, FFLAS\_UPLO uplo, string file, size\_t n, size\_t r, RandIter &G, size\_t threshold)
- template<class Field , class RandIter >  
bool test\_generic\_fsytrf (Field &F, FFLAS\_UPLO uplo, string file, size\_t n, RandIter &G, size\_t threshold)
- template<class Field >  
bool run\_with\_field (Givaro::Integer q, uint64\_t b, size\_t n, size\_t r, size\_t iters, string file, size\_t threshold, uint64\_t &seed)
- int main (int argc, char \*\*argv)

### 17.277.1 Function Documentation

**17.277.1.1 operator<<()**

```
std::ostream & operator<< (
    std::ostream & os,
    const std::vector< T > & x )
```

**17.277.1.2 test\_RPM\_fsytrf()**

```
bool test_RPM_fsytrf (
    Field & F,
    FFLAS_UPLO uplo,
    string file,
    size_t n,
    size_t r,
    RandIter & G,
    size_t threshold )
```

**17.277.1.3 test\_generic\_fsytrf()**

```
bool test_generic_fsytrf (
    Field & F,
    FFLAS_UPLO uplo,
    string file,
    size_t n,
    RandIter & G,
    size_t threshold )
```

**17.277.1.4 run\_with\_field()**

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t n,
    size_t r,
    size_t iters,
    string file,
    size_t threshold,
    uint64_t & seed )
```

**17.277.1.5 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.278 test-ffrmm.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
```

```
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

## Macros

- `#define __FFLASFFPACK_SEQUENTIAL`

## Functions

- `template<typename Field, class RandIter >`  
`bool check_ftmm (const Field &F, size_t m, size_t n, const typename Field::Element &alpha,`  
`FFLAS::FFLAS_SIDE side, FFLAS::FFLAS_UPLO uplo, FFLAS::FFLAS_TRANSPOSE trans, FFLAS::FFLAS_DIAG`  
`diag, RandIter &Rand)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t m, size_t n, uint64_t a, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.278.1 Macro Definition Documentation

### 17.278.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

## 17.278.2 Function Documentation

### 17.278.2.1 check\_ftmm()

```
bool check_ftmm (
    const Field & F,
    size_t m,
    size_t n,
    const typename Field::Element & alpha,
    FFLAS::FFLAS_SIDE side,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_TRANSPOSE trans,
    FFLAS::FFLAS_DIAG diag,
    RandIter & Rand )
```

### 17.278.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t n,
    uint64_t a,
    size_t iters,
    uint64_t seed )
```

### 17.278.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.279 test-ffrmv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <chrono>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

### Macros

- `#define __FFLASFFPACK_SEQUENTIAL`
- `#define ENABLE_ALL_CHECKINGS 1`

### Functions

- `template<typename Field, class RandIter >`  
`bool check_ffrmv (const Field &F, size_t n, FFLAS_UPLO uplo, FFLAS_TRANSPOSE trans, FFLAS_DIAG diag, RandIter &Rand)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.279.1 Macro Definition Documentation

### 17.279.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.279.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.279.2 Function Documentation

### 17.279.2.1 check\_ffrmv()

```
bool check_ffrmv (
    const Field & F,
    size_t n,
    FFLAS_UPLO uplo,
    FFLAS_TRANSPOSE trans,
```

```

    FFLAS_DIAG diag,
    RandIter & Rand )

```

### 17.279.2.2 run\_with\_field()

```

bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )

```

### 17.279.2.3 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.280 test-ffrsm-check.C File Reference

```

#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"

```

## Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- int [main](#) (int argc, char \*\*argv)

## 17.280.1 Macro Definition Documentation

### 17.280.1.1 ENABLE\_ALL\_CHECKINGS

```

#define ENABLE_ALL_CHECKINGS 1

```

## 17.280.2 Function Documentation

### 17.280.2.1 main()

```

int main (
    int argc,
    char ** argv )

```

## 17.281 test-ffrsm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

### Macros

- #define [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- template<typename [Field](#) , class RandIter >  
bool [check\\_ffrsm](#) (const [Field](#) &F, size\_t m, size\_t n, const typename [Field::Element](#) &alpha, [FFLAS::FFLAS\\_SIDE](#) side, [FFLAS::FFLAS\\_UPLO](#) uplo, [FFLAS::FFLAS\\_TRANSPOSE](#) trans, [FFLAS::FFLAS\\_DIAG](#) diag, RandIter &Rand)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t m, size\_t n, [uint64\\_t](#) a, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.281.1 Macro Definition Documentation

### 17.281.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.281.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.281.2 Function Documentation

### 17.281.2.1 check\_ffrsm()

```
bool check_ffrsm (
    const Field & F,
    size_t m,
    size_t n,
    const typename Field::Element & alpha,
    FFLAS::FFLAS\_SIDE side,
    FFLAS::FFLAS\_UPLO uplo,
    FFLAS::FFLAS\_TRANSPOSE trans,
    FFLAS::FFLAS\_DIAG diag,
    RandIter & Rand )
```

### 17.281.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t n,
    uint64_t a,
    size_t iters,
    uint64_t seed )
```

### 17.281.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.282 test-ffrssyr2k.C File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

## Macros

- #define `ENABLE_ALL_CHECKINGS` 1

## Functions

- template<typename `Field` , class `RandIter` >  
bool `check_ffrssyr2k` (const `Field` &F, size\_t n, `FFLAS::FFLAS_UPLO` uplo, `FFLAS::FFLAS_DIAG` diagA, `RandIter` &Rand)
- template<class `Field` >  
bool `run_with_field` (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, `uint64_t` seed)
- int `main` (int argc, char \*\*argv)

## 17.282.1 Macro Definition Documentation

### 17.282.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.282.2 Function Documentation

### 17.282.2.1 check\_ftrssyr2k()

```
bool check_ftrssyr2k (
    const Field & F,
    size_t n,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_DIAG diagA,
    RandIter & Rand )
```

### 17.282.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.282.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.283 test-ftrstr.C File Reference

```
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <random>
```

## Macros

- #define `ENABLE_ALL_CHECKINGS` 1

## Functions

- template<typename `Field` , class `RandIter` >  
bool `check_ftrstr` (const `Field` &F, size\_t n, `FFLAS::FFLAS_SIDE` side, `FFLAS::FFLAS_UPLO` uplo, `FFLAS::FFLAS_DIAG` diagA, `FFLAS::FFLAS_DIAG` diagB, `RandIter` &Rand)
- template<class `Field` >  
bool `run_with_field` (`Givaro::Integer` q, size\_t b, size\_t n, size\_t iters, `uint64_t` seed)
- int `main` (int argc, char \*\*argv)



## 17.283.1 Macro Definition Documentation

### 17.283.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.283.2 Function Documentation

### 17.283.2.1 check\_fftrstr()

```
bool check_fftrstr (
    const Field & F,
    size_t n,
    FFLAS::FFLAS_SIDE side,
    FFLAS::FFLAS_UPLO uplo,
    FFLAS::FFLAS_DIAG diagA,
    FFLAS::FFLAS_DIAG diagB,
    RandIter & Rand )
```

### 17.283.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.283.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.284 test-fftrsv.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
```

## Macros

- #define [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)
- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

## Functions

- template<typename [Field](#) , class RandIter >  
bool [check\\_ftsrv](#) (const [Field](#) &F, size\_t n, [FFLAS\\_UPLO](#) uplo, [FFLAS\\_TRANSPOSE](#) trans, [FFLAS\\_DIAG](#) diag, RandIter &Rand)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.284.1 Macro Definition Documentation

### 17.284.1.1 [\\_\\_FFLASFFPACK\\_SEQUENTIAL](#)

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.284.1.2 [ENABLE\\_ALL\\_CHECKINGS](#)

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.284.2 Function Documentation

### 17.284.2.1 [check\\_ftsrv\(\)](#)

```
bool check_ftsrv (
    const Field & F,
    size_t n,
    FFLAS\_UPLO uplo,
    FFLAS\_TRANSPOSE trans,
    FFLAS\_DIAG diag,
    RandIter & Rand )
```

### 17.284.2.2 [run\\_with\\_field\(\)](#)

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64\_t seed )
```

### 17.284.2.3 [main\(\)](#)

```
int main (
    int argc,
    char ** argv )
```

## 17.285 test-fftrtri.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iomanip>
#include <iostream>
#include <random>
```

```
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

## Macros

- `#define __FFLASFFPACK_SEQUENTIAL`
- `#define ENABLE_ALL_CHECKINGS 1`

## Functions

- `template<typename Field, class RandIter >`  
`bool check_ftrtri (const Field &F, size_t n, FFLAS_UPLO uplo, FFLAS_DIAG diag, RandIter &Rand)`
- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64_t seed)`
- `int main (int argc, char **argv)`

## 17.285.1 Macro Definition Documentation

### 17.285.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.285.1.2 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.285.2 Function Documentation

### 17.285.2.1 check\_ftrtri()

```
bool check_ftrtri (
    const Field & F,
    size_t n,
    FFLAS_UPLO uplo,
    FFLAS_DIAG diag,
    RandIter & Rand )
```

### 17.285.2.2 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.285.2.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.286 test-interfaces-c.c File Reference

```
#include <fflas-ffpack/interfaces/libs/fflas_c.h>
#include <fflas-ffpack/interfaces/libs/ffpack_c.h>
#include <stdlib.h>
#include <stdio.h>
```

### Functions

- int [main](#) ()

### 17.286.1 Function Documentation

#### 17.286.1.1 main()

```
int main (
    void )
```

## 17.287 test-invert-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
```

### Macros

- #define [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.287.1 Macro Definition Documentation

#### 17.287.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

### 17.287.2 Function Documentation

### 17.287.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.288 test-io.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <random>
#include <givaro/modular.h>
#include <givaro/zring.h>
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/args-parser.h"
```

### Data Structures

- struct [CompactElement< Element >](#)
- struct [CompactElement< double >](#)
- struct [CompactElement< float >](#)
- struct [CompactElement< int64\\_t >](#)
- struct [CompactElement< int32\\_t >](#)
- struct [CompactElement< int16\\_t >](#)

### Functions

- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, [uint64\\_t](#) b, size\_t m, size\_t n, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

## 17.288.1 Function Documentation

### 17.288.1.1 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t iters,
    uint64_t seed )
```

### 17.288.1.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.289 test-lu.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-balanced.h>
#include <iostream>
#include <iomanip>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
#include <random>
```

### Macros

- #define `BASECASE_K` 37
- #define `__FFLASFFPACK_SEQUENTIAL`
- #define `__LUDIVINE_CUTOFF` 1

### Functions

- template<class `Field` , `FFLAS_DIAG` diag, `FFLAS_TRANSPOSE` trans>  
bool `test_LUdivine` (const `Field` &F, typename `Field::ConstElement_ptr` A, size\_t lda, size\_t r, size\_t m, size\_t n)  
*Tests the LUdivine routine.*
- template<class `Field` , `FFLAS_DIAG` diag>  
bool `verifPLUQ` (const `Field` &F, typename `Field::ConstElement_ptr` A, size\_t lda, typename `Field::Element_ptr` PLUQ, size\_t ldpluq, size\_t \*P, size\_t \*Q, size\_t m, size\_t n, size\_t R)  
*Verifies that  $B = PLUQ$  where A stores  $[L|U]$ .*
- template<class `Field` , `FFLAS_DIAG` diag, class `Randlter` >  
bool `test_pluq` (const `Field` &F, typename `Field::ConstElement_ptr` A, size\_t r, size\_t m, size\_t n, size\_t lda, `Randlter` &G)  
*Tests the LUdivine routine.*
- template<class `Field` , `FFLAS_DIAG` diag, `FFLAS_TRANSPOSE` trans, class `Randlter` >  
bool `launch_test` (const `Field` &F, size\_t r, size\_t m, size\_t n, `Randlter` &G)
- template<class `Field` >  
bool `run_with_field` (`Givaro::Integer` q, `uint64_t` b, size\_t m, size\_t n, size\_t r, size\_t iters, `uint64_t` seed)
- int `main` (int argc, char \*\*argv)

### Variables

- `Givaro::Timer` `tperm`
- `Givaro::Timer` `tgemm`
- `Givaro::Timer` `tBC`
- `Givaro::Timer` `ttrsm`
- `Givaro::Timer` `trest`
- `Givaro::Timer` `timtot`
- size\_t `mvcnt` = 0

## 17.289.1 Macro Definition Documentation

### 17.289.1.1 BASECASE\_K

```
#define BASECASE_K 37
```

### 17.289.1.2 \_\_FflasFfpack\_Sequential

```
#define __FflasFfpack_Sequential
```

### 17.289.1.3 \_\_Ludivine\_Cutoff

```
#define __Ludivine_Cutoff 1
```

## 17.289.2 Function Documentation

### 17.289.2.1 test\_LUdivine()

```
bool test_LUdivine (
    const Field & F,
    typename Field::ConstElement_ptr A,
    size_t lda,
    size_t r,
    size_t m,
    size_t n )
```

Tests the LUdivine routine.

#### Template Parameters

|              |                    |
|--------------|--------------------|
| <i>Field</i> | Field              |
| <i>Diag</i>  | Unit diagonal in U |
| <i>Trans</i> |                    |

#### Parameters

|            |                       |
|------------|-----------------------|
| <i>F</i>   | field                 |
| <i>A</i>   | Matrix (preallocated) |
| <i>r</i>   | rank of A             |
| <i>m</i>   | rows                  |
| <i>n</i>   | cols                  |
| <i>lda</i> | leading dim of A      |

#### Returns

0 iff correct, 1 otherwise

### 17.289.2.2 verifPLUQ()

```
bool verifPLUQ (
    const Field & F,
    typename Field::ConstElement_ptr A,
    size_t lda,
    typename Field::Element_ptr PLUQ,
    size_t ldpluq,
    size_t * P,
    size_t * Q,
    size_t m,
    size_t n,
    size_t R )
```

Verifies that  $B = PLUQ$  where  $A$  stores  $[L\backslash U]$ .

#### Template Parameters

|              |                    |
|--------------|--------------------|
| <i>Field</i> | Field              |
| <i>Diag</i>  | Unit diagonal in U |

#### Parameters

|            |                       |
|------------|-----------------------|
| <i>F</i>   | field                 |
| <i>A</i>   | Matrix (preallocated) |
| <i>r</i>   | rank of A             |
| <i>m</i>   | rows                  |
| <i>n</i>   | cols                  |
| <i>lda</i> | leading dim of A      |

#### Returns

0 iff correct, 1 otherwise

### 17.289.2.3 test\_pluq()

```
bool test_pluq (
    const Field & F,
    typename Field::ConstElement_ptr A,
    size_t r,
    size_t m,
    size_t n,
    size_t lda,
    RandIter & G )
```

Tests the LUdivine routine.

#### Template Parameters

|              |                    |
|--------------|--------------------|
| <i>Field</i> | Field              |
| <i>Diag</i>  | Unit diagonal in U |
| <i>Trans</i> |                    |

#### Parameters

|            |                       |
|------------|-----------------------|
| <i>F</i>   | field                 |
| <i>A</i>   | Matrix (preallocated) |
| <i>r</i>   | rank of A             |
| <i>m</i>   | rows                  |
| <i>n</i>   | cols                  |
| <i>lda</i> | leading dim of A      |

#### Returns

0 iff correct, 1 otherwise



#### 17.289.2.4 launch\_test()

```
bool launch_test (
    const Field & F,
    size_t r,
    size_t m,
    size_t n,
    RandIter & G )
```

#### 17.289.2.5 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
    uint64_t seed )
```

#### 17.289.2.6 main()

```
int main (
    int argc,
    char ** argv )
```

### 17.289.3 Variable Documentation

#### 17.289.3.1 tperm

Givaro::Timer tperm

#### 17.289.3.2 tgemm

Givaro::Timer tgemm

#### 17.289.3.3 tBC

Givaro::Timer tBC

#### 17.289.3.4 ttrsm

Givaro::Timer ttrsm

#### 17.289.3.5 trest

Givaro::Timer trest

**17.289.3.6 timtot**

```
Givaro::Timer timtot
```

**17.289.3.7 mvcnt**

```
size_t mvcnt = 0
```

**17.290 test-maxdelayeddim.C File Reference**

```
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include <stdlib.h>
#include <stdio.h>
```

**Macros**

- #define [MAX\\_WITH\\_SIZE\\_T](#)(x) ( (static\_cast<uint64\_t>(std::numeric\_limits<size\_t>::max()) < x)? std::numeric\_limits<size\_t>::max() : x )

**Functions**

- template<class [Field](#) >  
  bool [test](#) (Givaro::Integer p, size\_t kmax)
- int [main](#) ()

**17.290.1 Macro Definition Documentation****17.290.1.1 MAX\_WITH\_SIZE\_T**

```
#define MAX_WITH_SIZE_T(  
    x ) ( (static_cast<uint64_t>(std::numeric_limits<size_t>::max()) < x)? std::  
::numeric_limits<size_t>::max() : x )
```

**17.290.2 Function Documentation****17.290.2.1 test()**

```
bool test (  
    Givaro::Integer p,  
    size_t kmax )
```

**17.290.2.2 main()**

```
int main (  
    void )
```

## 17.291 test-minpoly.C File Reference

```
#include <iomanip>
#include <iostream>
#include <random>
#include <chrono>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/ffpack/args-parser.h"
#include "fflas-ffpack/ffpack/fflas_randommatrix.h"
#include "fflas-ffpack/ffpack/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <givaro/modular-integer.h>
#include <givaro/givpoly1factor.h>
#include <givaro/givpoly1.h>
```

### Functions

- `template<typename Field , class RandIter >`  
`bool check\_minpoly (const Field &F, size_t n, RandIter &G)`
- `template<class Field >`  
`bool run\_with\_field (Givaro::Integer q, size_t b, size_t n, size_t iters, uint64\_t seed)`
- `int main (int argc, char **argv)`

### 17.291.1 Function Documentation

#### 17.291.1.1 [check\\_minpoly\(\)](#)

```
bool check\_minpoly (
    const Field & F,
    size_t n,
    RandIter & G )
```

#### 17.291.1.2 [run\\_with\\_field\(\)](#)

```
bool run\_with\_field (
    Givaro::Integer q,
    size_t b,
    size_t n,
    size_t iters,
    uint64\_t seed )
```

#### 17.291.1.3 [main\(\)](#)

```
int main (
    int argc,
    char ** argv )
```

## 17.292 test-multifile1.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

## 17.293 test-multifile2.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

### Functions

- int [main](#) (void)

### 17.293.1 Function Documentation

#### 17.293.1.1 main()

```
int main (
    void )
```

## 17.294 test-nullspace.C File Reference

```
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/timer.h"
```

### Functions

- template<class [Field](#) >  
std::string [checkingMessage](#) (const [Field](#) &F)
- template<class [Field](#) >  
[Field::Element\\_ptr](#) [readOrRandomMatrixWithRankAndRandomRPM](#) (const [Field](#) &F, std::string file, size\_t m, size\_t n, size\_t lda, size\_t r, [uint64\\_t](#) seed)  
*If file is not empty, read it and set m, n, lda and r.*
- template<class [Field](#) >  
bool [test\\_nullspace](#) ([Field](#) &F, [FFLAS::FFLAS\\_SIDE](#) side, size\_t m, size\_t n, size\_t r, typename [Field::Element\\_ptr](#) A, size\_t lda)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, [uint64\\_t](#) b, size\_t m, size\_t n, size\_t r, size\_t iters, std::string file, [uint64\\_t](#) &seed)
- int [main](#) (int argc, char \*\*argv)

### 17.294.1 Function Documentation

**17.294.1.1 checkingMessage()**

```
std::string checkingMessage (
    const Field & F )
```

**17.294.1.2 readOrRandomMatrixWithRankAndRandomRPM()**

```
Field::Element_ptr readOrRandomMatrixWithRankAndRandomRPM (
    const Field & F,
    std::string file,
    size_t & m,
    size_t & n,
    size_t & lda,
    size_t & r,
    uint64_t seed )
```

If file is not empty, read it and set m, n, lda and r.

Otherwise, generate a random matrix of size m x n with random lda.

**17.294.1.3 test\_nullspace()**

```
bool test_nullspace (
    Field & F,
    FFLAS::FFLAS_SIDE side,
    size_t m,
    size_t n,
    size_t r,
    typename Field::Element_ptr A,
    size_t lda )
```

**17.294.1.4 run\_with\_field()**

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
    std::string file,
    uint64_t & seed )
```

**17.294.1.5 main()**

```
int main (
    int argc,
    char ** argv )
```

**17.295 test-permutations.C File Reference**

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <iostream>
#include <givaro/modular.h>
#include "fflas-ffpack/ffpack/ffpack.h"
```

## Functions

- bool `checkMonotonicApplyP` (`FFLAS_SIDE` Side, `FFLAS_TRANSPOSE` trans, size\_t \*P, size\_t N, size\_t R)
- int `main` ()

## Variables

- Givaro::Timer `tperm`
- Givaro::Timer `tgemm`
- Givaro::Timer `tBC`
- Givaro::Timer `ttrsm`
- Givaro::Timer `trest`
- Givaro::Timer `timtot`

## 17.295.1 Function Documentation

### 17.295.1.1 `checkMonotonicApplyP()`

```
bool checkMonotonicApplyP (
    FFLAS_SIDE Side,
    FFLAS_TRANSPOSE trans,
    size_t * P,
    size_t N,
    size_t R )
```

### 17.295.1.2 `main()`

```
int main (
    void )
```

## 17.295.2 Variable Documentation

### 17.295.2.1 `tperm`

```
Givaro::Timer tperm
```

### 17.295.2.2 `tgemm`

```
Givaro::Timer tgemm
```

### 17.295.2.3 `tBC`

```
Givaro::Timer tBC
```

### 17.295.2.4 `ttrsm`

```
Givaro::Timer ttrsm
```

### 17.295.2.5 trest

```
Givaro::Timer trest
```

### 17.295.2.6 timtot

```
Givaro::Timer timtot
```

## 17.296 test-pluq-check.C File Reference

```
#include <iostream>
#include <stdlib.h>
#include <time.h>
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
```

### Macros

- `#define` [ENABLE\\_ALL\\_CHECKINGS](#) 1

### Functions

- `int` [main](#) (`int` argc, `char` \*\*argv)

## 17.296.1 Macro Definition Documentation

### 17.296.1.1 ENABLE\_ALL\_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

## 17.296.2 Function Documentation

### 17.296.2.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.297 test-rankprofiles.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/ffpack/ffpack.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <iostream>
#include <iomanip>
#include <random>
#include <chrono>
```

## Macros

- `#define __FFLASFFPACK_SEQUENTIAL`

## Functions

- `template<class Field >`  
`bool run_with_field (Givaro::Integer q, uint64_t b, size_t m, size_t n, size_t r, size_t iters, uint64_t seed, bool par)`
- `int main (int argc, char **argv)`

### 17.297.1 Macro Definition Documentation

#### 17.297.1.1 \_\_FFLASFFPACK\_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

### 17.297.2 Function Documentation

#### 17.297.2.1 run\_with\_field()

```
bool run_with_field (
    Givaro::Integer q,
    uint64_t b,
    size_t m,
    size_t n,
    size_t r,
    size_t iters,
    uint64_t seed,
    bool par )
```

#### 17.297.2.2 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.298 test-rpm.C File Reference

```
#include <iostream>
#include "givaro/modular.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/ffpack/ffpack.h"
```

## Functions

- `bool checkRPM (size_t M, size_t N, size_t R)`
- `bool checkSymmetricRPM (size_t N, size_t R)`
- `int main (int argc, char **argv)`



## 17.298.1 Function Documentation

### 17.298.1.1 checkRPM()

```
bool checkRPM (
    size_t M,
    size_t N,
    size_t R )
```

### 17.298.1.2 checkSymmetricRPM()

```
bool checkSymmetricRPM (
    size_t N,
    size_t R )
```

### 17.298.1.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.299 test-simd.C File Reference

```
#include "givaro/givinteger.h"
#include "givaro/givprint.h"
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas/fflas_simd.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <array>
#include <vector>
#include <random>
#include <string>
#include <functional>
#include <limits>
#include <type_traits>
#include <algorithm>
```

## Data Structures

- struct [ScalFunctions](#)< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >
- struct [ScalFunctions](#)< Element, typename enable\_if< is\_integral< Element >::value >::type >

## Macros

- #define [REGISTER\\_TYPE\\_NAME](#)(type) template<> const char \*[TypeName](#)<type>(){return #type;}
- #define [TEST\\_ONE\\_OP](#)(name) btest &= [test\\_op](#)<simd> (simd::name, Scal::name, #name);

## Typedefs

- typedef Givaro::Integer [integer](#)

## Functions

- `template<typename... >`  
`const char * TypeName ()`
- `REGISTER\_TYPE\_NAME (float)`
- `REGISTER\_TYPE\_NAME (double)`
- `REGISTER\_TYPE\_NAME (int16_t)`
- `REGISTER\_TYPE\_NAME (int32_t)`
- `REGISTER\_TYPE\_NAME (int64_t)`
- `REGISTER\_TYPE\_NAME (uint16_t)`
- `REGISTER\_TYPE\_NAME (uint32_t)`
- `REGISTER\_TYPE\_NAME (uint64_t)`
- `template<class Element , class Alloc >`  
`enable_if< is_integral< Element >::value >::type generate\_random\_vector (vector< Element, Alloc > &a)`
- `template<class Element , class Alloc >`  
`enable_if< is_floating_point< Element >::value >::type generate\_random\_vector (vector< Element, Alloc > &a)`
- `template<class Element >`  
`enable_if< is_integral< Element >::value, bool >::type check\_eq (Element x, Element y)`
- `template<class Element >`  
`enable_if< is_floating_point< Element >::value, bool >::type check\_eq (Element x, Element y)`
- `template<class Ret , class T >`  
`Ret eval\_func\_on\_array (function< Ret()> f, array< T, 0 > arr)`
- `template<class Ret , class T , class... TArgs>`  
`Ret eval\_func\_on\_array (function< Ret(T, TArgs...)> f, array< typename remove_reference< T >::type, sizeof...(TArgs)+1 > &arr)`
- `template<class Simd , class RScal , class... AScal, class RSimd , class... ASimd>`  
`enable_if< sizeof...(AScal)==sizeof...(ASimd), bool >::type test\_op (RSimd(&FSimd)(ASimd...), RScal(&FScal)(AScal...), string fname)`
- `template<class simd , class Element >`  
`enable_if< is_floating_point< Element >::value, bool >::type test\_impl ()`
- `template<class simd , class Element >`  
`enable_if< is_integral< Element >::value, bool >::type test\_impl ()`
- `template<class Element >`  
`enable_if< is_integral< Element >::value, bool >::type test ()`
- `template<class Element >`  
`enable_if< is_floating_point< Element >::value, bool >::type test ()`
- `int main (int argc, char *argv[])`

## 17.299.1 Macro Definition Documentation

### 17.299.1.1 REGISTER\_TYPE\_NAME

```
#define REGISTER_TYPE_NAME(  
    type )    template<> const char *TypeName<type>(){return #type;}
```

### 17.299.1.2 TEST\_ONE\_OP

```
#define TEST_ONE_OP(  
    name )    btest &= test\_op<simd> (simd::name, Scal::name, #name);
```

## 17.299.2 Typedef Documentation

### 17.299.2.1 integer

```
typedef Givaro::Integer integer
```

## 17.299.3 Function Documentation

### 17.299.3.1 TypeName()

```
const char * TypeName ( )
```

### 17.299.3.2 REGISTER\_TYPE\_NAME() [1/8]

```
REGISTER_TYPE_NAME (
    float )
```

### 17.299.3.3 REGISTER\_TYPE\_NAME() [2/8]

```
REGISTER_TYPE_NAME (
    double )
```

### 17.299.3.4 REGISTER\_TYPE\_NAME() [3/8]

```
REGISTER_TYPE_NAME (
    int16_t )
```

### 17.299.3.5 REGISTER\_TYPE\_NAME() [4/8]

```
REGISTER_TYPE_NAME (
    int32_t )
```

### 17.299.3.6 REGISTER\_TYPE\_NAME() [5/8]

```
REGISTER_TYPE_NAME (
    int64_t )
```

### 17.299.3.7 REGISTER\_TYPE\_NAME() [6/8]

```
REGISTER_TYPE_NAME (
    uint16_t )
```

### 17.299.3.8 REGISTER\_TYPE\_NAME() [7/8]

```
REGISTER_TYPE_NAME (
    uint32_t )
```

### 17.299.3.9 REGISTER\_TYPE\_NAME() [8/8]

```
REGISTER_TYPE_NAME (
    uint64_t )
```

**17.299.3.10 generate\_random\_vector() [1/2]**

```
enable_if< is_integral< Element >::value >::type generate_random_vector (
    vector< Element, Alloc > & a )
```

**17.299.3.11 generate\_random\_vector() [2/2]**

```
enable_if< is_floating_point< Element >::value >::type generate_random_vector (
    vector< Element, Alloc > & a )
```

**17.299.3.12 check\_eq() [1/2]**

```
enable_if< is_integral< Element >::value, bool >::type check_eq (
    Element x,
    Element y )
```

**17.299.3.13 check\_eq() [2/2]**

```
enable_if< is_floating_point< Element >::value, bool >::type check_eq (
    Element x,
    Element y )
```

**17.299.3.14 eval\_func\_on\_array() [1/2]**

```
Ret eval_func_on_array (
    function< Ret()> f,
    array< T, 0 > arr )
```

**17.299.3.15 eval\_func\_on\_array() [2/2]**

```
Ret eval_func_on_array (
    function< Ret(T, TArgs...)> f,
    array< typename remove_reference< T >::type, sizeof...(TArgs)+1 > & arr )
```

**17.299.3.16 test\_op()**

```
enable_if< sizeof...(AScal)==sizeof...(ASimd), bool >::type test_op (
    RSimd(&)(ASimd...) FSimd,
    RScal(&)(AScal...) FScal,
    string fname )
```

**17.299.3.17 test\_impl() [1/2]**

```
enable_if< is_floating_point< Element >::value, bool >::type test_impl ( )
```

**17.299.3.18 test\_impl() [2/2]**

```
enable_if< is_integral< Element >::value, bool >::type test_impl ( )
```

**17.299.3.19 test() [1/2]**

```
enable_if< is_integral< Element >::value, bool >::type test ( )
```

**17.299.3.20 test() [2/2]**

```
enable_if< is_floating_point< Element >::value, bool >::type test ( )
```

**17.299.3.21 main()**

```
int main (
    int argc,
    char * argv[] )
```

**17.300 test-solve.C File Reference**

```
#include <givaro/modular-integer.h>
#include <iomanip>
#include <iostream>
#include "fflas-ffpack/utils/timer.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
```

**Functions**

- template<typename [Field](#) , class RandIter >  
bool [check\\_solve](#) (const [Field](#) &F, size\_t m, RandIter &Rand, bool isParallel)
- template<class [Field](#) >  
bool [run\\_with\\_field](#) (Givaro::Integer q, size\_t b, size\_t m, size\_t iters, [uint64\\_t](#) seed)
- int [main](#) (int argc, char \*\*argv)

**17.300.1 Function Documentation****17.300.1.1 check\_solve()**

```
bool check_solve (
    const Field & F,
    size_t m,
    RandIter & Rand,
    bool isParallel )
```

**17.300.1.2 run\_with\_field()**

```
bool run_with_field (
    Givaro::Integer q,
    size_t b,
    size_t m,
    size_t iters,
    uint64\_t seed )
```

### 17.300.1.3 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.301 101-fgemm.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.301.1 Function Documentation

#### 17.301.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.302 2x2-fgemm.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.302.1 Function Documentation

#### 17.302.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.303 2x2-fftrsv.C File Reference

```
#include <fflas-ffpack/fflas-ffpack-config.h>
#include <givaro/modular-balanced.h>
#include <fflas-ffpack/fflas/fflas.h>
#include <fflas-ffpack/utils/timer.h>
#include <fflas-ffpack/utils/fflas_io.h>
#include <fflas-ffpack/utils/args-parser.h>
#include <iostream>
#include <array>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

#### 17.303.1 Function Documentation

##### 17.303.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.304 2x2-pluq.C File Reference

```
#include <iostream>
#include <vector>
#include <givaro/modular.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/fflas-ffpack.h"
#include "fflas-ffpack/utils/fflas_io.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

#### 17.304.1 Function Documentation

##### 17.304.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.305 fflas-101\_1.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <iostream>
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.305.1 Function Documentation

#### 17.305.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

### 17.306 fflas-101\_3.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include <iostream>
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.306.1 Function Documentation

#### 17.306.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

### 17.307 fflas\_101.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <iostream>
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.307.1 Function Documentation



### 17.307.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.308 fflas\_101\_lvl1.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include <iostream>
#include "fflas-ffpack/utils/fflas_io.h"
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.308.1 Function Documentation

#### 17.308.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.309 ffpack-fgesv.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <fflas-ffpack/ffpack/ffpack.h>
#include <iostream>
```

### Functions

- int [main](#) (int argc, char \*\*argv)

### 17.309.1 Function Documentation

#### 17.309.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

## 17.310 ffpack-solve.C File Reference

```
#include <fflas-ffpack/fflas/fflas.h>
#include <givaro/modular.h>
```

```
#include <givaro/modular-balanced.h>
#include "fflas-ffpack/utils/fflas_io.h"
#include <fflas-ffpack/ffpack/ffpack.h>
#include <iostream>
```

## Functions

- int [main](#) (int argc, char \*\*argv)

### 17.310.1 Function Documentation

#### 17.310.1.1 main()

```
int main (
    int argc,
    char ** argv )
```

PS: the function Solve will modify the matrix A so here we used a duplicate matrix A2 otherwise  $A \cdot x$  will not be equal to b for the later verification stage

# Index

- [\\_F](#)
  - [RNSIntegerMod< RNS >, 554](#)
- [\\_M](#)
  - [rns\\_double, 532](#)
  - [rns\\_double\\_extended, 544](#)
- [\\_MAX\\_SIZE\\_MATRICES](#)
  - [benchmark-checkers.C, 779](#)
- [\\_MMi](#)
  - [rns\\_double, 532](#)
  - [rns\\_double\\_extended, 544](#)
- [\\_Mi](#)
  - [rns\\_double, 532](#)
  - [rns\\_double\\_extended, 544](#)
- [\\_Mi\\_modp\\_rns](#)
  - [RNSIntegerMod< RNS >, 554](#)
- [\\_NR\\_TESTS](#)
  - [benchmark-checkers.C, 779](#)
- [\\_PLUQ](#)
  - [FFPACK, 364](#)
- [\\_RNSdelayed](#)
  - [RNSIntegerMod< RNS >, 555](#)
- [\\_\\_FFLASFFPACK\\_ARITHPROG\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 827](#)
- [\\_\\_FFLASFFPACK\\_CACHE\\_LINE\\_SIZE](#)
  - [fflas\\_sparse.h, 886](#)
- [\\_\\_FFLASFFPACK\\_CHARPOLY\\_Danilevskii\\_LUKrylov\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 827](#)
- [\\_\\_FFLASFFPACK\\_CHARPOLY\\_LUKrylov\\_ArithProg\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 827](#)
- [\\_\\_FFLASFFPACK\\_CONFIGURATION](#)
  - [cblas.C, 1058](#)
  - [clapack.C, 1058](#)
  - [fblas.C, 1059](#)
  - [lapack.C, 1064](#)
- [\\_\\_FFLASFFPACK\\_DIMKPENALTY](#)
  - [fflas\\_pfgemm.inl, 872](#)
- [\\_\\_FFLASFFPACK\\_FORCE\\_SEQ](#)
  - [benchmark-charpoly-mp.C, 777](#)
- [\\_\\_FFLASFFPACK\\_FSYRK\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 827](#)
- [\\_\\_FFLASFFPACK\\_FSYTRF\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 827](#)
- [\\_\\_FFLASFFPACK\\_FTRSSYR2K\\_THRESHOLD](#)
  - [ffpack.h, 924](#)
- [\\_\\_FFLASFFPACK\\_FTRSTR\\_THRESHOLD](#)
  - [ffpack.h, 924](#)
- [\\_\\_FFLASFFPACK\\_FTRTRI\\_THRESHOLD](#)
  - [fflas-ffpack-default-thresholds.h, 827](#)
- [\\_\\_FFLASFFPACK\\_GAUSSJORDAN\\_BASECASE](#)
  - [ffpack\\_echelonforms.inl, 931](#)
  - [test-echelon.C, 1071](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_BLAS](#)
  - [fflas-ffpack/config.h, 808](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CBLAS](#)
  - [cblas.C, 1058](#)
  - [fflas-ffpack/config.h, 808](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CLAPACK](#)
  - [clapack.C, 1058](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_CXX11](#)
  - [fflas-ffpack/config.h, 809](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DGETRF](#)
  - [benchmark-dgetrf.C, 781](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DLFCN\\_H](#)
  - [fflas-ffpack/config.h, 809](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_DTRTRI](#)
  - [benchmark-dtrtri.C, 784](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_FLOAT\\_H](#)
  - [fflas-ffpack/config.h, 809](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_INT128](#)
  - [fflas-ffpack/config.h, 809](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_INTTYPES\\_H](#)
  - [fflas-ffpack/config.h, 809](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LAPACK](#)
  - [clapack.C, 1058](#)
  - [fflas-ffpack/config.h, 809](#)
  - [lapack.C, 1064](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LIMITS\\_H](#)
  - [fflas-ffpack/config.h, 809](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_LITTLE\\_ENDIAN](#)
  - [fflas-ffpack/config.h, 809](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_PTHREAD\\_H](#)
  - [fflas-ffpack/config.h, 809](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDDEF\\_H](#)
  - [fflas-ffpack/config.h, 809](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDINT\\_H](#)
  - [fflas-ffpack/config.h, 809](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDIO\\_H](#)
  - [fflas-ffpack/config.h, 809](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STDLIB\\_H](#)
  - [fflas-ffpack/config.h, 810](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STRINGS\\_H](#)
  - [fflas-ffpack/config.h, 810](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_STRING\\_H](#)
  - [fflas-ffpack/config.h, 810](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_STAT\\_H](#)
  - [fflas-ffpack/config.h, 810](#)
- [\\_\\_FFLASFFPACK\\_HAVE\\_SYS\\_TIME\\_H](#)
  - [fflas-ffpack/config.h, 810](#)

- \_\_FFLASFFPACK\_HAVE\_SYS\_TYPES\_H  
fflas-ffpack/config.h, [810](#)
- \_\_FFLASFFPACK\_HAVE\_UNISTD\_H  
fflas-ffpack/config.h, [810](#)
- \_\_FFLASFFPACK\_KAAPI\_ROUTINES\_INL  
kaapi\_routines.inl, [1039](#)
- \_\_FFLASFFPACK\_LT\_OBJDIR  
fflas-ffpack/config.h, [810](#)
- \_\_FFLASFFPACK\_MINBLOCKCUTS  
blockcuts.inl, [1038](#)
- \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET  
benchmark-charpoly.C, [778](#)  
benchmark-fadd-lvl2.C, [785](#)  
benchmark-fdot.C, [786](#)  
benchmark-fgemm-mp.C, [787](#)  
benchmark-fgemm-rns.C, [788](#)  
benchmark-fgemv-mp.C, [790](#)  
benchmark-fgemv.C, [792](#)  
benchmark-fgesv.C, [795](#)  
benchmark-fsyrc.C, [795](#)  
benchmark-fsytrf.C, [796](#)  
benchmark-ftrsm-mp.C, [797](#)  
benchmark-ftrsm.C, [798](#)  
benchmark-ftrsv.C, [798](#)  
benchmark-ftrtri.C, [799](#)  
benchmark-pluq.C, [802](#)
- \_\_FFLASFFPACK\_OPENBLAS\_NUM\_THREADS  
fflas-ffpack/config.h, [810](#)
- \_\_FFLASFFPACK\_PACKAGE  
fflas-ffpack/config.h, [810](#)
- \_\_FFLASFFPACK\_PACKAGE\_BUGREPORT  
fflas-ffpack/config.h, [810](#)
- \_\_FFLASFFPACK\_PACKAGE\_NAME  
fflas-ffpack/config.h, [810](#)
- \_\_FFLASFFPACK\_PACKAGE\_STRING  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_PACKAGE\_TARNAME  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_PACKAGE\_URL  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_PACKAGE\_VERSION  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_PLUQ\_THRESHOLD  
fflas-ffpack-default-thresholds.h, [827](#)  
test-echelon.C, [1071](#)
- \_\_FFLASFFPACK\_SEQPARTHRESHOLD  
fflas\_ptgemm.inl, [872](#)
- \_\_FFLASFFPACK\_SEQUENTIAL  
parallel.h, [1040](#)  
test-echelon.C, [1071](#)  
test-ftrmm.C, [1091](#)  
test-ftrmv.C, [1092](#)  
test-ftrsm.C, [1094](#)  
test-ftrsv.C, [1098](#)  
test-ftrtri.C, [1099](#)  
test-lu.C, [1102](#)  
test-rankprofiles.C, [1112](#)
- \_\_FFLASFFPACK\_SIZEOF\_CHAR  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_SIZEOF\_INT  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_SIZEOF\_LONG  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_SIZEOF\_LONG\_LONG  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_SIZEOF\_SHORT  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_SIZEOF\_\_INT64  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_STDC\_HEADERS  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_USE\_OPENMP  
fflas-ffpack/config.h, [811](#)
- \_\_FFLASFFPACK\_VERSION  
fflas-ffpack/config.h, [812](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD  
fflas-ffpack-default-thresholds.h, [826](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL  
fflas-ffpack-default-thresholds.h, [826](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL\_FLT  
fflas-ffpack-default-thresholds.h, [827](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_FLT  
fflas-ffpack-default-thresholds.h, [826](#)
- \_\_FFLASFFPACK\_charpoly\_INL  
ffpack\_charpoly.inl, [926](#)
- \_\_FFLASFFPACK\_checker\_charpoly\_INL  
checker\_charpoly.inl, [814](#)
- \_\_FFLASFFPACK\_checker\_det\_INL  
checker\_det.inl, [814](#)
- \_\_FFLASFFPACK\_checker\_fgemm\_INL  
checker\_fgemm.inl, [815](#)
- \_\_FFLASFFPACK\_checker\_ftrsm\_INL  
checker\_ftrsm.inl, [815](#)
- \_\_FFLASFFPACK\_checker\_invert\_INL  
checker\_invert.inl, [816](#)
- \_\_FFLASFFPACK\_checker\_pluq\_INL  
checker\_pluq.inl, [816](#)
- \_\_FFLASFFPACK\_fadd\_INL  
fflas\_fadd.inl, [833](#)
- \_\_FFLASFFPACK\_fassign\_INL  
fflas\_fassign.inl, [834](#)
- \_\_FFLASFFPACK\_faxpy\_INL  
fflas\_faxpy.inl, [834](#)
- \_\_FFLASFFPACK\_fdot\_INL  
fflas\_fdot.inl, [835](#)
- \_\_FFLASFFPACK\_fflas\_blockcuts\_INL  
blockcuts.inl, [1038](#)
- \_\_FFLASFFPACK\_fflas\_bounds\_INL  
fflas\_bounds.inl, [830](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_classical\_INL  
fgemm\_classical.inl, [838](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL  
fgemm\_winograd.inl, [841](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL  
fflas\_level1.inl, [866](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL

- fflas\_level2.inl, [869](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL
- fflas\_level3.inl, [871](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_mmhelper\_INL
- fflas\_helpers.inl, [860](#)
- \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL
- fflas\_sparse.inl, [889](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_INL
- simd128.inl, [875](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_double\_INL
- simd128\_double.inl, [876](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_float\_INL
- simd128\_float.inl, [876](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL
- simd128\_int16.inl, [876](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL
- simd128\_int32.inl, [877](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL
- simd128\_int64.inl, [877](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_INL
- simd256.inl, [878](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_double\_INL
- simd256\_double.inl, [878](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL
- simd256\_float.inl, [878](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL
- simd256\_int16.inl, [879](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL
- simd256\_int32.inl, [879](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int64\_INL
- simd256\_int64.inl, [880](#)
- \_\_FFLASFFPACK\_fflas\_freduce\_INL
- fflas\_freduce.inl, [852](#)
- \_\_FFLASFFPACK\_fflas\_freduce\_mp\_INL
- fflas\_freduce\_mp.inl, [852](#)
- \_\_FFLASFFPACK\_fflas\_fsyr2k\_INL
- fflas\_fsyr2k.inl, [856](#)
- \_\_FFLASFFPACK\_fflas\_fsyrrk\_INL
- fflas\_fsyrrk.inl, [857](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_INL
- igemm.inl, [861](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL
- igemm\_kernels.inl, [863](#)
- \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL
- igemm\_tools.inl, [864](#)
- \_\_FFLASFFPACK\_fflas\_pfgemm\_INL
- fflas\_pfgemm.inl, [872](#)
- \_\_FFLASFFPACK\_fflas\_pftsm\_INL
- fflas\_pftsm.inl, [873](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL
- csr\_hyb\_pspmm.inl, [897](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL
- csr\_hyb\_pspmv.inl, [898](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spm INL
- csr\_hyb\_spm.inl, [899](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL
- csr\_hyb\_spmv.inl, [899](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL
- csr\_hyb\_utils.inl, [900](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL
- csr\_pspmm.inl, [893](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL
- csr\_pspmv.inl, [894](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spm INL
- csr\_spm.inl, [895](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL
- csr\_spmv.inl, [896](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL
- ell\_pspmm.inl, [901](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL
- ell\_pspmv.inl, [902](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL
- ell\_simd\_pspmv.inl, [906](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL
- ell\_simd\_spmv.inl, [906](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL
- ell\_simd\_utils.inl, [907](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spm INL
- ell\_spm.inl, [903](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL
- ell\_spmv.inl, [904](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_utils\_INL
- ell\_utils.inl, [904](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL
- hyb\_zo\_pspmm.inl, [908](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL
- hyb\_zo\_pspmv.inl, [908](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spm INL
- hyb\_zo\_spm.inl, [909](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL
- hyb\_zo\_spmv.inl, [909](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_utils\_INL
- hyb\_zo\_utils.inl, [910](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spm INL
- coo\_spm.inl, [890](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL
- coo\_spmv.inl, [891](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL
- coo\_utils.inl, [891](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL
- sell\_pspmv.inl, [912](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL
- sell\_spmv.inl, [913](#)
- \_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL
- sell\_utils.inl, [914](#)
- \_\_FFLASFFPACK\_ffpack\_INL
- ffpack.inl, [925](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL
- ffpack\_charpoly\_danilevski.inl, [926](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL
- ffpack\_charpoly\_kgfast.inl, [927](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL
- ffpack\_charpoly\_kgfastgeneralized.inl, [928](#)
- \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL
- ffpack\_charpoly\_kglu.inl, [928](#)
- \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL

- ffpack\_echelonforms.inl, [931](#)
- \_\_FFLASFFPACK\_ffpack\_fgesv\_INL
  - ffpack\_fgesv.inl, [931](#)
- \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL
  - ffpack\_fgetrs.inl, [932](#)
- \_\_FFLASFFPACK\_ffpack\_fsytrf\_INL
  - ffpack\_fsytrf.inl, [934](#)
- \_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL
  - ffpack\_ftrssyr2k.inl, [934](#)
- \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL
  - ffpack\_ftrstr.inl, [935](#)
- \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL
  - ffpack\_ftrtr.inl, [936](#)
- \_\_FFLASFFPACK\_ffpack\_invert\_INL
  - ffpack\_invert.inl, [936](#)
- \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL
  - ffpack\_krylovelim.inl, [937](#)
- \_\_FFLASFFPACK\_ffpack\_ludivine\_INL
  - ffpack\_ludivine.inl, [937](#)
- \_\_FFLASFFPACK\_ffpack\_minpoly\_INL
  - ffpack\_minpoly.inl, [939](#)
- \_\_FFLASFFPACK\_ffpack\_permutation\_INL
  - ffpack\_permutation.inl, [941](#)
- \_\_FFLASFFPACK\_ffpack\_pluq\_INL
  - ffpack\_pluq.inl, [942](#)
- \_\_FFLASFFPACK\_ffpack\_ppluq\_INL
  - ffpack\_ppluq.inl, [943](#)
- \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL
  - ffpack\_rankprofiles.inl, [945](#)
- \_\_FFLASFFPACK\_ffgemm\_INL
  - fflas\_fgemm.inl, [838](#)
- \_\_FFLASFFPACK\_ffgemm\_bini\_INL
  - schedule\_bini.inl, [842](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_INL
  - schedule\_winograd.inl, [843](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_acc\_INL
  - schedule\_winograd\_acc.inl, [843](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_acc\_ip\_INL
  - schedule\_winograd\_acc\_ip.inl, [844](#)
- \_\_FFLASFFPACK\_ffgemm\_winograd\_ip\_INL
  - schedule\_winograd\_ip.inl, [845](#)
- \_\_FFLASFFPACK\_ffgemv\_INL
  - fflas\_ffgemv.inl, [846](#)
- \_\_FFLASFFPACK\_ffgemv\_mp\_INL
  - fflas\_ffgemv\_mp.inl, [847](#)
- \_\_FFLASFFPACK\_fger\_INL
  - fflas\_fger.inl, [848](#)
- \_\_FFLASFFPACK\_field\_rns\_INL
  - rns.inl, [951](#)
- \_\_FFLASFFPACK\_field\_rns\_double\_INL
  - rns-double.inl, [949](#)
- \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL
  - rns-double-recint.inl, [948](#)
- \_\_FFLASFFPACK\_freivalds\_INL
  - fflas\_freivalds.inl, [853](#)
- \_\_FFLASFFPACK\_fscal\_INL
  - fflas\_fscal.inl, [854](#)
- \_\_FFLASFFPACK\_fscal\_mp\_INL
  - fflas\_fscal\_mp.inl, [855](#)
- \_\_FFLASFFPACK\_ftrmm\_INL
  - fflas\_ftrmm.inl, [857](#)
- \_\_FFLASFFPACK\_ftrsm\_INL
  - fflas\_ftrsm.inl, [858](#)
- \_\_FFLASFFPACK\_ftrsv\_INL
  - fflas\_ftrsv.inl, [859](#)
- \_\_FFLASFFPACK\_simd512\_INL
  - simd512.inl, [880](#)
- \_\_FFLASFFPACK\_simd512\_double\_INL
  - simd512\_double.inl, [881](#)
- \_\_FFLASFFPACK\_simd512\_float\_INL
  - simd512\_float.inl, [881](#)
- \_\_FFLASFFPACK\_simd512\_int32\_INL
  - simd512\_int32.inl, [881](#)
- \_\_FFLAS\_L1\_INST\_C
  - fflas\_L1\_inst.C, [965](#)
- \_\_FFLAS\_L2\_INST\_C
  - fflas\_L2\_inst.C, [968](#)
- \_\_FFLAS\_L3\_INST\_C
  - fflas\_L3\_inst.C, [973](#)
- \_\_FFLAS\_TRSM\_READONLY
  - fflas\_L3\_inst\_implem.inl, [975](#)
  - fflas\_level3.inl, [871](#)
  - ffpack\_ppluq.inl, [943](#)
- \_\_FFPACK\_FSYTRF\_BC\_CROUT
  - benchmark-fsytrf.C, [796](#)
- \_\_FFPACK\_INST\_C
  - ffpack\_inst.C, [1031](#)
- \_\_FFPACK\_charpoly\_mp\_INL
  - ffpack\_charpoly\_mp.inl, [929](#)
- \_\_FFPACK\_det\_mp\_INL
  - ffpack\_det\_mp.inl, [929](#)
- \_\_FFPACK\_ffgemm\_classical\_INL
  - ffgemm\_classical\_mp.inl, [840](#)
- \_\_FFPACK\_fger\_mp\_INL
  - fflas\_fger\_mp.inl, [849](#)
- \_\_FFPACK\_ftrsm\_mp\_INL
  - fflas\_ftrsm\_mp.inl, [859](#)
- \_\_FFPACK\_ludivine\_mp\_INL
  - ffpack\_ludivine\_mp.inl, [938](#)
- \_\_FFPACK\_pluq\_mp\_INL
  - ffpack\_pluq\_mp.inl, [943](#)
- \_\_LUDIVINE\_CUTOFF
  - test-lu.C, [1103](#)
- \_\_has\_builtin
  - bit\_manipulation.h, [1049](#)
- \_alloc
  - rns\_double\_elt, [534](#)
  - rns\_double\_elt\_cstptr, [537](#)
  - rns\_double\_elt\_ptr, [540](#)
- \_basis
  - rns\_double, [531](#)
  - rns\_double\_extended, [543](#)
- \_basisMax
  - rns\_double, [531](#)
  - rns\_double\_extended, [543](#)
- \_coo

- SpMat< Field, flag >, 758
- \_coo16
  - CooMat< Field >, 433
- \_coo16\_zo
  - CooMat< Field >, 433
- \_coo32
  - CooMat< Field >, 433
- \_coo32\_zo
  - CooMat< Field >, 433
- \_coo64
  - CooMat< Field >, 433
- \_coo64\_zo
  - CooMat< Field >, 433
- \_crt\_in
  - rns\_double, 532
  - rns\_double\_extended, 544
- \_crt\_out
  - rns\_double, 532
  - rns\_double\_extended, 544
- \_csr
  - SpMat< Field, flag >, 758
- \_csr16
  - CsrMat< Field >, 434
- \_csr16\_zo
  - CsrMat< Field >, 434
- \_csr32
  - CsrMat< Field >, 434
- \_csr32\_zo
  - CsrMat< Field >, 434
- \_csr64
  - CsrMat< Field >, 434
- \_csr64\_zo
  - CsrMat< Field >, 434
- \_ell
  - SpMat< Field, flag >, 758
- \_ell16
  - EllMat< Field >, 441
- \_ell16\_zo
  - EllMat< Field >, 441
- \_ell32
  - EllMat< Field >, 441
- \_ell32\_zo
  - EllMat< Field >, 441
- \_ell64
  - EllMat< Field >, 441
- \_ell64\_zo
  - EllMat< Field >, 441
- \_errorStream
  - Failure, 443
- \_field\_rns
  - rns\_double, 531
  - rns\_double\_extended, 544
- \_iM\_modp\_rns
  - RNSIntegerMod< RNS >, 554
- \_ibeg
  - ForStrategy2D< blocksize\_t, Cut, Param >, 463
- \_iend
  - ForStrategy2D< blocksize\_t, Cut, Param >, 463
- \_invbasis
  - rns\_double, 531
  - rns\_double\_extended, 543
- \_jbeg
  - ForStrategy2D< blocksize\_t, Cut, Param >, 463
- \_jend
  - ForStrategy2D< blocksize\_t, Cut, Param >, 463
- \_ldm
  - rns\_double, 532
  - rns\_double\_extended, 544
- \_mi\_sum
  - rns\_double, 532
- \_negbasis
  - rns\_double, 531
  - rns\_double\_extended, 543
- \_p
  - RNSIntegerMod< RNS >, 554
- \_pbits
  - rns\_double, 532
  - rns\_double\_extended, 544
- \_ptr
  - rns\_double\_elt, 534
  - rns\_double\_elt\_cstptr, 537
  - rns\_double\_elt\_ptr, 540
- \_rns
  - RNSInteger< RNS >, 548
  - RNSIntegerMod< RNS >, 554
- \_simd512\_int64\_INL
  - simd512\_int64.inl, 882
- \_size
  - rns\_double, 532
  - rns\_double\_extended, 544
- \_stride
  - rns\_double\_elt, 534
  - rns\_double\_elt\_cstptr, 537
  - rns\_double\_elt\_ptr, 540
- ~CheckerImplem\_Det
  - CheckerImplem\_Det< Field >, 413
- ~CheckerImplem\_PLUQ
  - CheckerImplem\_PLUQ< Field >, 417
- ~CheckerImplem\_charpoly
  - CheckerImplem\_charpoly< Field, Polynomial >, 412
- ~CheckerImplem\_fgemm
  - CheckerImplem\_fgemm< Field >, 414
- ~CheckerImplem\_ftsm
  - CheckerImplem\_ftsm< Field >, 415
- ~CheckerImplem\_invert
  - CheckerImplem\_invert< Field >, 416
- ~rns\_double\_elt
  - rns\_double\_elt, 533
- 101-fgemm.C, 1118
  - main, 1118
- 2x2-fgemm.C, 1118
  - main, 1118
- 2x2-ftsm.C, 1119
  - main, 1119
- 2x2-pluq.C, 1119
  - main, 1119

- main, 1119
- add
  - FFLAS::vectorised, 289
  - FieldSimd< \_Field >, 446
  - RNSIntegerMod< RNS >, 552
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 558
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 562
  - Simd128\_impl< true, true, false, 2 >, 574
  - Simd128\_impl< true, true, false, 4 >, 584
  - Simd128\_impl< true, true, false, 8 >, 594
  - Simd128\_impl< true, true, true, 2 >, 601
  - Simd128\_impl< true, true, true, 4 >, 610
  - Simd128\_impl< true, true, true, 8 >, 618
  - Simd256\_impl< true, false, true, 8 >, 629
  - Simd256\_impl< true, true, false, 2 >, 641
  - Simd256\_impl< true, true, false, 4 >, 656
  - Simd256\_impl< true, true, false, 8 >, 668
  - Simd256\_impl< true, true, true, 2 >, 675
  - Simd256\_impl< true, true, true, 4 >, 685, 691
  - Simd256\_impl< true, true, true, 8 >, 700
  - Simd512\_impl< true, false, true, 8 >, 710
  - Simd512\_impl< true, true, false, 8 >, 720
  - Simd512\_impl< true, true, true, 8 >, 729
- add\_r
  - FieldSimd< \_Field >, 446
- addin
  - FieldSimd< \_Field >, 446
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 558
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 562
  - Simd128\_impl< true, true, false, 2 >, 574
  - Simd128\_impl< true, true, false, 4 >, 584
  - Simd128\_impl< true, true, false, 8 >, 594
  - Simd128\_impl< true, true, true, 2 >, 601
  - Simd128\_impl< true, true, true, 4 >, 610
  - Simd128\_impl< true, true, true, 8 >, 619
  - Simd256\_impl< true, false, true, 8 >, 629
  - Simd256\_impl< true, true, false, 2 >, 641
  - Simd256\_impl< true, true, false, 4 >, 656
  - Simd256\_impl< true, true, false, 8 >, 668
  - Simd256\_impl< true, true, true, 2 >, 675
  - Simd256\_impl< true, true, true, 4 >, 685, 691
  - Simd256\_impl< true, true, true, 8 >, 700
  - Simd512\_impl< true, false, true, 8 >, 710
  - Simd512\_impl< true, true, false, 8 >, 721
  - Simd512\_impl< true, true, true, 8 >, 729
- addin\_r
  - FieldSimd< \_Field >, 447
- addp
  - FFLAS::vectorised, 288
- AlgoChooser< ModeCategories::ConvertTo< Element-  
Categories::RNSElementTag >, ParSeq >, 405
  - value, 405
- AlgoChooser< ModeT, ParSeq >, 405
  - value, 405
- align-allocator.h, 1047
- alignable
  - FFLAS, 196
- alignable< Givaro::Integer \* >
  - FFLAS, 196
- alignment
  - FieldSimd< \_Field >, 450
  - Simd128\_impl< true, true, false, 2 >, 577
  - Simd128\_impl< true, true, false, 4 >, 587
  - Simd128\_impl< true, true, false, 8 >, 597
  - Simd128\_impl< true, true, true, 2 >, 606
  - Simd128\_impl< true, true, true, 4 >, 614
  - Simd128\_impl< true, true, true, 8 >, 623
  - Simd256\_impl< true, false, true, 8 >, 633
  - Simd256\_impl< true, true, false, 2 >, 643
  - Simd256\_impl< true, true, false, 4 >, 660
  - Simd256\_impl< true, true, false, 8 >, 670
  - Simd256\_impl< true, true, true, 2 >, 679
  - Simd256\_impl< true, true, true, 4 >, 695
  - Simd256\_impl< true, true, true, 8 >, 704
  - Simd512\_impl< true, false, true, 8 >, 712
  - Simd512\_impl< true, true, false, 8 >, 723
  - Simd512\_impl< true, true, true, 8 >, 733
- Amax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
Trait >, 502
- Amin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
Trait >, 502
- applyP
  - FFPACK, 310, 311, 367
- applyP\_block
  - FFPACK, 358
- applyP\_modular\_double
  - ffpack.C, 993
  - ffpack\_c.h, 1017
- ArbitraryPreclntTag, 405
- areEqual
  - RNSIntegerMod< RNS >, 553
- AreEqual< X, X >, 406
  - value, 406
- AreEqual< X, Y >, 406
  - value, 406
- args-parser.h, 1047
  - ArgumentType, 1048
  - END\_OF\_ARGUMENTS, 1048
  - findArgument, 1049
  - getListArgs, 1049
  - printHelpMessage, 1048
  - TYPE\_BOOL, 1048
  - TYPE\_DOUBLE, 1048
  - TYPE\_INT, 1048
  - TYPE\_INTEGER, 1048
  - type\_integer, 1048
  - TYPE\_INTLIST, 1048



- TYPE\_LONGLONG, 1048
- TYPE\_NONE, 1048
- TYPE\_STR, 1048
- TYPE\_UINT64, 1048
- Argument, 406
  - c, 407
  - data, 407
  - example, 407
  - helpString, 407
  - type, 407
- ArgumentType
  - args-parser.h, 1048
- ArithProg
  - FFPACK::Protected, 399
- arithprog.C, 769
  - CUBE, 769
  - GFOPS, 769
  - main, 770
  - TTimer, 770
- assign
  - RNSInteger< RNS >, 548
  - RNSIntegerMod< RNS >, 552
- associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >, 407
  - field, 408
  - type, 408
- associatedDelayedField< const Givaro::Modular< T, X > >, 408
  - field, 408
  - type, 408
- associatedDelayedField< const Givaro::ModularBalanced< T > >, 408
  - field, 409
  - type, 409
- associatedDelayedField< const Givaro::ZRing< T > >, 409
  - field, 409
  - type, 409
- associatedDelayedField< Field >, 407
  - field, 407
  - type, 407
- assume\_aligned
  - fflas\_sparse.h, 886
- AtlasConj
  - config-blas.h, 820
- Aunfit
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 501
- aut
  - HelperFlag, 472
- Auto, 409
- autotune/charpoly.C
  - CUBE, 770
  - GFOPS, 770
  - main, 771
  - TTimer, 771
- autotune/pluq.C
  - CUBE, 775
- GFOPS, 775
  - main, 775
  - TTimer, 775
- averageCol
  - StatsMatrix, 760
- averageColDifference
  - StatsMatrix, 760
- averageRow
  - StatsMatrix, 760
- averageRowDifference
  - StatsMatrix, 761
- axpy
  - FieldSimd< \_Field >, 448, 449
- axpy\_r
  - FieldSimd< \_Field >, 449
- axpyin
  - FieldSimd< \_Field >, 449
  - RNSIntegerMod< RNS >, 553
- axpyin\_r
  - FieldSimd< \_Field >, 449
- balanced
  - FieldTraits< FFPACK::RNSInteger< T > >, 451
  - FieldTraits< FFPACK::RNSIntegerMod< T > >, 452
  - FieldTraits< Field >, 450
  - FieldTraits< Givaro::Modular< Element > >, 452
  - FieldTraits< Givaro::ModularBalanced< Element > >, 453
  - FieldTraits< Givaro::ZRing< double > >, 453
  - FieldTraits< Givaro::ZRing< float > >, 454
  - FieldTraits< Givaro::ZRing< Givaro::Integer > >, 454
  - FieldTraits< Givaro::ZRing< int16\_t > >, 455
  - FieldTraits< Givaro::ZRing< int32\_t > >, 455
  - FieldTraits< Givaro::ZRing< int64\_t > >, 456
  - FieldTraits< Givaro::ZRing< Reclnt::ruint< K > >, 457
  - FieldTraits< Givaro::ZRing< uint16\_t > >, 457
  - FieldTraits< Givaro::ZRing< uint32\_t > >, 458
  - FieldTraits< Givaro::ZRing< uint64\_t > >, 458
  - winograd.C, 777
- BARRIER
  - parallel.h, 1040
- BASECASE\_K
  - test-lu.C, 1102
- BaseTimer
  - FFLAS, 80
- BasisElement
  - rns\_double, 528
  - rns\_double\_extended, 541
  - RNSInteger< RNS >, 546
  - RNSIntegerMod< RNS >, 550
- begin
  - ForStrategy1D< blocksize\_t, Cut, Param >, 460
  - Info, 477, 478
- BEGIN\_PARALLEL\_MAIN
  - parallel.h, 1041
- benchmark-charpoly-mp.C, 777

- \_\_FFLASFFPACK\_FORCE\_SEQ, 777
  - main, 777
- benchmark-charpoly.C, 778
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 778
  - main, 778
  - run\_with\_field, 778
- benchmark-checkers.C, 779
  - \_MAX\_SIZE\_MATRICES, 779
  - \_NR\_TESTS, 779
  - CUBE, 779
  - ENABLE\_ALL\_CHECKINGS, 779
  - main, 779
- benchmark-dgemm.C, 780
  - CBLAS\_GEMM, 780
  - Floats, 780
  - main, 780
  - TTimer, 780
- benchmark-dgetrf.C, 781
  - \_\_FFLASFFPACK\_HAVE\_DGETRF, 781
  - main, 781
  - TTimer, 781
- benchmark-dgetri.C, 781
  - main, 782
  - TTimer, 782
- benchmark-dsytrf.C, 782
  - EFFGFF, 782
  - main, 783
  - TTimer, 783
- benchmark-dtrsm.C, 783
  - main, 783
  - TTimer, 783
- benchmark-dtrtri.C, 784
  - \_\_FFLASFFPACK\_HAVE\_DTRTRI, 784
  - main, 784
  - TTimer, 784
- benchmark-fadd-lvl2.C, 785
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 785
  - main, 785
- benchmark-fdot.C, 785
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 786
  - main, 786
  - run\_with\_field, 786
- benchmark-fgemm-mp.C, 786
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 787
  - main, 787
  - MG\_DEFAULT, 787
  - STD\_RECINT\_SIZE, 787
  - tmain, 787
- benchmark-fgemm-rns.C, 787
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 788
  - ConstElement\_ptr, 788
  - Element\_ptr, 788
  - Field, 788
- GRAIN, 788
  - main, 789
- PSeq, 789
- RNS, 788
- THREADS, 788
- THREED, 789
- THREEDA, 789
- THREEDIP, 789
- TWOD, 788
- TWODA, 788
- benchmark-fgemm.C, 789
  - CLASSIC\_HYBRID, 789
  - main, 790
- benchmark-fgemv-mp.C, 790
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 790
  - main, 791
  - MG\_DEFAULT, 790
  - STD\_RECINT\_SIZE, 790
  - tmain, 791
  - write\_matrix, 791
- benchmark-fgemv.C, 791
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 792
  - benchmark\_disp, 793
  - benchmark\_in\_Field, 793
  - benchmark\_with\_field, 794
  - benchmark\_with\_timer, 793
  - check\_result, 793
  - fill\_value, 792
  - genData, 792
  - main, 794
- benchmark-fgesv.C, 794
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 795
  - main, 795
- benchmark-fsyrk.C, 795
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 795
  - CUBE, 796
  - main, 796
- benchmark-fsytrf.C, 796
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 796
  - \_\_FFPACK\_FSYTRF\_BC\_CROUT, 796
  - CUBE, 796
  - main, 797
- benchmark-ftsm-mp.C, 797
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 797
  - main, 797
- benchmark-ftsm.C, 797
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 798
  - main, 798
- benchmark-ftsv.C, 798
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, 798

- main, [798](#)
- benchmark-ftrtri.C, [799](#)
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, [799](#)
  - CUBE, [799](#)
  - main, [799](#)
- benchmark-inverse.C, [799](#)
  - CUBE, [800](#)
  - main, [800](#)
- benchmark-lqup-mp.C, [800](#)
  - main, [800](#)
- benchmark-lqup.C, [801](#)
  - CUBE, [801](#)
  - main, [801](#)
- benchmark-pluq.C, [801](#)
  - \_\_FFLASFFPACK\_OPENBLAS\_NT\_ALREADY\_SET, [802](#)
  - CUBE, [802](#)
  - Field, [802](#)
  - main, [802](#)
  - Rec\_Initialize, [802](#)
  - verification\_PLUQ, [802](#)
- benchmark-wino.C, [803](#)
  - CUBE, [803](#)
  - launch\_wino, [803](#)
  - main, [803](#)
- benchmark\_disp
  - benchmark-fgemv.C, [793](#)
- benchmark\_in\_Field
  - benchmark-fgemv.C, [793](#)
- benchmark\_with\_field
  - benchmark-fgemv.C, [794](#)
- benchmark\_with\_timer
  - benchmark-fgemv.C, [793](#)
- Bini, [409](#)
  - FFLAS::BLAS3, [200](#)
- bit\_manipulation.h, [1049](#)
  - \_\_has\_builtin, [1049](#)
  - clz, [1050](#)
  - ctz, [1050](#)
- bitsize
  - FFLAS, [145](#)
- bitsize< Givaro::ZRing< Givaro::Integer > >
  - FFLAS, [146](#)
- blas\_enum
  - config-blas.h, [820](#)
- blend
  - Simd128\_impl< true, true, false, 2 >, [574](#)
  - Simd128\_impl< true, true, false, 4 >, [584](#)
  - Simd128\_impl< true, true, false, 8 >, [594](#)
  - Simd128\_impl< true, true, true, 2 >, [601](#)
  - Simd128\_impl< true, true, true, 4 >, [610](#)
  - Simd128\_impl< true, true, true, 8 >, [618](#)
  - Simd256\_impl< true, false, true, 8 >, [629](#)
  - Simd256\_impl< true, true, false, 4 >, [655](#)
  - Simd256\_impl< true, true, false, 8 >, [668](#)
  - Simd256\_impl< true, true, true, 4 >, [685](#)
  - Simd256\_impl< true, true, true, 8 >, [700](#)
  - Simd512\_impl< true, false, true, 8 >, [709](#)
  - Simd512\_impl< true, true, false, 8 >, [720](#)
  - Simd512\_impl< true, true, true, 8 >, [728](#)
- blend\_twice
  - Simd256\_impl< true, true, false, 2 >, [640](#)
  - Simd256\_impl< true, true, true, 2 >, [675](#)
- blendv
  - Simd256\_impl< true, false, true, 8 >, [629](#)
  - Simd512\_impl< true, false, true, 8 >, [709](#)
- Block, [409](#)
- BlockCuts
  - FFLAS, [187](#), [189](#)
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >
  - FFLAS, [189](#)
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >
  - FFLAS, [188](#)
- BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >
  - FFLAS, [189](#)
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >
  - FFLAS, [188](#)
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >
  - FFLAS, [188](#)
- BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >
  - FFLAS, [189](#)
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >
  - FFLAS, [188](#)
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >
  - FFLAS, [188](#)
- BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >
  - FFLAS, [189](#)
- BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >
  - FFLAS, [187](#)
- blockcuts.inl, [1037](#)
  - \_\_FFLASFFPACK\_MINBLOCKCUTS, [1038](#)
  - \_\_FFLASFFPACK\_fflas\_blockcuts\_INL, [1038](#)
- blockindex
  - ForStrategy1D< blocksize\_t, Cut, Param >, [460](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [463](#)
- BlockingFactor
  - FFLAS::details, [214](#)
- BLOCKS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [464](#)
- Bmax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [502](#)
- Bmin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [502](#)

- build
  - ForStrategy1D< blocksize\_t, Cut, Param >, [459](#)
- buildMatrix
  - FFPACK, [351](#)
- Bunfit
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [501](#)
- c
  - Argument, [407](#)
- callLUdivine\_small< double >, [410](#)
  - operator(), [410](#)
- callLUdivine\_small< Element >, [410](#)
  - operator(), [410](#)
- callLUdivine\_small< float >, [411](#)
  - operator(), [411](#)
- cardinality
  - RNSInteger< RNS >, [547](#)
  - RNSIntegerMod< RNS >, [551](#)
- cast.h, [1050](#)
- category
  - FieldTraits< FFPACK::RNSInteger< T > >, [451](#)
  - FieldTraits< FFPACK::RNSIntegerMod< T > >, [451](#)
  - FieldTraits< Field >, [450](#)
  - FieldTraits< Givaro::Modular< Element > >, [452](#)
  - FieldTraits< Givaro::ModularBalanced< Element > >, [453](#)
  - FieldTraits< Givaro::ZRing< double > >, [453](#)
  - FieldTraits< Givaro::ZRing< float > >, [454](#)
  - FieldTraits< Givaro::ZRing< Givaro::Integer > >, [454](#)
  - FieldTraits< Givaro::ZRing< int16\_t > >, [455](#)
  - FieldTraits< Givaro::ZRing< int32\_t > >, [455](#)
  - FieldTraits< Givaro::ZRing< int64\_t > >, [456](#)
  - FieldTraits< Givaro::ZRing< Reclnt::ruint< K > >, [456](#)
  - FieldTraits< Givaro::ZRing< uint16\_t > >, [457](#)
  - FieldTraits< Givaro::ZRing< uint32\_t > >, [457](#)
  - FieldTraits< Givaro::ZRing< uint64\_t > >, [458](#)
- cblas.C, [1057](#)
  - \_\_FFLASFFPACK\_CONFIGURATION, [1058](#)
  - \_\_FFLASFFPACK\_HAVE\_CBLAS, [1058](#)
  - main, [1058](#)
- CBLAS\_DIAG
  - config-blas.h, [820](#)
- CBLAS\_ENUM\_DEFINED\_H
  - config-blas.h, [820](#)
- CBLAS\_EXTERNALS
  - config-blas.h, [820](#)
- CBLAS\_GEMM
  - benchmark-dgemm.C, [780](#)
- cblas\_imptrsm
  - FFLAS, [133](#)
- CBLAS\_INT
  - config-blas.h, [819](#)
- CBLAS\_ORDER
  - config-blas.h, [820](#)
- CBLAS\_SIDE
  - config-blas.h, [821](#)
- CBLAS\_TRANSPOSE
  - config-blas.h, [820](#)
- CBLAS\_UPLO
  - config-blas.h, [820](#)
- CblasColMajor
  - config-blas.h, [820](#)
- CblasConjTrans
  - config-blas.h, [820](#)
- CblasLeft
  - config-blas.h, [821](#)
- CblasLower
  - config-blas.h, [820](#)
- CblasNonUnit
  - config-blas.h, [821](#)
- CblasNoTrans
  - config-blas.h, [820](#)
- CblasRight
  - config-blas.h, [821](#)
- CblasRowMajor
  - config-blas.h, [820](#)
- CblasTrans
  - config-blas.h, [820](#)
- CblasUnit
  - config-blas.h, [821](#)
- CblasUpper
  - config-blas.h, [820](#)
- ceil
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [558](#)
  - Simd256\_impl< true, false, true, 8 >, [632](#)
  - Simd512\_impl< true, false, true, 8 >, [712](#)
- changeBS
  - ForStrategy1D< blocksize\_t, Cut, Param >, [461](#)
- changeCBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [464](#)
- changeRBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [464](#)
- characteristic
  - RNSInteger< RNS >, [547](#)
  - RNSIntegerMod< RNS >, [551](#)
- CharPoly
  - FFPACK, [329](#), [330](#), [352](#), [372](#)
- charpoly.C, [770](#), [771](#)
- CharpolyFailed, [411](#)
- check
  - Checker\_Empty< Field >, [412](#)
  - CheckerImplem\_charpoly< Field, Polynomial >, [412](#)
  - CheckerImplem\_Det< Field >, [413](#)
  - CheckerImplem\_fgemm< Field >, [414](#)
  - CheckerImplem\_ftsm< Field >, [416](#)
  - CheckerImplem\_invert< Field >, [417](#)
  - CheckerImplem\_PLUQ< Field >, [417](#)
- check1
  - regression-check.C, [1065](#)
- check2

- regression-check.C, 1065
- check3
  - regression-check.C, 1065
- check4
  - regression-check.C, 1065
- CHECK\_DEPENDENCIES
  - parallel.h, 1040
- check\_eq
  - test-simd.C, 1116
- check\_fdot
  - test-fdot.C, 1074
- check\_fger
  - test-fger.C, 1081
- check\_fsyr2k
  - test-fsyr2k.C, 1087
- check\_fsyrk
  - test-fsyrk.C, 1088
- check\_fsyrk\_bkdiag
  - test-fsyrk.C, 1088
- check\_fsyrk\_diag
  - test-fsyrk.C, 1088
- check\_ftrmm
  - test-ftrmm.C, 1091
- check\_ftrmv
  - test-ftrmv.C, 1092
- check\_ftrsm
  - test-ftrsm.C, 1094
- check\_ftrssyr2k
  - test-ftrssyr2k.C, 1096
- check\_ftrstr
  - test-ftrstr.C, 1097
- check\_ftrsv
  - test-ftrsv.C, 1098
- check\_ftrtri
  - test-ftrtri.C, 1099
- check\_minpoly
  - test-minpoly.C, 1107
- check\_MM
  - test-fgemm.C, 1077
- check\_MV
  - test-fgemv.C, 1079
- check\_result
  - benchmark-fgemv.C, 793
- check\_solve
  - test-solve.C, 1117
- checkA
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 501
- checkB
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 501
- CHECKER, 49
- Checker\_charpoly
  - FFPACK, 309
- checker\_charpoly.inl, 813
  - \_\_FFLASFFPACK\_checker\_charpoly\_INL, 814
- Checker\_Det
  - FFPACK, 309
- checker\_det.inl, 814
  - \_\_FFLASFFPACK\_checker\_det\_INL, 814
- Checker\_Empty
  - Checker\_Empty< Field >, 411
- Checker\_Empty< Field >, 411
  - check, 412
  - Checker\_Empty, 411
- checker\_empty.h, 814
- Checker\_fgemm
  - FFLAS, 78
- checker\_fgemm.inl, 815
  - \_\_FFLASFFPACK\_checker\_fgemm\_INL, 815
- Checker\_ftrsm
  - FFLAS, 78
- checker\_ftrsm.inl, 815
  - \_\_FFLASFFPACK\_checker\_ftrsm\_INL, 815
- Checker\_invert
  - FFPACK, 309
- checker\_invert.inl, 815
  - \_\_FFLASFFPACK\_checker\_invert\_INL, 816
- Checker\_PLUQ
  - FFPACK, 309
- checker\_pluq.inl, 816
  - \_\_FFLASFFPACK\_checker\_pluq\_INL, 816
- CheckerImplem\_charpoly
  - CheckerImplem\_charpoly< Field, Polynomial >, 412
- CheckerImplem\_charpoly< Field, Polynomial >, 412
  - ~CheckerImplem\_charpoly, 412
  - check, 412
  - CheckerImplem\_charpoly, 412
- CheckerImplem\_Det
  - CheckerImplem\_Det< Field >, 413
- CheckerImplem\_Det< Field >, 413
  - ~CheckerImplem\_Det, 413
  - check, 413
  - CheckerImplem\_Det, 413
- CheckerImplem\_fgemm
  - CheckerImplem\_fgemm< Field >, 414
- CheckerImplem\_fgemm< Field >, 414
  - ~CheckerImplem\_fgemm, 414
  - check, 414
  - CheckerImplem\_fgemm, 414
- CheckerImplem\_ftrsm
  - CheckerImplem\_ftrsm< Field >, 415
- CheckerImplem\_ftrsm< Field >, 415
  - ~CheckerImplem\_ftrsm, 415
  - check, 416
  - CheckerImplem\_ftrsm, 415
- CheckerImplem\_invert
  - CheckerImplem\_invert< Field >, 416
- CheckerImplem\_invert< Field >, 416
  - ~CheckerImplem\_invert, 416
  - check, 417
  - CheckerImplem\_invert, 416
- CheckerImplem\_PLUQ
  - CheckerImplem\_PLUQ< Field >, 417
- CheckerImplem\_PLUQ< Field >, 417

- ~CheckerImplem\_PLUQ, [417](#)
- check, [417](#)
- CheckerImplem\_PLUQ, [417](#)
- checkers.doxy, [816](#)
- checkers\_fflas.h, [816](#)
- checkers\_fflas.inl, [817](#)
- FFLASFFPACK\_checkers\_fflas\_inl\_H, [817](#)
- checkers\_ffpack.h, [817](#)
- checkers\_ffpack.inl, [818](#)
- FFLASFFPACK\_checkers\_ffpack\_inl\_H, [818](#)
- checkingMessage
  - test-nullspace.C, [1108](#)
- checkMonotonicApplyP
  - test-permutations.C, [1110](#)
- checkOut
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [501](#)
- checkRPM
  - test-rpm.C, [1113](#)
- checkSymmetricRPM
  - test-rpm.C, [1113](#)
- checkZeroDimCharpoly
  - regression-check.C, [1065](#)
- checkZeroDimMinPoly
  - regression-check.C, [1065](#)
- chooseField
  - FFPACK, [394](#)
- chooseField< Givaro::ZRing< double > >
  - FFPACK, [395](#)
- chooseField< Givaro::ZRing< float > >
  - FFPACK, [395](#)
- chooseField< Givaro::ZRing< int32\_t > >
  - FFPACK, [394](#)
- chooseField< Givaro::ZRing< int64\_t > >
  - FFPACK, [395](#)
- chunk
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [747](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [749](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [754](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [756](#)
- chunkSize
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [755](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [757](#)
- clapack.C, [1058](#)
  - \_\_FFLASFFPACK\_CONFIGURATION, [1058](#)
  - \_\_FFLASFFPACK\_HAVE\_CLAPACK, [1058](#)
  - \_\_FFLASFFPACK\_HAVE\_LAPACK, [1058](#)
  - main, [1059](#)
- Classic, [418](#)
- CLASSIC\_HYBRID
  - benchmark-fgemm.C, [789](#)
- clz
  - bit\_manipulation.h, [1050](#)
- Cmax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [502](#)
- Cmin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [502](#)
- col
  - Coo< Field >, [431](#)
  - Coo< ValT, IdxT >, [429](#), [433](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [741](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [742](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [745](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [746](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [748](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [750](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [751](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [755](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [757](#)
- colblockindex
  - ForStrategy2D< blocksize\_t, Cut, Param >, [463](#)
- colBlockSize
  - ForStrategy2D< blocksize\_t, Cut, Param >, [463](#)
- coldim
  - StatsMatrix, [759](#)
- colnumblocks
  - ForStrategy2D< blocksize\_t, Cut, Param >, [462](#)
- ColRankProfileSubmatrix
  - FFPACK, [342](#), [377](#)
- ColRankProfileSubmatrix\_modular\_double
  - ffpack.C, [1007](#)
  - ffpack\_c.h, [1028](#)
- ColRankProfileSubmatrixIndices
  - FFPACK, [341](#), [377](#)
- ColRankProfileSubmatrixIndices\_modular\_double
  - ffpack.C, [1006](#)
  - ffpack\_c.h, [1028](#)
- Column, [418](#)
- ColumnEchelonForm
  - FFPACK, [322](#), [323](#), [371](#)
- ColumnEchelonForm\_modular\_double
  - ffpack.C, [996](#)
  - ffpack\_c.h, [1020](#)
- ColumnEchelonForm\_modular\_float
  - ffpack.C, [997](#)
  - ffpack\_c.h, [1020](#)
- ColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [998](#)
  - ffpack\_c.h, [1021](#)
- ColumnRankProfile
  - FFPACK, [338](#), [339](#), [376](#)
- ColumnRankProfile\_modular\_double
  - ffpack.C, [1005](#)
  - ffpack\_c.h, [1027](#)
- COMMA
  - parallel.h, [1043](#)
- CompactElement< double >, [418](#)
- type, [419](#)
- CompactElement< Element >, [418](#)

- type, [418](#)
- CompactElement< float >, [419](#)
  - type, [419](#)
- CompactElement< int16\_t >, [419](#)
  - type, [419](#)
- CompactElement< int32\_t >, [419](#)
  - type, [419](#)
- CompactElement< int64\_t >, [420](#)
  - type, [420](#)
- compatible\_data\_type< Field >, [420](#)
  - value, [420](#)
- compatible\_data\_type< Givaro::ZRing< double > >, [420](#)
  - value, [420](#)
- compatible\_data\_type< Givaro::ZRing< float > >, [420](#)
  - value, [421](#)
- compliant
  - NoSimd< T >, [521](#)
  - Simd128\_impl< true, true, false, 2 >, [572](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [592](#)
  - Simd128\_impl< true, true, true, 2 >, [599](#)
  - Simd128\_impl< true, true, true, 4 >, [608](#)
  - Simd128\_impl< true, true, true, 8 >, [617](#)
  - Simd256\_impl< true, false, true, 8 >, [628](#)
  - Simd256\_impl< true, true, false, 2 >, [638](#)
  - Simd256\_impl< true, true, false, 4 >, [652](#)
  - Simd256\_impl< true, true, false, 8 >, [666](#)
  - Simd256\_impl< true, true, true, 2 >, [673](#)
  - Simd256\_impl< true, true, true, 4 >, [683](#), [689](#)
  - Simd256\_impl< true, true, true, 8 >, [698](#)
  - Simd512\_impl< true, false, true, 8 >, [708](#)
  - Simd512\_impl< true, true, false, 8 >, [718](#)
  - Simd512\_impl< true, true, true, 8 >, [726](#)
- Compose
  - Compose< H1, H2 >, [421](#)
- Compose< H1, H2 >, [421](#)
  - Compose, [421](#)
  - first\_component, [422](#)
  - operator<<, [422](#)
  - second\_component, [422](#)
- composePermutationsLLL
  - FFPACK, [361](#)
  - ffpack.C, [993](#)
  - ffpack\_c.h, [1016](#)
- composePermutationsLLM
  - FFPACK, [362](#)
  - ffpack.C, [992](#)
  - ffpack\_c.h, [1016](#)
- composePermutationsMLM
  - FFPACK, [362](#)
  - ffpack.C, [993](#)
  - ffpack\_c.h, [1016](#)
- CompressRows
  - FFPACK::Protected, [401](#)
- CompressRowsQA
  - FFPACK::Protected, [402](#)
- CompressRowsQK
  - FFPACK::Protected, [401](#)
- computeDeviation
  - FFLAS, [159](#)
- computeFactorClassic
  - FFLAS::Protected, [219](#)
- config-blas.h, [819](#)
  - AtlasConj, [820](#)
  - blas\_enum, [820](#)
  - CBLAS\_DIAG, [820](#)
  - CBLAS\_ENUM\_DEFINED\_H, [820](#)
  - CBLAS\_EXTERNALS, [820](#)
  - CBLAS\_INT, [819](#)
  - CBLAS\_ORDER, [820](#)
  - CBLAS\_SIDE, [821](#)
  - CBLAS\_TRANSPOSE, [820](#)
  - CBLAS\_UPLO, [820](#)
  - CblasColMajor, [820](#)
  - CblasConjTrans, [820](#)
  - CblasLeft, [821](#)
  - CblasLower, [820](#)
  - CblasNonUnit, [821](#)
  - CblasNoTrans, [820](#)
  - CblasRight, [821](#)
  - CblasRowMajor, [820](#)
  - CblasTrans, [820](#)
  - CblasUnit, [821](#)
  - CblasUpper, [820](#)
- dasum\_, [822](#)
- daxpy\_, [821](#)
- dcopy\_, [823](#)
- ddot\_, [821](#)
- dgemm\_, [825](#)
- dgemv\_, [822](#)
- dger\_, [823](#)
- dnrm2\_, [822](#)
- dscal\_, [823](#)
- dtrmm\_, [824](#)
- dtrsm\_, [824](#)
- idamax\_, [822](#)
- saxpy\_, [821](#)
- scopy\_, [823](#)
- sdot\_, [821](#)
- sgemm\_, [825](#)
- sgemv\_, [822](#)
- sger\_, [823](#)
- sscal\_, [824](#)
- strmm\_, [825](#)
- strsm\_, [824](#)
- config.h, [804](#), [808](#)
  - HAVE\_BLAS, [804](#)
  - HAVE\_CBLAS, [804](#)
  - HAVE\_CXX11, [804](#)
  - HAVE\_DLFCN\_H, [805](#)
  - HAVE\_FLOAT\_H, [805](#)
  - HAVE\_INT128, [805](#)
  - HAVE\_INTPTR\_T, [805](#)
  - HAVE\_LAPACK, [805](#)
  - HAVE\_LIMITS\_H, [805](#)



- HAVE\_LITTLE\_ENDIAN, 805
- HAVE\_PTHREAD\_H, 805
- HAVE\_STDDEF\_H, 805
- HAVE\_STDINT\_H, 805
- HAVE\_STDIO\_H, 805
- HAVE\_STDLIB\_H, 805
- HAVE\_STRING\_H, 806
- HAVE\_STRINGS\_H, 806
- HAVE\_SYS\_STAT\_H, 806
- HAVE\_SYS\_TIME\_H, 806
- HAVE\_SYS\_TYPES\_H, 806
- HAVE\_UNISTD\_H, 806
- LT\_OBJDIR, 806
- OPENBLAS\_NUM\_THREADS, 806
- PACKAGE, 806
- PACKAGE\_BUGREPORT, 806
- PACKAGE\_NAME, 806
- PACKAGE\_STRING, 806
- PACKAGE\_TARNAME, 807
- PACKAGE\_URL, 807
- PACKAGE\_VERSION, 807
- SIZEOF\_\_\_INT64, 807
- SIZEOF\_CHAR, 807
- SIZEOF\_INT, 807
- SIZEOF\_LONG, 807
- SIZEOF\_LONG\_LONG, 807
- SIZEOF\_SHORT, 807
- STDC\_HEADERS, 807
- USE\_OPENMP, 807
- VERSION, 807
- CONST
  - fflas\_simd.h, 874
- const\_int
  - instrset.h, 1061
- Const\_int\_t< n >, 422
- const\_uint
  - instrset.h, 1061
- Const\_uint\_t< n >, 422
- ConstElement\_ptr
  - benchmark-fgemm-rns.C, 788
  - rns\_double, 529
  - rns\_double\_extended, 541
  - RNSInteger< RNS >, 546
  - RNSIntegerMod< RNS >, 550
- CONSTREFERENCE
  - parallel.h, 1041
- convert
  - rns\_double, 530, 531
  - rns\_double\_extended, 542, 543
  - RNSInteger< RNS >, 547
  - RNSIntegerMod< RNS >, 552
- convert\_transpose
  - rns\_double, 530
- ConvertTo< T >, 428
- COO
  - FFLAS, 83
- Coo
  - Coo< Field >, 430
  - Coo< ValT, IdxT >, 429, 432
- coo
  - HelperFlag, 472
- Coo< Field >, 430
  - col, 431
  - Coo, 430
  - deleted, 431
  - operator=, 430, 431
  - row, 431
  - val, 431
- Coo< ValT, IdxT >, 428, 431
  - col, 429, 433
  - Coo, 429, 432
  - operator=, 429, 432
  - row, 429, 432
  - Self, 428, 432
  - val, 429, 432
- coo.h, 889
- coo\_spmv.inl, 889
  - \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL, 890
- coo\_spmv.inl, 890
  - \_\_FFLASFFPACK\_fflas\_sparse\_coo\_spmv\_INL, 891
- coo\_utils.inl, 891
  - \_\_FFLASFFPACK\_fflas\_sparse\_coo\_utils\_INL, 891
- COO\_ZO
  - FFLAS, 83
- CooMat< Field >, 433
  - \_coo16, 433
  - \_coo16\_zo, 433
  - \_coo32, 433
  - \_coo32\_zo, 433
  - \_coo64, 433
  - \_coo64\_zo, 433
- CROUT
  - ffpack\_pluq.inl, 942
- CSC
  - FFLAS, 83
- CSC\_ZO
  - FFLAS, 83
- CSR
  - FFLAS, 83
- csr
  - HelperFlag, 472
- csr.h, 892
- CSR\_HYB
  - FFLAS, 83
- csr\_hyb.h, 896
- csr\_hyb\_pspmm.inl, 897
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmm\_INL, 897
- csr\_hyb\_pspmv.inl, 898
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_pspmv\_INL, 898
- csr\_hyb\_spmv.inl, 898



- \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL, ForStrategy2D< blocksize\_t, Cut, Param >, [464](#)
  - [899](#)
- csr\_hyb\_spmv.inl, [899](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_spmv\_INL, [899](#)
- csr\_hyb\_utils.inl, [899](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_HYB\_utils\_INL, [900](#)
- csr\_pspmm.inl, [892](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmm\_INL, [893](#)
- csr\_pspmv.inl, [893](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_pspmv\_INL, [894](#)
- csr\_spm.inl, [894](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spm\_INL, [895](#)
- csr\_spmv.inl, [895](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_CSR\_spmv\_INL, [896](#)
- csr\_utils.inl, [896](#)
- CSR\_ZO
  - FFLAS, [83](#)
- CsrMat< Field >, [434](#)
  - \_csr16, [434](#)
  - \_csr16\_zo, [434](#)
  - \_csr32, [434](#)
  - \_csr32\_zo, [434](#)
  - \_csr64, [434](#)
  - \_csr64\_zo, [434](#)
- cst
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [749](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [751](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [756](#)
- ctz
  - bit\_manipulation.h, [1050](#)
- CUBE
  - arithprog.C, [769](#)
  - autotune/charpoly.C, [770](#)
  - autotune/pluq.C, [775](#)
  - benchmark-checkers.C, [779](#)
  - benchmark-fsyk.C, [796](#)
  - benchmark-fsytrf.C, [796](#)
  - benchmark-ftsri.C, [799](#)
  - benchmark-inverse.C, [800](#)
  - benchmark-lqp.C, [801](#)
  - benchmark-pluq.C, [802](#)
  - benchmark-wino.C, [803](#)
  - fsyk.C, [772](#)
  - fsytrf.C, [773](#)
  - ftsri.C, [774](#)
- cuda.C, [1059](#)
  - main, [1059](#)
- current
  - ForStrategy1D< blocksize\_t, Cut, Param >, [460](#)
- Cut
    - Parallel< C, P >, [522](#)
  - cyclic\_shift\_col
    - FFPACK, [363](#), [366](#)
  - cyclic\_shift\_col\_modular\_double
    - ffpack.C, [993](#)
    - ffpack\_c.h, [1017](#)
  - cyclic\_shift\_mathPerm
    - FFPACK, [362](#)
    - ffpack.C, [993](#)
    - ffpack\_c.h, [1016](#)
  - cyclic\_shift\_row
    - FFPACK, [363](#), [366](#)
  - cyclic\_shift\_row\_col
    - FFPACK, [363](#), [366](#)
  - cyclic\_shift\_row\_modular\_double
    - ffpack.C, [993](#)
    - ffpack\_c.h, [1016](#)
- Danilevski
  - FFPACK, [351](#)
  - FFPACK::Protected, [399](#)
- dasum\_
  - config-blas.h, [822](#)
- dat
  - Sparse< \_Field, SparseMatrix\_t::COO >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [741](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [742](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [745](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [746](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [748](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [750](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [751](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [753](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [755](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [757](#)
- data
  - Argument, [407](#)
- daxpy\_
  - config-blas.h, [821](#)
- dcopy\_
  - config-blas.h, [823](#)
- ddot\_
  - config-blas.h, [821](#)
- debug.h, [1050](#)
  - FFLASFFPACK\_abort, [1051](#)
  - FFLASFFPACK\_check, [1051](#)
- DeCompressRows
  - FFPACK::Protected, [401](#)
- DeCompressRowsQA
  - FFPACK::Protected, [402](#)
- DeCompressRowsQK
  - FFPACK::Protected, [401](#)
- DefaultBoundedTag, [434](#)
- DefaultTag, [435](#)

- delayed
  - RNSIntegerMod< RNS >, 550
  - Sparse< \_Field, SparseMatrix\_t::COO >, 737
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, 739
  - Sparse< \_Field, SparseMatrix\_t::CSR >, 741
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 742
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 744
  - Sparse< \_Field, SparseMatrix\_t::ELL >, 746
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, 747
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, 749
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, 751
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 752
  - Sparse< \_Field, SparseMatrix\_t::SELL >, 754
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 756
- DelayedField
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 499
- delayedField
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 503
- DelayedField\_t
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 499
- DelayedTag, 435
- deleted
  - Coo< Field >, 431
- DENSE\_THRESHOLD
  - fflas\_sparse.h, 886
- denseCols
  - StatsMatrix, 761
- denseRows
  - StatsMatrix, 761
- Det
  - FFPACK, 334, 352, 374, 375
- det.C, 812
  - main, 812
- Det\_modular\_double
  - ffpack.C, 1004
  - ffpack\_c.h, 1025
- deviationCol
  - StatsMatrix, 760
- deviationColDifference
  - StatsMatrix, 761
- deviationRow
  - StatsMatrix, 760
- deviationRowDifference
  - StatsMatrix, 761
- DFElt
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 499
- dgemm\_
  - config-blas.h, 825
  - fblas.C, 1060
- dgemv\_
  - config-blas.h, 822
- dger\_
  - config-blas.h, 823
- digits
  - limits< char >, 486
  - limits< double >, 487
  - limits< float >, 488
  - limits< int >, 489
  - limits< long >, 490
  - limits< long long >, 491
  - limits< short int >, 493
  - limits< signed char >, 493
  - limits< unsigned char >, 494
  - limits< unsigned int >, 495
  - limits< unsigned long >, 496
  - limits< unsigned long long >, 496
  - limits< unsigned short int >, 497
- div
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 559
  - Simd256\_impl< true, false, true, 8 >, 630
  - Simd512\_impl< true, false, true, 8 >, 710
- dnrm2\_
  - config-blas.h, 822
- DNS\_BIN\_VER
  - read\_sparse.h, 911
- doApplyS
  - FFPACK, 358
- doApplyT
  - FFPACK, 360
- DotProdBoundClassic
  - FFLAS::Protected, 220
- DOUBLE\_TO\_FLOAT\_CROSSOVER
  - fflas.h, 829
  - winograd.C, 776
- dscal\_
  - config-blas.h, 823
- dtrmm\_
  - config-blas.h, 824
- dtrsm\_
  - config-blas.h, 824
- DynamicPeeling
  - FFLAS::Protected, 224
- DynamicPeeling2
  - FFLAS::Protected, 225
- EFFGFF
  - benchmark-dsytrf.C, 782
- Element
  - FieldSimd< \_Field >, 445
  - readMyMachineType< Field, mpz\_t >, 526
  - readMyMachineType< Field, T >, 525
  - rns\_double, 528
  - rns\_double\_extended, 541
  - RNSInteger< RNS >, 546
  - RNSIntegerMod< RNS >, 550
- Element\_ptr
  - benchmark-fgemm-rns.C, 788
  - readMyMachineType< Field, mpz\_t >, 526
  - readMyMachineType< Field, T >, 526

- rns\_double, 528
- rns\_double\_extended, 541
- RNSInteger< RNS >, 546
- RNSIntegerMod< RNS >, 550
- ElementTraits< double >, 435
  - value, 436
- ElementTraits< Element >, 435
  - value, 435
- ElementTraits< FFPACK::rns\_double\_elt >, 436
  - value, 436
- ElementTraits< float >, 436
  - value, 436
- ElementTraits< Givaro::Integer >, 436
  - value, 437
- ElementTraits< int16\_t >, 437
  - value, 437
- ElementTraits< int32\_t >, 437
  - value, 437
- ElementTraits< int64\_t >, 437
  - value, 438
- ElementTraits< int8\_t >, 438
  - value, 438
- ElementTraits< Reclnt::rint< K > >, 438
  - value, 438
- ElementTraits< Reclnt::rmin< K, MG > >, 438
  - value, 439
- ElementTraits< Reclnt::ruint< K > >, 439
  - value, 439
- ElementTraits< uint16\_t >, 439
  - value, 439
- ElementTraits< uint32\_t >, 439
  - value, 440
- ElementTraits< uint64\_t >, 440
  - value, 440
- ElementTraits< uint8\_t >, 440
  - value, 440
- ELL
  - FFLAS, 83
- ell
  - HelperFlag, 472
- ell.h, 900
- ell\_pspmm.inl, 900
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmm\_INL, 901
- ell\_pspmv.inl, 901
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_pspmv\_INL, 902
- ELL\_simd
  - FFLAS, 83
- ell\_simd.h, 904
- ell\_simd\_pspmv.inl, 905
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_pspmv\_INL, 906
- ell\_simd\_spmv.inl, 906
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_spmv\_INL, 906
- ell\_simd\_utils.inl, 907
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_simd\_utils\_INL, 907
- ELL\_simd\_ZO
  - FFLAS, 83
- ell\_spm.inl, 902
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spm\_INL, 903
- ell\_spmv.inl, 903
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_spmv\_INL, 904
- ell\_utils.inl, 904
  - \_\_FFLASFFPACK\_fflas\_sparse\_ELL\_utils\_INL, 904
- ELL\_ZO
  - FFLAS, 83
- ElIMat< Field >, 440
  - \_ell16, 441
  - \_ell16\_zo, 441
  - \_ell32, 441
  - \_ell32\_zo, 441
  - \_ell64, 441
  - \_ell64\_zo, 441
- ENABLE\_ALL\_CHECKINGS
  - benchmark-checkers.C, 779
  - ffpack\_ftrtr.inl, 936
  - test-fdot.C, 1074
  - test-fgemm-check.C, 1075
  - test-fsyr2k.C, 1086
  - test-fsyrk.C, 1088
  - test-ftrmv.C, 1092
  - test-ftrsm-check.C, 1093
  - test-ftrsm.C, 1094
  - test-ftrssyr2k.C, 1095
  - test-ftrstr.C, 1097
  - test-ftrsv.C, 1098
  - test-ftrtri.C, 1099
  - test-invert-check.C, 1100
  - test-pluq-check.C, 1111
- ENABLE\_CHECKER\_charpoly
  - test-charpoly-check.C, 1066
- ENABLE\_CHECKER\_Det
  - test-det-check.C, 1069
- ENABLE\_CHECKER\_fgemm
  - test-fgemm.C, 1077
- end
  - ForStrategy1D< blocksize\_t, Cut, Param >, 460
- END\_OF\_ARGUMENTS
  - args-parser.h, 1048
- END\_PARALLEL\_MAIN
  - parallel.h, 1041
- eq
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 560
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 565
- Simd128\_impl< true, true, false, 2 >, 576
- Simd128\_impl< true, true, false, 4 >, 585

- Simd128\_impl< true, true, false, 8 >, [595](#)
- Simd128\_impl< true, true, true, 2 >, [604](#)
- Simd128\_impl< true, true, true, 4 >, [612](#)
- Simd128\_impl< true, true, true, 8 >, [621](#)
- Simd256\_impl< true, false, true, 8 >, [631](#)
- Simd256\_impl< true, true, false, 2 >, [642](#)
- Simd256\_impl< true, true, false, 4 >, [659](#)
- Simd256\_impl< true, true, false, 8 >, [669](#)
- Simd256\_impl< true, true, true, 2 >, [678](#)
- Simd256\_impl< true, true, true, 4 >, [688](#), [693](#)
- Simd256\_impl< true, true, true, 8 >, [703](#)
- Simd512\_impl< true, false, true, 8 >, [711](#)
- Simd512\_impl< true, true, false, 8 >, [722](#)
- Simd512\_impl< true, true, true, 8 >, [731](#)
- eval\_func\_on\_array
  - test-simd.C, [1116](#)
- example
  - Argument, [407](#)
- examples/charpoly.C
  - main, [771](#)
- examples/pluq.C
  - main, [776](#)
- fadd
  - FFLAS, [85–87](#), [90](#), [170](#), [171](#), [178](#), [179](#)
  - FFLAS::details, [207](#), [208](#)
- fadd\_1\_modular\_double
  - fflas\_c.h, [958](#)
  - fflas\_lvl1.C, [979](#)
- fadd\_2\_modular\_double
  - fflas\_c.h, [961](#)
  - fflas\_lvl2.C, [984](#)
- faddin
  - FFLAS, [85](#), [89](#), [170](#), [180](#)
- faddin\_1\_modular\_double
  - fflas\_c.h, [958](#)
  - fflas\_lvl1.C, [979](#)
- faddin\_2\_modular\_double
  - fflas\_c.h, [962](#)
  - fflas\_lvl2.C, [985](#)
- Failure, [441](#)
  - \_errorStream, [443](#)
  - Failure, [442](#)
  - operator(), [442](#)
  - print, [443](#)
  - setErrorStream, [442](#)
- failure
  - FFPACK, [381](#)
- FailureCharpolyCheck, [443](#)
- FailureDetCheck, [443](#)
- FailureFgemmCheck, [443](#)
- FailureInvertCheck, [443](#)
- FailurePLUQCheck, [444](#)
- FailureTrsmCheck, [444](#)
- fassign
  - FFLAS, [91](#), [92](#), [166](#), [171](#)
- fassign\_1\_modular\_double
  - fflas\_c.h, [956](#)
  - fflas\_lvl1.C, [978](#)
- fassign\_2\_modular\_double
  - fflas\_c.h, [959](#)
  - fflas\_lvl2.C, [981](#)
- faxpby
  - FFLAS, [138](#), [144](#)
- faxpy
  - FFLAS, [93](#), [94](#), [168](#), [176](#)
- faxpy\_1\_modular\_double
  - fflas\_c.h, [957](#)
  - fflas\_lvl1.C, [978](#)
- faxpy\_2\_modular\_double
  - fflas\_c.h, [961](#)
  - fflas\_lvl2.C, [983](#)
- fblas.C, [1059](#)
  - \_\_FFLASFFPACK\_CONFIGURATION, [1059](#)
  - dgemm\_, [1060](#)
  - main, [1060](#)
- fconvert
  - FFLAS, [135](#), [143](#), [164](#)
- fconvert\_rns
  - FFLAS, [161](#), [162](#)
- fconvert\_trans\_rns
  - FFLAS, [161](#)
- fdot
  - FFLAS, [94–96](#), [139](#), [169](#), [190](#)
- fdot\_1\_modular\_double
  - fflas\_c.h, [957](#)
  - fflas\_lvl1.C, [979](#)
- fequal
  - FFLAS, [138](#), [141](#), [166](#), [172](#)
- fequal\_1\_modular\_double
  - fflas\_c.h, [956](#)
  - fflas\_lvl1.C, [978](#)
- fequal\_2\_modular\_double
  - fflas\_c.h, [959](#)
  - fflas\_lvl2.C, [981](#)
- FFLAS, [50](#), [53](#)
  - alignable, [196](#)
  - alignable< Givaro::Integer \* >, [196](#)
  - BaseTimer, [80](#)
  - bitsize, [145](#)
  - bitsize< Givaro::ZRing< Givaro::Integer > >, [146](#)
  - BlockCuts, [187](#), [189](#)
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >, [189](#)
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >, [188](#)
  - BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >, [189](#)
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >, [188](#)
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >, [188](#)
  - BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >, [189](#)
  - BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >, [188](#)

BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >, 188  
BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >, 189  
BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >, 187  
cblas\_imprsm, 133  
Checker\_fgemm, 78  
Checker\_ftrsm, 78  
computeDeviation, 159  
COO, 83  
COO\_ZO, 83  
CSC, 83  
CSC\_ZO, 83  
CSR, 83  
CSR\_HYB, 83  
CSR\_ZO, 83  
ELL, 83  
ELL\_simd, 83  
ELL\_simd\_ZO, 83  
ELL\_ZO, 83  
fadd, 85–87, 90, 170, 171, 178, 179  
faddin, 85, 89, 170, 180  
fassign, 91, 92, 166, 171  
faxpby, 138, 144  
faxpy, 93, 94, 168, 176  
fconvert, 135, 143, 164  
fconvert\_rns, 161, 162  
fconvert\_trans\_rns, 161  
fdot, 94–96, 139, 169, 190  
fequal, 138, 141, 166, 172  
FFLAS\_BASE, 82  
fflas\_delete, 160, 197  
FFLAS\_DIAG, 82  
FFLAS\_FORMAT, 83  
fflas\_new, 160–162, 197  
FFLAS\_ORDER, 81  
FFLAS\_SIDE, 82  
FFLAS\_TRANSPOSE, 81  
FFLAS\_UPLO, 81  
FflasAuto, 84  
FflasBinary, 84  
FflasColMajor, 81  
FflasDense, 84  
FflasDouble, 83  
FflasFloat, 83  
FflasGeneric, 83  
FflasLeft, 82  
FflasLower, 82  
FflasMaple, 84  
FflasMath, 84  
FflasNonUnit, 82  
FflasNoTrans, 81  
FflasRight, 82  
FflasRowMajor, 81  
FflasSageMath, 84  
FflasSMS, 84  
FflasTrans, 81  
FflasUnit, 82  
FflasUpper, 82  
fgemm, 97–99, 101–106, 149, 185, 186  
fgemv, 106–112, 180, 193  
fger, 112–116, 182  
fidentity, 142, 173  
finit, 117, 119, 120, 135, 142, 163, 174  
finit\_rns, 160, 162  
finit\_trans\_rns, 161  
fiszero, 138, 141, 166, 172  
fmove, 145, 177  
fneg, 136, 144, 165, 175  
fnegin, 136, 143, 164, 175  
ForceCheck\_fgemm, 78  
ForceCheck\_ftrsm, 78  
frand, 137, 140  
freduce, 116–118, 120, 162, 163, 173, 174  
freduce\_constoverride, 117, 119  
freivalds, 120  
fscal, 122–126, 167, 176  
fscaln, 121, 123–126, 167, 176  
fspmm, 150  
fspmv, 150, 158  
fsquare, 99–101, 187  
fsub, 85, 89, 170, 178  
fsubin, 85, 90, 179  
fswap, 139, 169  
fsyr2k, 126  
fsyrk, 127–129  
ftrmm, 130, 131, 184  
ftrmv, 146  
ftrsm, 132, 133, 146, 149, 150, 183  
ftrsv, 134, 182  
fzero, 137, 140, 165, 171  
getDataTypes, 157  
getSeed, 198  
getStat, 159  
getTLBSize, 198  
has\_equal, 80  
has\_minus, 79  
has\_minus\_eq, 80  
has\_mul, 80  
has\_mul\_eq, 80  
has\_plus, 79  
has\_plus\_eq, 80  
HYB\_ZO, 83  
igemm\_, 134  
InfNorm, 84  
max3, 84  
max4, 84  
min3, 84  
min4, 84  
MKLSparseMatrixFormat, 79  
mone, 83  
NoSimdSparseMatrix, 79  
NotMKLSparseMatrixFormat, 79  
NotZOSparseMatrix, 79  
number\_kind, 83

- one, [83](#)
- operator<<, [156](#)
- other, [83](#)
- parseArguments, [194](#)
- pfadd, [86](#)
- pfaddin, [87](#)
- pfgemm, [147](#), [190–192](#)
- pfgemm\_1D\_rec, [147](#)
- pfgemm\_2D\_rec, [148](#)
- pfgemm\_3D\_rec, [148](#)
- pfgemm\_3D\_rec2, [149](#)
- pfrand, [190](#)
- pfreduce, [118](#)
- pfsub, [86](#)
- pfsubin, [87](#)
- pfzero, [190](#)
- preamble, [195](#)
- prefetch, [197](#)
- queryCacheSizes, [198](#)
- queryL1CacheSize, [198](#)
- queryTopLevelCacheSize, [198](#)
- readDnsFormat, [158](#)
- readMachineType, [157](#)
- ReadMatrix, [195](#)
- readSmsFormat, [156](#)
- readSprFormat, [157](#)
- SELL, [83](#)
- SELL\_ZO, [83](#)
- SimdSparseMatrix, [79](#)
- sparse\_delete, [151](#), [152](#), [154–156](#), [158](#)
- sparse\_init, [151–156](#), [159](#)
- sparse\_print, [152](#), [156](#), [159](#)
- SparseMatrix\_t, [83](#)
- SysTimer, [81](#)
- Timer, [80](#)
- UserTimer, [81](#)
- writeCommandString, [194](#)
- writeDnsFormat, [158](#)
- WriteMatrix, [194](#), [196](#)
- WritePermutation, [196](#)
- zero, [83](#)
- ZOSparseMatrix, [79](#)
- fflas-101\_1.C, [1119](#)
- main, [1120](#)
- fflas-101\_3.C, [1120](#)
- main, [1120](#)
- FFLAS-FFPACK, [49](#)
- FFLAS-FFPACK fields, [51](#)
- fflas-ffpack-config.h, [825](#)
- GCC\_VERSION, [826](#)
- fflas-ffpack-default-thresholds.h, [826](#)
- \_\_FFLASFFPACK\_ARITHPROG\_THRESHOLD, [827](#)
- \_\_FFLASFFPACK\_CHARPOLY\_Danilevskii\_LUKrylov\_ArithProg\_THRESHOLD, [827](#)
- \_\_FFLASFFPACK\_CHARPOLY\_LUKrylov\_ArithProg\_THRESHOLD, [827](#)
- \_\_FFLASFFPACK\_FSYRK\_THRESHOLD, [827](#)
- \_\_FFLASFFPACK\_FSYTRF\_THRESHOLD, [827](#)
- \_\_FFLASFFPACK\_FTRTRI\_THRESHOLD, [827](#)
- \_\_FFLASFFPACK\_PLUQ\_THRESHOLD, [827](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD, [826](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL, [826](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_BAL\_FLT, [827](#)
- \_\_FFLASFFPACK\_WINOTHRESHOLD\_FLT, [826](#)
- fflas-ffpack-thresholds.h, [827](#)
- fflas-ffpack.doxy, [827](#)
- fflas-ffpack.h, [827](#)
- fflas-ffpack/config.h
- \_\_FFLASFFPACK\_HAVE\_BLAS, [808](#)
- \_\_FFLASFFPACK\_HAVE\_CBLAS, [808](#)
- \_\_FFLASFFPACK\_HAVE\_CXX11, [809](#)
- \_\_FFLASFFPACK\_HAVE\_DLFCN\_H, [809](#)
- \_\_FFLASFFPACK\_HAVE\_FLOAT\_H, [809](#)
- \_\_FFLASFFPACK\_HAVE\_INT128, [809](#)
- \_\_FFLASFFPACK\_HAVE\_INTTYPES\_H, [809](#)
- \_\_FFLASFFPACK\_HAVE\_LAPACK, [809](#)
- \_\_FFLASFFPACK\_HAVE\_LIMITS\_H, [809](#)
- \_\_FFLASFFPACK\_HAVE\_LITTLE\_ENDIAN, [809](#)
- \_\_FFLASFFPACK\_HAVE\_PTHREAD\_H, [809](#)
- \_\_FFLASFFPACK\_HAVE\_STDDEF\_H, [809](#)
- \_\_FFLASFFPACK\_HAVE\_STDINT\_H, [809](#)
- \_\_FFLASFFPACK\_HAVE\_STDIO\_H, [809](#)
- \_\_FFLASFFPACK\_HAVE\_STDLIB\_H, [810](#)
- \_\_FFLASFFPACK\_HAVE\_STRINGS\_H, [810](#)
- \_\_FFLASFFPACK\_HAVE\_STRING\_H, [810](#)
- \_\_FFLASFFPACK\_HAVE\_SYS\_STAT\_H, [810](#)
- \_\_FFLASFFPACK\_HAVE\_SYS\_TIME\_H, [810](#)
- \_\_FFLASFFPACK\_HAVE\_SYS\_TYPES\_H, [810](#)
- \_\_FFLASFFPACK\_HAVE\_UNISTD\_H, [810](#)
- \_\_FFLASFFPACK\_LT\_OBJDIR, [810](#)
- \_\_FFLASFFPACK\_OPENBLAS\_NUM\_THREADS, [810](#)
- \_\_FFLASFFPACK\_PACKAGE, [810](#)
- \_\_FFLASFFPACK\_PACKAGE\_BUGREPORT, [810](#)
- \_\_FFLASFFPACK\_PACKAGE\_NAME, [810](#)
- \_\_FFLASFFPACK\_PACKAGE\_STRING, [811](#)
- \_\_FFLASFFPACK\_PACKAGE\_TARNAME, [811](#)
- \_\_FFLASFFPACK\_PACKAGE\_URL, [811](#)
- \_\_FFLASFFPACK\_PACKAGE\_VERSION, [811](#)
- \_\_FFLASFFPACK\_SIZEOF\_CHAR, [811](#)
- \_\_FFLASFFPACK\_SIZEOF\_INT, [811](#)
- \_\_FFLASFFPACK\_SIZEOF\_LONG, [811](#)
- \_\_FFLASFFPACK\_SIZEOF\_LONG\_LONG, [811](#)
- \_\_FFLASFFPACK\_SIZEOF\_SHORT, [811](#)
- \_\_FFLASFFPACK\_SIZEOF\_\_INT64, [811](#)
- \_\_FFLASFFPACK\_STDC\_HEADERS, [811](#)
- \_\_FFLASFFPACK\_USE\_OPENMP, [811](#)
- \_\_FFLASFFPACK\_VERSION, [812](#)
- fflas.doxy, [828](#)
- fflas.thresholds, [828](#)
- DOUBLE\_TO\_FLOAT\_CROSSOVER, [829](#)
- FFLAS::BLAS3, [199](#)
- Bini, [200](#)

- Winograd, 201
- Winograd\_L\_S, 204
- Winograd\_LR\_S, 204
- Winograd\_R\_S, 205
- WinogradAcc\_2\_24, 202
- WinogradAcc\_2\_27, 202
- WinogradAcc\_3\_21, 202
- WinogradAcc\_3\_23, 201
- WinogradAcc\_L\_S, 204
- WinogradAcc\_LR, 203
- WinogradAcc\_R\_S, 203
- WinoPar, 200
- FFLAS::csr\_hyb\_details, 205
- FFLAS::CuttingStrategy, 205
  - RNSModulus, 206
- FFLAS::details, 206
  - BlockingFactor, 214
  - fadd, 207, 208
  - freduce, 209
  - fscal, 210
  - fscaln, 210
  - gebp, 213
  - igebb11, 212
  - igebb14, 211
  - igebb21, 212
  - igebb24, 211
  - igebb41, 211
  - igebb44, 211
  - igebp, 212
  - pack\_lhs, 213
  - pack\_rhs, 213
- FFLAS::details\_spmv, 214
- FFLAS::ElementCategories, 214
- FFLAS::FieldCategories, 215
- FFLAS::MMHelperAlgo, 215
- FFLAS::ModeCategories, 215
- FFLAS::ParSeqHelper, 216
- FFLAS::Protected, 216
  - computeFactorClassic, 219
  - DotProdBoundClassic, 220
  - DynamicPeeling, 224
  - DynamicPeeling2, 225
  - fgemm\_convert, 220
  - fgemv\_convert, 225
  - fger\_convert, 226
  - fsquareCommon, 223
  - igemm, 228
  - igemm\_colmajor, 228
  - MatF2MatD\_Triangular, 229
  - MatF2MatFI\_Triangular, 229
  - min\_types, 226, 227
  - NeedDoublePreAddReduction, 222
  - NeedPreAddReduction, 221
  - NeedPreSubReduction, 221
  - ScalAndReduce, 222
  - TRSMBound, 220
  - unfit, 227
  - WinogradCalc, 225
  - WinogradSteps, 224
  - WinogradThreshold, 223, 224
- FFLAS::sell\_details, 229
- FFLAS::sparse\_details, 230
  - fspmm, 236–238
  - fspmm\_dispatch, 236
  - fspmv, 233–235, 243, 244
  - fspmv\_dispatch, 233
  - init\_y, 232, 233
  - pfspmm, 239–241
  - pfspmm\_dispatch, 239
  - pfspmv, 242, 243
- FFLAS::sparse\_details\_impl, 244
  - fspmm, 252, 253, 260, 261, 267, 271, 272, 281
  - fspmm\_mone, 254, 262, 272, 273
  - fspmm\_mone\_simd\_aligned, 255, 262, 274
  - fspmm\_mone\_simd\_unaligned, 255, 263, 274
  - fspmm\_one, 254, 261, 272, 273
  - fspmm\_one\_simd\_aligned, 254, 262, 273
  - fspmm\_one\_simd\_unaligned, 254, 262, 273
  - fspmm\_simd\_aligned, 253, 261
  - fspmm\_simd\_unaligned, 253, 261
  - fspmv, 255, 256, 263, 267, 268, 274, 275, 277, 278, 281–284
  - fspmv\_mone, 256, 264, 275, 278, 279, 285
  - fspmv\_mone\_simd, 279, 285
  - fspmv\_one, 256, 264, 275, 278, 284, 285
  - fspmv\_one\_simd, 279, 285
  - fspmv\_simd, 277, 278, 284
  - pfspmm, 257, 264–266, 268, 269, 279, 280
  - pfspmm\_mone, 258
  - pfspmm\_one, 257, 258
  - pfspmm\_zo, 269, 270
  - pfspmv, 258, 259, 266, 270, 276, 280, 282
  - pfspmv\_mone, 259, 260, 271, 276, 277, 283
  - pfspmv\_one, 259, 260, 271, 276, 277, 283
  - pfspmv\_task, 259
- FFLAS::StrategyParameter, 286
- FFLAS::StructureHelper, 286
- FFLAS::vectorised, 286
  - add, 289
  - addp, 288
  - modp, 291
  - reduce, 289–291
  - scalp, 291
  - sub, 289
  - subp, 288
  - VEC\_ADD, 288
  - VEC\_SUB, 288
- FFLAS::vectorised::unswitch, 292
  - modp, 292
  - scalp, 293
- fflas\_101.C, 1120
  - main, 1120
- fflas\_101\_lml1.C, 1121
  - main, 1121
- FFLAS\_BASE
  - FFLAS, 82



- fflas\_bounds.inl, 829
  - \_\_FFLASFFPACK\_fflas\_bounds\_INL, 830
  - FFLAS\_INT\_TYPE, 830
- fflas\_c.h, 951
  - fadd\_1\_modular\_double, 958
  - fadd\_2\_modular\_double, 961
  - faddin\_1\_modular\_double, 958
  - faddin\_2\_modular\_double, 962
  - fassign\_1\_modular\_double, 956
  - fassign\_2\_modular\_double, 959
  - faxpy\_1\_modular\_double, 957
  - faxpy\_2\_modular\_double, 961
  - fdot\_1\_modular\_double, 957
  - fequal\_1\_modular\_double, 956
  - fequal\_2\_modular\_double, 959
  - FFLAS\_C\_BASE, 955
  - FFLAS\_C\_DIAG, 954
  - FFLAS\_C\_ORDER, 954
  - FFLAS\_C\_SIDE, 955
  - FFLAS\_C\_TRANSPOSE, 954
  - FFLAS\_C\_UPLO, 954
  - FFLAS\_COMPILED, 953
  - FflasColMajor, 954
  - FflasDouble, 955
  - FflasFloat, 955
  - FflasGeneric, 955
  - FflasLeft, 955
  - FflasLower, 954
  - FflasNonUnit, 955
  - FflasNoTrans, 954
  - FflasRight, 955
  - FflasRowMajor, 954
  - FflasTrans, 954
  - FflasUnit, 955
  - FflasUpper, 954
  - fgemm\_3\_modular\_double, 964
  - fgemv\_2\_modular\_double, 962
  - fger\_2\_modular\_double, 963
  - fidentity\_2\_modular\_double, 959
  - fiszero\_1\_modular\_double, 956
  - fiszero\_2\_modular\_double, 959
  - fmove\_2\_modular\_double, 961
  - fneg\_1\_modular\_double, 956
  - fneg\_2\_modular\_double, 960
  - fnegin\_1\_modular\_double, 956
  - fnegin\_2\_modular\_double, 960
  - freduce\_1\_modular\_double, 955
  - freduce\_2\_modular\_double, 960
  - freducein\_1\_modular\_double, 955
  - freducein\_2\_modular\_double, 960
  - fscal\_1\_modular\_double, 957
  - fscal\_2\_modular\_double, 961
  - fscalin\_1\_modular\_double, 957
  - fscalin\_2\_modular\_double, 960
  - fsquare\_3\_modular\_double, 964
  - fsub\_1\_modular\_double, 958
  - fsub\_2\_modular\_double, 962
  - fsubin\_1\_modular\_double, 958
  - fsubin\_2\_modular\_double, 962
  - fswap\_1\_modular\_double, 957
  - ftmm\_3\_modular\_double, 963
  - ftsm\_3\_modular\_double, 963
  - ftsv\_2\_modular\_double, 963
  - fzero\_1\_modular\_double, 956
  - fzero\_2\_modular\_double, 959
  - FFLAS\_C\_BASE
    - fflas\_c.h, 955
  - FFLAS\_C\_DIAG
    - fflas\_c.h, 954
    - ffpack\_c.h, 1013
  - FFLAS\_C\_ORDER
    - fflas\_c.h, 954
    - ffpack\_c.h, 1013
  - FFLAS\_C\_SIDE
    - fflas\_c.h, 955
    - ffpack\_c.h, 1014
  - FFLAS\_C\_TRANSPOSE
    - fflas\_c.h, 954
    - ffpack\_c.h, 1013
  - FFLAS\_C\_UPLO
    - fflas\_c.h, 954
    - ffpack\_c.h, 1013
  - FFLAS\_COMPILED
    - fflas\_c.h, 953
    - ffpack\_inst.C, 1031
    - ffpack\_inst.h, 1032
  - fflas\_const\_cast
    - FFPACK, 366, 380
  - fflas\_delete
    - FFLAS, 160, 197
  - FFLAS\_DIAG
    - FFLAS, 82
  - FFLAS\_ELT
    - fflas\_L1\_inst.C, 965, 966
    - fflas\_L1\_inst.h, 966, 967
    - fflas\_L2\_inst.C, 969
    - fflas\_L2\_inst.h, 970
    - fflas\_L3\_inst.C, 973
    - fflas\_L3\_inst.h, 974
    - ffpack\_inst.C, 1031, 1032
    - ffpack\_inst.h, 1033
  - fflas\_enum.h, 830
  - fflas\_fadd.h, 831
  - fflas\_fadd.inl, 832
    - \_\_FFLASFFPACK\_fadd\_INL, 833
  - fflas\_fassign.h, 833
  - fflas\_fassign.inl, 833
    - \_\_FFLASFFPACK\_fassign\_INL, 834
  - fflas\_faxpy.inl, 834
    - \_\_FFLASFFPACK\_faxpy\_INL, 834
  - fflas\_fdot.inl, 835
    - \_\_FFLASFFPACK\_fdot\_INL, 835
  - fflas\_fgemm.inl, 836
    - \_\_FFLASFFPACK\_fgemm\_INL, 838
  - fflas\_fgemv.inl, 845
    - \_\_FFLASFFPACK\_fgemv\_INL, 846



fflas\_fgernv\_mp.inl, 847  
     \_\_FFLASFFPACK\_fgernv\_mp\_INL, 847  
 fflas\_fger.inl, 847  
     \_\_FFLASFFPACK\_fger\_INL, 848  
 fflas\_fger\_mp.inl, 849  
     \_\_FFPACK\_fger\_mp\_INL, 849  
 FFLAS\_FIELD  
     fflas\_L1\_inst.C, 965  
     fflas\_L1\_inst.h, 966  
     fflas\_L2\_inst.C, 969  
     fflas\_L2\_inst.h, 970  
     fflas\_L3\_inst.C, 973  
     fflas\_L3\_inst.h, 974  
     ffpack\_inst.C, 1031, 1032  
     ffpack\_inst.h, 1033  
 FFLAS\_FORMAT  
     FFLAS, 83  
 fflas\_freduces.h, 849  
 fflas\_freduces.inl, 850  
     \_\_FFLASFFPACK\_fflas\_freduces\_INL, 852  
     FFLASFFPACK\_COPY\_REDUCE, 852  
 fflas\_freduces\_mp.inl, 852  
     \_\_FFLASFFPACK\_fflas\_freduces\_mp\_INL, 852  
 fflas\_freivalds.inl, 852  
     \_\_FFLASFFPACK\_freivalds\_INL, 853  
 fflas\_fscal.h, 853  
 fflas\_fscal.inl, 853  
     \_\_FFLASFFPACK\_fscal\_INL, 854  
 fflas\_fscal\_mp.inl, 855  
     \_\_FFLASFFPACK\_fscal\_mp\_INL, 855  
 fflas\_fsyr2k.inl, 855  
     \_\_FFLASFFPACK\_fflas\_fsyr2k\_INL, 856  
 fflas\_fsyrk.inl, 856  
     \_\_FFLASFFPACK\_fflas\_fsyrk\_INL, 857  
 fflas\_ftrmm.inl, 857  
     \_\_FFLASFFPACK\_ftrmm\_INL, 857  
 fflas\_ftrsm.inl, 857  
     \_\_FFLASFFPACK\_ftrsm\_INL, 858  
 fflas\_ftrsm\_mp.inl, 858  
     \_\_FFPACK\_ftrsm\_mp\_INL, 859  
 fflas\_ftrsv.inl, 859  
     \_\_FFLASFFPACK\_ftrsv\_INL, 859  
 fflas\_helpers.inl, 859  
     \_\_FFLASFFPACK\_fflas\_fflas\_mmhelper\_INL, 860  
 FFLAS\_INT\_TYPE  
     fflas\_bounds.inl, 830  
 fflas\_intrinsic.h, 1051  
 fflas\_io.h, 1051  
 fflas\_L1\_inst.C, 964  
     \_\_FFLAS\_L1\_INST\_C, 965  
     FFLAS\_ELT, 965, 966  
     FFLAS\_FIELD, 965  
     INST\_OR\_DECL, 965  
 fflas\_L1\_inst.h, 966  
     FFLAS\_ELT, 966, 967  
     FFLAS\_FIELD, 966  
     INST\_OR\_DECL, 966  
 fflas\_L1\_inst\_implement.inl, 967  
 fflas\_L2\_inst.C, 968  
     \_\_FFLAS\_L2\_INST\_C, 968  
     FFLAS\_ELT, 969  
     FFLAS\_FIELD, 969  
     INST\_OR\_DECL, 969  
 fflas\_L2\_inst.h, 969  
     FFLAS\_ELT, 970  
     FFLAS\_FIELD, 970  
     INST\_OR\_DECL, 970  
 fflas\_L2\_inst\_implement.inl, 971  
 fflas\_L3\_inst.C, 972  
     \_\_FFLAS\_L3\_INST\_C, 973  
     FFLAS\_ELT, 973  
     FFLAS\_FIELD, 973  
     INST\_OR\_DECL, 973  
 fflas\_L3\_inst.h, 973  
     FFLAS\_ELT, 974  
     FFLAS\_FIELD, 974  
     INST\_OR\_DECL, 974  
 fflas\_L3\_inst\_implement.inl, 975  
     \_\_FFLAS\_\_TRSM\_READONLY, 975  
 fflas\_level1.inl, 864  
     \_\_FFLASFFPACK\_fflas\_fflas\_level1\_INL, 866  
 fflas\_level2.inl, 866  
     \_\_FFLASFFPACK\_fflas\_fflas\_level2\_INL, 869  
 fflas\_level3.inl, 869  
     \_\_FFLASFFPACK\_fflas\_fflas\_level3\_INL, 871  
     \_\_FFLAS\_\_TRSM\_READONLY, 871  
 fflas\_lvl1.C, 976  
     fadd\_1\_modular\_double, 979  
     faddin\_1\_modular\_double, 979  
     fassign\_1\_modular\_double, 978  
     faxpy\_1\_modular\_double, 978  
     fdot\_1\_modular\_double, 979  
     fequal\_1\_modular\_double, 978  
     fiszero\_1\_modular\_double, 977  
     fneg\_1\_modular\_double, 977  
     fnegin\_1\_modular\_double, 977  
     freduce\_1\_modular\_double, 977  
     freducein\_1\_modular\_double, 976  
     fscal\_1\_modular\_double, 978  
     fscalin\_1\_modular\_double, 978  
     fsub\_1\_modular\_double, 979  
     fsubin\_1\_modular\_double, 980  
     fswap\_1\_modular\_double, 979  
     fzero\_1\_modular\_double, 977  
 fflas\_lvl2.C, 980  
     fadd\_2\_modular\_double, 984  
     faddin\_2\_modular\_double, 985  
     fassign\_2\_modular\_double, 981  
     faxpy\_2\_modular\_double, 983  
     fequal\_2\_modular\_double, 981  
     fgemv\_2\_modular\_double, 985  
     fger\_2\_modular\_double, 985  
     fidentity\_2\_modular\_double, 982  
     fiszero\_2\_modular\_double, 982  
     fmove\_2\_modular\_double, 984  
     fneg\_2\_modular\_double, 983

- fnegin\_2\_modular\_double, 982
- freduce\_2\_modular\_double, 982
- freducein\_2\_modular\_double, 982
- fscal\_2\_modular\_double, 983
- fscalin\_2\_modular\_double, 983
- fsub\_2\_modular\_double, 984
- fsubin\_2\_modular\_double, 984
- ftsv\_2\_modular\_double, 985
- fzero\_2\_modular\_double, 981
- fflas\_lvl3.C, 986
  - fgemm\_3\_modular\_double, 987
  - fsquare\_3\_modular\_double, 987
  - ftmm\_3\_modular\_double, 987
  - ftsm\_3\_modular\_double, 986
- fflas\_memory.h, 1052
- fflas\_new
  - FFLAS, 160–162, 197
- FFLAS\_ORDER
  - FFLAS, 81
- fflas\_pfgemm.inl, 871
  - \_\_FFLASFFPACK\_DIMKPENALTY, 872
  - \_\_FFLASFFPACK\_SEQPARTHRESHOLD, 872
  - \_\_FFLASFFPACK\_fflas\_pfgemm\_INL, 872
- fflas\_pftsm.inl, 872
  - \_\_FFLASFFPACK\_fflas\_pftsm\_INL, 873
  - PTRSM\_HYBRID\_THRESHOLD, 873
- fflas\_plevel1.h, 1038
- fflas\_randommatrix.h, 1053
- FFLAS\_SIDE
  - FFLAS, 82
- fflas\_simd.h, 873
  - CONST, 874
  - FLOAT\_MOD, 874
  - INLINE, 874
  - NORML\_MOD, 874
  - PURE, 874
  - Simd, 875
  - SIMD\_INT, 874
- fflas\_sparse.C, 988
- fflas\_sparse.h, 882
  - \_\_FFLASFFPACK\_CACHE\_LINE\_SIZE, 886
  - assume\_aligned, 886
  - DENSE\_THRESHOLD, 886
  - index\_t, 886
  - ROUND\_DOWN, 886
- fflas\_sparse.inl, 887
  - \_\_FFLASFFPACK\_fflas\_fflas\_sparse\_INL, 889
- FFLAS\_TRANSPOSE
  - FFLAS, 81
- FFLAS\_UPLO
  - FFLAS, 81
- FflasAuto
  - FFLAS, 84
- FflasBinary
  - FFLAS, 84
- FflasColMajor
  - FFLAS, 81
  - fflas\_c.h, 954
- ffpack\_c.h, 1013
- FflasDense
  - FFLAS, 84
- FflasDouble
  - FFLAS, 83
  - fflas\_c.h, 955
- FFLASFFPACK\_abort
  - debug.h, 1051
- FFLASFFPACK\_check
  - debug.h, 1051
- FFLASFFPACK\_checkers\_fflas\_inl\_H
  - checkers\_fflas.inl, 817
- FFLASFFPACK\_checkers\_ffpack\_inl\_H
  - checkers\_ffpack.inl, 818
- FFLASFFPACK\_COPY\_REDUCE
  - fflas\_freduce.inl, 852
- FFLASFFPACK\_PERM\_BKSIZE
  - ffpack\_permutation.inl, 941
- FflasFloat
  - FFLAS, 83
  - fflas\_c.h, 955
- FflasGeneric
  - FFLAS, 83
  - fflas\_c.h, 955
- FflasLeft
  - FFLAS, 82
  - fflas\_c.h, 955
  - ffpack\_c.h, 1014
- FflasLower
  - FFLAS, 82
  - fflas\_c.h, 954
  - ffpack\_c.h, 1013
- FflasMaple
  - FFLAS, 84
- FflasMath
  - FFLAS, 84
- FflasNonUnit
  - FFLAS, 82
  - fflas\_c.h, 955
  - ffpack\_c.h, 1014
- FflasNoTrans
  - FFLAS, 81
  - fflas\_c.h, 954
  - ffpack\_c.h, 1013
- FflasRight
  - FFLAS, 82
  - fflas\_c.h, 955
  - ffpack\_c.h, 1014
- FflasRowMajor
  - FFLAS, 81
  - fflas\_c.h, 954
  - ffpack\_c.h, 1013
- FflasSageMath
  - FFLAS, 84
- FflasSMS
  - FFLAS, 84
- FflasTrans
  - FFLAS, 81

- fflas\_c.h, [954](#)
- ffpack\_c.h, [1013](#)
- FflasUnit
  - FFLAS, [82](#)
  - fflas\_c.h, [955](#)
  - ffpack\_c.h, [1014](#)
- FflasUpper
  - FFLAS, [82](#)
  - fflas\_c.h, [954](#)
  - ffpack\_c.h, [1013](#)
- FFPACK, [50](#), [293](#)
  - \_PLUQ, [364](#)
  - applyP, [310](#), [311](#), [367](#)
  - applyP\_block, [358](#)
  - buildMatrix, [351](#)
  - CharPoly, [329](#), [330](#), [352](#), [372](#)
  - Checker\_charpoly, [309](#)
  - Checker\_Det, [309](#)
  - Checker\_invert, [309](#)
  - Checker\_PLUQ, [309](#)
  - chooseField, [394](#)
  - chooseField< Givaro::ZRing< double > >, [395](#)
  - chooseField< Givaro::ZRing< float > >, [395](#)
  - chooseField< Givaro::ZRing< int32\_t > >, [394](#)
  - chooseField< Givaro::ZRing< int64\_t > >, [395](#)
  - ColRankProfileSubmatrix, [342](#), [377](#)
  - ColRankProfileSubmatrixIndices, [341](#), [377](#)
  - ColumnEchelonForm, [322](#), [323](#), [371](#)
  - ColumnRankProfile, [338](#), [339](#), [376](#)
  - composePermutationsLLL, [361](#)
  - composePermutationsLLM, [362](#)
  - composePermutationsMLM, [362](#)
  - cyclic\_shift\_col, [363](#), [366](#)
  - cyclic\_shift\_mathPerm, [362](#)
  - cyclic\_shift\_row, [363](#), [366](#)
  - cyclic\_shift\_row\_col, [363](#), [366](#)
  - Danilevski, [351](#)
  - Det, [334](#), [352](#), [374](#), [375](#)
  - doApplyS, [358](#)
  - doApplyT, [360](#)
  - failure, [381](#)
  - fflas\_const\_cast, [366](#), [380](#)
  - fgesv, [314](#), [315](#), [368](#)
  - fgetrs, [313](#), [367](#)
  - ForceCheck\_charpoly, [310](#)
  - ForceCheck\_Det, [309](#)
  - ForceCheck\_invert, [309](#)
  - ForceCheck\_PLUQ, [309](#)
  - fsytrf, [318](#), [319](#)
  - fsytrf\_BC\_Crout, [353](#)
  - fsytrf\_BC\_RL, [353](#)
  - fsytrf\_LOW\_RPM\_BC\_Crout, [353](#)
  - fsytrf\_nonunit, [319](#), [354](#)
  - fsytrf\_RPM, [355](#)
  - fsytrf\_UP\_RPM, [354](#)
  - fsytrf\_UP\_RPM\_BC\_Crout, [354](#)
  - fsytrf\_UP\_RPM\_BC\_RL, [353](#)
  - ftssyr2k, [317](#)
  - ftstr, [317](#)
  - fttrtri, [316](#), [368](#)
  - fttrrm, [316](#), [369](#)
  - getEchelonForm, [345](#)
  - getEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >, [378](#), [379](#)
  - getEchelonTransform, [346](#)
  - getEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >, [379](#)
  - getReducedEchelonForm, [347](#), [348](#)
  - getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >, [379](#), [380](#)
  - getReducedEchelonTransform, [348](#)
  - getReducedEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >, [380](#)
  - getTriangular, [343](#), [344](#)
  - getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >, [378](#)
  - getTridiagonal, [355](#)
  - Invert, [327](#), [328](#), [371](#)
  - Invert2, [328](#), [372](#)
  - isOdd, [381](#)
  - IsSingular, [333](#), [374](#)
  - KrylovElim, [373](#)
  - LAPACKPerm2MathPerm, [310](#)
  - LeadingSubmatrixRankProfiles, [340](#)
  - LQUPtoInverseOfFullRankMinor, [349](#), [380](#)
  - LUdivine, [321](#), [356](#), [369](#)
  - LUdivine\_gauss, [355](#), [370](#)
  - LUdivine\_small, [355](#), [369](#)
  - MathPerm2LAPACKPerm, [310](#)
  - MatrixApplyS, [358](#), [359](#)
  - MatrixApplyT, [360](#), [361](#)
  - MatVecMinPoly, [331](#), [373](#)
  - maxFieldElt, [394](#)
  - maxFieldElt< Givaro::ZRing< Givaro::Integer > >, [394](#)
  - MinPoly, [331](#), [373](#)
  - MonotonicApplyP, [312](#)
  - MonotonicCompress, [356](#)
  - MonotonicCompressCycles, [357](#)
  - MonotonicCompressMorePivots, [357](#)
  - MonotonicExpand, [357](#)
  - NonZeroRandomMatrix, [381](#), [382](#)
  - NullSpaceBasis, [336](#), [376](#)
  - pColumnEchelonForm, [323](#)
  - pColumnRankProfile, [339](#)
  - pDet, [334](#)
  - PermApplyS, [359](#)
  - PermApplyT, [361](#)
  - PLUQ, [320](#), [321](#), [365](#), [369](#)
  - PLUQ\_basecaseCrout, [364](#)
  - PLUQ\_basecaseV2, [364](#)
  - PLUQ\_basecaseV3, [364](#)
  - PLUQtoEchelonPermutation, [349](#)
  - pPLUQ, [320](#)
  - pRank, [332](#)
  - pReducedColumnEchelonForm, [325](#)

- pReducedRowEchelonForm, 326
- pRowEchelonForm, 324
- pRowRankProfile, 338
- pSolve, 336
- RandInt, 385
- RandomIndexSubset, 387
- RandomMatrix, 382, 383
- RandomMatrixWithDet, 393
- RandomMatrixWithRank, 385, 386
- RandomMatrixWithRankandRandomRPM, 391
- RandomMatrixWithRankandRPM, 388, 389
- RandomNullSpaceVector, 336, 350, 376
- RandomPermutation, 387
- RandomRankProfileMatrix, 387
- RandomSymmetricMatrix, 385
- RandomSymmetricMatrixWithRankandRandomRPM, 392
- RandomSymmetricMatrixWithRankandRPM, 390
- RandomSymmetricRankProfileMatrix, 388
- RandomTriangularMatrix, 383, 384
- Rank, 332, 333, 374
- RankProfileFromLU, 339
- ReducedColumnEchelonForm, 324, 325, 371
- ReducedRowEchelonForm, 326, 327, 370
- RowEchelonForm, 323, 324, 370
- RowRankProfile, 337, 338, 376
- RowRankProfileSubmatrix, 342, 377
- RowRankProfileSubmatrixIndices, 340, 377
- Solve, 335, 375
- solveLB, 351, 375
- solveLB2, 351, 375
- SpecRankProfile, 374
- swapval, 388
- threads\_fgemm, 365
- threads\_ftsm, 365
- trinv\_left, 316, 368
- ffpack-fgesv.C, 1121
  - main, 1121
- ffpack-solve.C, 1121
  - main, 1122
- ffpack.C, 988
  - applyP\_modular\_double, 993
  - ColRankProfileSubmatrix\_modular\_double, 1007
  - ColRankProfileSubmatrixIndices\_modular\_double, 1006
  - ColumnEchelonForm\_modular\_double, 996
  - ColumnEchelonForm\_modular\_float, 997
  - ColumnEchelonForm\_modular\_int32\_t, 998
  - ColumnRankProfile\_modular\_double, 1005
  - composePermutationsLLL, 993
  - composePermutationsLLM, 992
  - composePermutationsMLM, 993
  - cyclic\_shift\_col\_modular\_double, 993
  - cyclic\_shift\_mathPerm, 993
  - cyclic\_shift\_row\_modular\_double, 993
  - Det\_modular\_double, 1004
  - fgesv\_modular\_double, 995
  - fgesvin\_modular\_double, 994
  - fgetrsin\_modular\_double, 994
  - fgetrsv\_modular\_double, 994
  - fttrtri\_modular\_double, 995
  - fttrrm\_modular\_double, 995
  - getEchelonForm\_modular\_double, 1007
  - getEchelonFormin\_modular\_double, 1008
  - getEchelonTransform\_modular\_double, 1008
  - getReducedEchelonForm\_modular\_double, 1008
  - getReducedEchelonFormin\_modular\_double, 1009
  - getReducedEchelonTransform\_modular\_double, 1009
  - getTriangular\_modular\_double, 1007
  - getTriangularin\_modular\_double, 1007
  - Invert2\_modular\_double, 1003
  - Invert\_modular\_double, 1002
  - Invertin\_modular\_double, 1002
  - IsSingular\_modular\_double, 1003
  - KrylovElim\_modular\_double, 1003
  - LAPACKPerm2MathPerm, 991
  - LeadingSubmatrixRankProfiles, 1006
  - LUdivine\_modular\_double, 996
  - MathPerm2LAPACKPerm, 991
  - MatrixApplyS\_modular\_double, 991
  - MatrixApplyT\_modular\_double, 992
  - NullSpaceBasis\_modular\_double, 1005
  - pColumnEchelonForm\_modular\_double, 999
  - pColumnEchelonForm\_modular\_float, 1000
  - pColumnEchelonForm\_modular\_int32\_t, 1001
  - PermApplyS\_double, 992
  - PermApplyT\_double, 992
  - PLUQ\_modular\_double, 995
  - PLUQtoEchelonPermutation, 1009
  - pReducedColumnEchelonForm\_modular\_double, 1000
  - pReducedColumnEchelonForm\_modular\_float, 1001
  - pReducedColumnEchelonForm\_modular\_int32\_t, 1002
  - pReducedRowEchelonForm\_modular\_double, 1000
  - pReducedRowEchelonForm\_modular\_float, 1001
  - pReducedRowEchelonForm\_modular\_int32\_t, 1002
  - pRowEchelonForm\_modular\_double, 999
  - pRowEchelonForm\_modular\_float, 1000
  - pRowEchelonForm\_modular\_int32\_t, 1001
  - RandomNullSpaceVector\_modular\_double, 1005
  - Rank\_modular\_double, 1003
  - RankProfileFromLU, 1006
  - ReducedColumnEchelonForm\_modular\_double, 997
  - ReducedColumnEchelonForm\_modular\_float, 998
  - ReducedColumnEchelonForm\_modular\_int32\_t, 999
  - ReducedRowEchelonForm\_modular\_double, 997
  - ReducedRowEchelonForm\_modular\_float, 998
  - ReducedRowEchelonForm\_modular\_int32\_t, 999

- RowEchelonForm\_modular\_double, 996
- RowEchelonForm\_modular\_float, 997
- RowEchelonForm\_modular\_int32\_t, 998
- RowRankProfile\_modular\_double, 1005
- RowRankProfileSubmatrix\_modular\_double, 1006
- RowRankProfileSubmatrixIndices\_modular\_double, 1006
- Solve\_modular\_double, 1004
- solveLB2\_modular\_double, 1004
- solveLB\_modular\_double, 1004
- SpecRankProfile\_modular\_double, 1003
- trinv\_left\_modular\_double, 995
- ffpack.dox, 916
- ffpack.h, 916
  - \_\_FFLASFFPACK\_FTRSSYR2K\_THRESHOLD, 924
  - \_\_FFLASFFPACK\_FTRSTR\_THRESHOLD, 924
- ffpack.inl, 924
  - \_\_FFLASFFPACK\_ffpack\_INL, 925
- FFPACK::Protected, 395
  - ArithProg, 399
  - CompressRows, 401
  - CompressRowsQA, 402
  - CompressRowsQK, 401
  - Danilevski, 399
  - DeCompressRows, 401
  - DeCompressRowsQA, 402
  - DeCompressRowsQK, 401
  - fgemv\_kgf, 398
  - GaussJordan, 397
  - Hybrid\_KGF\_LUK\_MinPoly, 400
  - KellerGehrig, 398
  - KGFast, 398
  - KGFast\_generalized, 398
  - LUdivine\_construct, 397, 402
  - LUKrylov, 399
  - LUKrylov\_KGFast, 400
  - MatVecMinPoly, 400
  - newD, 401
  - RandomKrylovPrecond, 399
  - updatedD, 400
- ffpack\_c.h, 1009
  - applyP\_modular\_double, 1017
  - ColRankProfileSubmatrix\_modular\_double, 1028
  - ColRankProfileSubmatrixIndices\_modular\_double, 1028
  - ColumnEchelonForm\_modular\_double, 1020
  - ColumnEchelonForm\_modular\_float, 1020
  - ColumnEchelonForm\_modular\_int32\_t, 1021
  - ColumnRankProfile\_modular\_double, 1027
  - composePermutationsLLL, 1016
  - composePermutationsLLM, 1016
  - composePermutationsMLM, 1016
  - cyclic\_shift\_col\_modular\_double, 1017
  - cyclic\_shift\_mathPerm, 1016
  - cyclic\_shift\_row\_modular\_double, 1016
  - Det\_modular\_double, 1025
  - FFLAS\_C\_DIAG, 1013
  - FFLAS\_C\_ORDER, 1013
  - FFLAS\_C\_SIDE, 1014
  - FFLAS\_C\_TRANSPOSE, 1013
  - FFLAS\_C\_UPLO, 1013
  - FflasColMajor, 1013
  - FflasLeft, 1014
  - FflasLower, 1013
  - FflasNonUnit, 1014
  - FflasNoTrans, 1013
  - FflasRight, 1014
  - FflasRowMajor, 1013
  - FflasTrans, 1013
  - FflasUnit, 1014
  - FflasUpper, 1013
  - FFPACK\_C\_CHARPOLY\_TAG, 1014
  - FFPACK\_C\_LU\_TAG, 1014
  - FFPACK\_C\_MINPOLY\_TAG, 1014
  - FFPACK\_COMPILED, 1013
  - FfpackArithProg, 1014
  - FfpackDanilevski, 1014
  - FfpackDense, 1015
  - FfpackHybrid, 1014
  - FfpackKG, 1014
  - FfpackKGF, 1015
  - FfpackKGFast, 1014
  - FfpackKGFastG, 1014
  - FfpackLUK, 1014
  - FfpackSingular, 1014
  - FfpackSlabRecursive, 1014
  - FfpackTileRecursive, 1014
  - fgesv\_modular\_double, 1018
  - fgesvin\_modular\_double, 1018
  - fgetrs\_modular\_double, 1017
  - fgetrsin\_modular\_double, 1017
  - fttrtri\_modular\_double, 1018
  - fttrrm\_modular\_double, 1019
  - getEchelonForm\_modular\_double, 1029
  - getEchelonFormin\_modular\_double, 1029
  - getEchelonTransform\_modular\_double, 1029
  - getReducedEchelonForm\_modular\_double, 1030
  - getReducedEchelonFormin\_modular\_double, 1030
  - getReducedEchelonTransform\_modular\_double, 1030
  - getTriangular\_modular\_double, 1028
  - getTriangularin\_modular\_double, 1028
  - Invert2\_modular\_double, 1024
  - Invert\_modular\_double, 1024
  - Invertin\_modular\_double, 1023
  - IsSingular\_modular\_double, 1025
  - KrylovElim\_modular\_double, 1024
  - LAPACKPerm2MathPerm, 1015
  - LeadingSubmatrixRankProfiles, 1027
  - LUdivine\_gauss\_modular\_double, 1020
  - LUdivine\_modular\_double, 1019
  - LUdivine\_small\_modular\_double, 1019
  - MathPerm2LAPACKPerm, 1015
  - MatrixApplyS\_modular\_double, 1015

- MatrixApplyT\_modular\_double, [1015](#)
- NullSpaceBasis\_modular\_double, [1026](#)
- PermApplyS\_double, [1015](#)
- PermApplyT\_double, [1016](#)
- PLUQ\_modular\_double, [1019](#)
- PLUQtoEchelonPermutation, [1030](#)
- RandomNullSpaceVector\_modular\_double, [1026](#)
- Rank\_modular\_double, [1025](#)
- RankProfileFromLU, [1027](#)
- ReducedColumnEchelonForm\_modular\_double, [1021](#)
- ReducedColumnEchelonForm\_modular\_float, [1022](#)
- ReducedColumnEchelonForm\_modular\_int32\_t, [1022](#)
- ReducedRowEchelonForm2\_modular\_double, [1023](#)
- ReducedRowEchelonForm\_modular\_double, [1022](#)
- ReducedRowEchelonForm\_modular\_float, [1022](#)
- ReducedRowEchelonForm\_modular\_int32\_t, [1023](#)
- REF\_modular\_double, [1023](#)
- RowEchelonForm\_modular\_double, [1020](#)
- RowEchelonForm\_modular\_float, [1021](#)
- RowEchelonForm\_modular\_int32\_t, [1021](#)
- RowRankProfile\_modular\_double, [1026](#)
- RowRankProfileSubmatrix\_modular\_double, [1028](#)
- RowRankProfileSubmatrixIndices\_modular\_double, [1027](#)
- Solve\_modular\_double, [1025](#)
- solveLB2\_modular\_double, [1026](#)
- solveLB\_modular\_double, [1025](#)
- SpecRankProfile\_modular\_double, [1024](#)
- trinv\_left\_modular\_double, [1019](#)
- FFPACK\_C\_CHARPOLY\_TAG
  - ffpack\_c.h, [1014](#)
- FFPACK\_C\_LU\_TAG
  - ffpack\_c.h, [1014](#)
- FFPACK\_C\_MINPOLY\_TAG
  - ffpack\_c.h, [1014](#)
- ffpack\_charpoly.inl, [925](#)
  - \_\_FFLASFFPACK\_charpoly\_INL, [926](#)
- ffpack\_charpoly\_danilevski.inl, [926](#)
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_danilveski\_INL, [926](#)
- ffpack\_charpoly\_kgfast.inl, [927](#)
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfast\_INL, [927](#)
- ffpack\_charpoly\_kgfastgeneralized.inl, [927](#)
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_kgfastgeneralized\_INL, [928](#)
- ffpack\_charpoly\_kglu.inl, [928](#)
  - \_\_FFLASFFPACK\_ffpack\_charpoly\_kglu\_INL, [928](#)
- ffpack\_charpoly\_mp.inl, [928](#)
  - \_\_FFPACK\_charpoly\_mp\_INL, [929](#)
- FFPACK\_COMPILED
  - ffpack\_c.h, [1013](#)
- ffpack\_det\_mp.inl, [929](#)
  - \_\_FFPACK\_det\_mp\_INL, [929](#)
- ffpack\_echelonforms.inl, [930](#)
  - \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE, [931](#)
  - \_\_FFLASFFPACK\_ffpack\_echelon\_forms\_INL, [931](#)
- ffpack\_fgesv.inl, [931](#)
  - \_\_FFLASFFPACK\_ffpack\_fgesv\_INL, [931](#)
- ffpack\_fgetrs.inl, [932](#)
  - \_\_FFLASFFPACK\_ffpack\_fgetrs\_INL, [932](#)
- ffpack\_frobenius.inl, [932](#)
- ffpack\_fsytrf.inl, [933](#)
  - \_\_FFLASFFPACK\_ffpack\_fsytrf\_INL, [934](#)
- ffpack\_ftrssyr2k.inl, [934](#)
  - \_\_FFLASFFPACK\_ffpack\_ftrssyr2k\_INL, [934](#)
- ffpack\_ftrstr.inl, [935](#)
  - \_\_FFLASFFPACK\_ffpack\_ftrstr\_INL, [935](#)
- ffpack\_ftrtr.inl, [935](#)
  - \_\_FFLASFFPACK\_ffpack\_ftrtr\_INL, [936](#)
- ENABLE\_ALL\_CHECKINGS, [936](#)
- ffpack\_inst.C, [1031](#)
  - \_\_FFPACK\_INST\_C, [1031](#)
  - FFLAS\_COMPILED, [1031](#)
  - FFLAS\_ELT, [1031](#), [1032](#)
  - FFLAS\_FIELD, [1031](#), [1032](#)
  - INST\_OR\_DECL, [1031](#)
- ffpack\_inst.h, [1032](#)
  - FFLAS\_COMPILED, [1032](#)
  - FFLAS\_ELT, [1033](#)
  - FFLAS\_FIELD, [1033](#)
  - INST\_OR\_DECL, [1032](#)
- ffpack\_inst\_implem.inl, [1033](#)
- ffpack\_invert.inl, [936](#)
  - \_\_FFLASFFPACK\_ffpack\_invert\_INL, [936](#)
- ffpack\_krylovelim.inl, [936](#)
  - \_\_FFLASFFPACK\_ffpack\_krylovelim\_INL, [937](#)
- ffpack\_ludivine.inl, [937](#)
  - \_\_FFLASFFPACK\_ffpack\_ludivine\_INL, [937](#)
- ffpack\_ludivine\_mp.inl, [938](#)
  - \_\_FFPACK\_ludivine\_mp\_INL, [938](#)
- ffpack\_minpoly.inl, [938](#)
  - \_\_FFLASFFPACK\_ffpack\_minpoly\_INL, [939](#)
- ffpack\_permutation.inl, [939](#)
  - \_\_FFLASFFPACK\_ffpack\_permutation\_INL, [941](#)
  - FFLASFFPACK\_PERM\_BKSIZE, [941](#)
- ffpack\_pluq.inl, [941](#)
  - \_\_FFLASFFPACK\_ffpack\_pluq\_INL, [942](#)
  - CROUT, [942](#)
- ffpack\_pluq\_mp.inl, [942](#)
  - \_\_FFPACK\_pluq\_mp\_INL, [943](#)
- ffpack\_ppluq.inl, [943](#)
  - \_\_FFLASFFPACK\_ffpack\_ppluq\_INL, [943](#)
  - \_\_FFLAS\_TRSM\_READONLY, [943](#)
  - PBASECASE\_K, [943](#)
- ffpack\_rankprofiles.inl, [944](#)
  - \_\_FFLASFFPACK\_ffpack\_rank\_profiles\_INL, [945](#)
- FpackArithProg
  - ffpack\_c.h, [1014](#)
- FpackDanilevski



- ffpack\_c.h, [1014](#)
- FpackDense
  - ffpack\_c.h, [1015](#)
- FpackHybrid
  - ffpack\_c.h, [1014](#)
- FpackKG
  - ffpack\_c.h, [1014](#)
- FpackKGF
  - ffpack\_c.h, [1015](#)
- FpackKGFast
  - ffpack\_c.h, [1014](#)
- FpackKGFastG
  - ffpack\_c.h, [1014](#)
- FpackLUK
  - ffpack\_c.h, [1014](#)
- FpackSingular
  - ffpack\_c.h, [1014](#)
- FpackSlabRecursive
  - ffpack\_c.h, [1014](#)
- FpackTileRecursive
  - ffpack\_c.h, [1014](#)
- fgemm
  - FFLAS, [97–99](#), [101–106](#), [149](#), [185](#), [186](#)
- fgemm\_3\_modular\_double
  - fflas\_c.h, [964](#)
  - fflas\_lvl3.C, [987](#)
- fgemm\_classical.inl, [838](#)
  - \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_classical\_INL, [838](#)
- fgemm\_classical\_mp.inl, [838](#)
  - \_\_FFPACK\_fgemm\_classical\_INL, [840](#)
- fgemm\_convert
  - FFLAS::Protected, [220](#)
- fgemm\_winograd.inl, [840](#)
  - \_\_FFLASFFPACK\_fflas\_fflas\_fgemm\_winograd\_INL, [841](#)
  - NEWWINO, [841](#)
- fgemv
  - FFLAS, [106–112](#), [180](#), [193](#)
- fgemv\_2\_modular\_double
  - fflas\_c.h, [962](#)
  - fflas\_lvl2.C, [985](#)
- fgemv\_convert
  - FFLAS::Protected, [225](#)
- fgemv\_kgf
  - FFPACK::Protected, [398](#)
- fger
  - FFLAS, [112–116](#), [182](#)
- fger\_2\_modular\_double
  - fflas\_c.h, [963](#)
  - fflas\_lvl2.C, [985](#)
- fger\_convert
  - FFLAS::Protected, [226](#)
- fgesv
  - FFPACK, [314](#), [315](#), [368](#)
- fgesv\_modular\_double
  - ffpack.C, [995](#)
  - ffpack\_c.h, [1018](#)
- fgesvin\_modular\_double
  - ffpack.C, [994](#)
  - ffpack\_c.h, [1018](#)
- fgetrs
  - FFPACK, [313](#), [367](#)
- fgetrs\_modular\_double
  - ffpack\_c.h, [1017](#)
- fgetrsin\_modular\_double
  - ffpack.C, [994](#)
  - ffpack\_c.h, [1017](#)
- fgetrsv\_modular\_double
  - ffpack.C, [994](#)
- fidentity
  - FFLAS, [142](#), [173](#)
- fidentity\_2\_modular\_double
  - fflas\_c.h, [959](#)
  - fflas\_lvl2.C, [982](#)
- Field
  - benchmark-fgemm-rns.C, [788](#)
  - benchmark-pluq.C, [802](#)
  - FieldSimd< \_Field >, [445](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [740](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [742](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [746](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [751](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [752](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [754](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [756](#)
  - test-compressQ.C, [1068](#)
- field
  - associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > >, [408](#)
  - associatedDelayedField< const Givaro::Modular< T, X > >, [408](#)
  - associatedDelayedField< const Givaro::ModularBalanced< T > >, [409](#)
  - associatedDelayedField< const Givaro::ZRing< T > >, [409](#)
  - associatedDelayedField< Field >, [407](#)
- field-traits.h, [945](#)
- field.doxy, [947](#)
- FieldMax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [502](#)
- FieldMin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [502](#)
- FieldSimd
  - FieldSimd< \_Field >, [445](#)
- FieldSimd< \_Field >, [444](#)
  - add, [446](#)
  - add\_r, [446](#)
  - addin, [446](#)
  - addin\_r, [447](#)

- alignment, [450](#)
- axpy, [448](#), [449](#)
- axpy\_r, [449](#)
- axpyin, [449](#)
- axpyin\_r, [449](#)
- Element, [445](#)
- Field, [445](#)
- FieldSimd, [445](#)
- init, [446](#)
- maxpy, [449](#)
- maxpyin, [450](#)
- mod, [448](#)
- mul, [448](#)
- mul\_r, [448](#)
- mulin, [448](#)
- operator=, [446](#)
- scalar\_t, [445](#)
- simd, [445](#)
- sub, [447](#)
- sub\_r, [447](#)
- subin, [447](#)
- subin\_r, [447](#)
- vect\_size, [450](#)
- vect\_t, [445](#)
- zero, [447](#), [448](#)
- FieldTraits< FFPACK::RNSInteger< T > >, [451](#)
  - balanced, [451](#)
  - category, [451](#)
- FieldTraits< FFPACK::RNSIntegerMod< T > >, [451](#)
  - balanced, [452](#)
  - category, [451](#)
- FieldTraits< Field >, [450](#)
  - balanced, [450](#)
  - category, [450](#)
- FieldTraits< Givaro::Modular< Element > >, [452](#)
  - balanced, [452](#)
  - category, [452](#)
- FieldTraits< Givaro::ModularBalanced< Element > >, [452](#)
  - balanced, [453](#)
  - category, [453](#)
- FieldTraits< Givaro::ZRing< double > >, [453](#)
  - balanced, [453](#)
  - category, [453](#)
- FieldTraits< Givaro::ZRing< float > >, [453](#)
  - balanced, [454](#)
  - category, [454](#)
- FieldTraits< Givaro::ZRing< Givaro::Integer > >, [454](#)
  - balanced, [454](#)
  - category, [454](#)
- FieldTraits< Givaro::ZRing< int16\_t > >, [455](#)
  - balanced, [455](#)
  - category, [455](#)
- FieldTraits< Givaro::ZRing< int32\_t > >, [455](#)
  - balanced, [455](#)
  - category, [455](#)
- FieldTraits< Givaro::ZRing< int64\_t > >, [456](#)
  - balanced, [456](#)
  - category, [456](#)
- FieldTraits< Givaro::ZRing< RecInt::ruint< K > > >, [456](#)
  - balanced, [457](#)
  - category, [456](#)
- FieldTraits< Givaro::ZRing< uint16\_t > >, [457](#)
  - balanced, [457](#)
  - category, [457](#)
- FieldTraits< Givaro::ZRing< uint32\_t > >, [457](#)
  - balanced, [458](#)
  - category, [457](#)
- FieldTraits< Givaro::ZRing< uint64\_t > >, [458](#)
  - balanced, [458](#)
  - category, [458](#)
- fill\_value
  - benchmark-fgemv.C, [792](#)
- findArgument
  - args-parser.h, [1049](#)
- finit
  - FFLAS, [117](#), [119](#), [120](#), [135](#), [142](#), [163](#), [174](#)
- finit\_rns
  - FFLAS, [160](#), [162](#)
- finit\_trans\_rns
  - FFLAS, [161](#)
- first\_component
  - Compose< H1, H2 >, [422](#)
- firstBlockSize
  - ForStrategy1D< blocksize\_t, Cut, Param >, [460](#)
- fiszero
  - FFLAS, [138](#), [141](#), [166](#), [172](#)
- fiszero\_1\_modular\_double
  - fflas\_c.h, [956](#)
  - fflas\_lvl1.C, [977](#)
- fiszero\_2\_modular\_double
  - fflas\_c.h, [959](#)
  - fflas\_lvl2.C, [982](#)
- Fixed, [458](#)
- FixedPreIntTag, [458](#)
- flimits.h, [1055](#)
  - in\_range, [1055](#), [1056](#)
- FLOAT\_MOD
  - fflas\_simd.h, [874](#)
- Floats
  - benchmark-dgemm.C, [780](#)
- floor
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [558](#)
  - Simd256\_impl< true, false, true, 8 >, [632](#)
  - Simd512\_impl< true, false, true, 8 >, [712](#)
- fmaddd
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [559](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [563](#)
  - Simd128\_impl< true, true, false, 2 >, [575](#)
  - Simd128\_impl< true, true, false, 4 >, [585](#)



[illegible]

- Simd512\_impl< true, false, true, 8 >, [711](#)
- Simd512\_impl< true, true, false, 8 >, [722](#)
- Simd512\_impl< true, true, true, 8 >, [731](#)
- fmsubx
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [564](#)
  - Simd128\_impl< true, true, false, 2 >, [572](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 2 >, [603](#)
  - Simd128\_impl< true, true, true, 4 >, [612](#)
  - Simd128\_impl< true, true, true, 8 >, [621](#)
  - Simd256\_impl< true, true, false, 2 >, [638](#)
  - Simd256\_impl< true, true, false, 4 >, [649](#), [652](#)
  - Simd256\_impl< true, true, false, 8 >, [665](#)
  - Simd256\_impl< true, true, true, 2 >, [677](#)
  - Simd256\_impl< true, true, true, 4 >, [688](#), [693](#)
  - Simd256\_impl< true, true, true, 8 >, [702](#)
  - Simd512\_impl< true, true, false, 8 >, [718](#)
  - Simd512\_impl< true, true, true, 8 >, [731](#)
- fmsubxin
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [564](#)
  - Simd128\_impl< true, true, false, 2 >, [572](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [592](#), [595](#)
  - Simd128\_impl< true, true, true, 2 >, [604](#)
  - Simd128\_impl< true, true, true, 4 >, [612](#)
  - Simd128\_impl< true, true, true, 8 >, [621](#)
  - Simd256\_impl< true, true, false, 2 >, [638](#)
  - Simd256\_impl< true, true, false, 4 >, [649](#), [652](#)
  - Simd256\_impl< true, true, false, 8 >, [665](#)
  - Simd256\_impl< true, true, true, 2 >, [678](#)
  - Simd256\_impl< true, true, true, 4 >, [688](#), [693](#)
  - Simd256\_impl< true, true, true, 8 >, [702](#)
  - Simd512\_impl< true, true, false, 8 >, [718](#)
  - Simd512\_impl< true, true, true, 8 >, [731](#)
- fneg
  - FFLAS, [136](#), [144](#), [165](#), [175](#)
- fneg\_1\_modular\_double
  - fflas\_c.h, [956](#)
  - fflas\_lvl1.C, [977](#)
- fneg\_2\_modular\_double
  - fflas\_c.h, [960](#)
  - fflas\_lvl2.C, [983](#)
- fnegin
  - FFLAS, [136](#), [143](#), [164](#), [175](#)
- fnegin\_1\_modular\_double
  - fflas\_c.h, [956](#)
  - fflas\_lvl1.C, [977](#)
- fnegin\_2\_modular\_double
  - fflas\_c.h, [960](#)
  - fflas\_lvl2.C, [982](#)
- fnmadd
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [559](#)
- ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [564](#)
- Simd128\_impl< true, true, false, 2 >, [575](#)
- Simd128\_impl< true, true, false, 4 >, [585](#)
- Simd128\_impl< true, true, false, 8 >, [595](#)
- Simd128\_impl< true, true, true, 2 >, [603](#)
- Simd128\_impl< true, true, true, 4 >, [611](#)
- Simd128\_impl< true, true, true, 8 >, [620](#)
- Simd256\_impl< true, false, true, 8 >, [630](#)
- Simd256\_impl< true, true, false, 2 >, [641](#)
- Simd256\_impl< true, true, false, 4 >, [657](#), [658](#)
- Simd256\_impl< true, true, false, 8 >, [669](#)
- Simd256\_impl< true, true, true, 2 >, [677](#)
- Simd256\_impl< true, true, true, 4 >, [687](#), [692](#)
- Simd256\_impl< true, true, true, 8 >, [701](#)
- Simd512\_impl< true, false, true, 8 >, [711](#)
- Simd512\_impl< true, true, false, 8 >, [721](#)
- Simd512\_impl< true, true, true, 8 >, [730](#)
- fnmaddin
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [559](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [564](#)
  - Simd128\_impl< true, true, false, 2 >, [575](#)
  - Simd128\_impl< true, true, false, 4 >, [585](#)
  - Simd128\_impl< true, true, false, 8 >, [595](#)
  - Simd128\_impl< true, true, true, 2 >, [603](#)
  - Simd128\_impl< true, true, true, 4 >, [611](#)
  - Simd128\_impl< true, true, true, 8 >, [620](#)
  - Simd256\_impl< true, false, true, 8 >, [630](#)
  - Simd256\_impl< true, true, false, 2 >, [642](#)
  - Simd256\_impl< true, true, false, 4 >, [658](#)
  - Simd256\_impl< true, true, false, 8 >, [669](#)
  - Simd256\_impl< true, true, true, 2 >, [677](#)
  - Simd256\_impl< true, true, true, 4 >, [687](#), [692](#)
  - Simd256\_impl< true, true, true, 8 >, [702](#)
  - Simd512\_impl< true, false, true, 8 >, [711](#)
  - Simd512\_impl< true, true, false, 8 >, [721](#)
  - Simd512\_impl< true, true, true, 8 >, [730](#)
- fnmaddx
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [564](#)
  - Simd128\_impl< true, true, false, 2 >, [572](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 2 >, [603](#)
  - Simd128\_impl< true, true, true, 4 >, [612](#)
  - Simd128\_impl< true, true, true, 8 >, [620](#)
  - Simd256\_impl< true, true, false, 2 >, [638](#)
  - Simd256\_impl< true, true, false, 4 >, [649](#), [652](#)
  - Simd256\_impl< true, true, false, 8 >, [665](#)
  - Simd256\_impl< true, true, true, 2 >, [677](#)
  - Simd256\_impl< true, true, true, 4 >, [687](#), [693](#)
  - Simd256\_impl< true, true, true, 8 >, [702](#)
  - Simd512\_impl< true, true, false, 8 >, [717](#)
  - Simd512\_impl< true, true, true, 8 >, [730](#)
- fnmaddxin

- ScalFunctions< Element, typename enable\_if<  
is\_integral< Element >::value >::type >, 564
- Simd128\_impl< true, true, false, 2 >, 572
- Simd128\_impl< true, true, false, 4 >, 582
- Simd128\_impl< true, true, false, 8 >, 591
- Simd128\_impl< true, true, true, 2 >, 603
- Simd128\_impl< true, true, true, 4 >, 612
- Simd128\_impl< true, true, true, 8 >, 620
- Simd256\_impl< true, true, false, 2 >, 638
- Simd256\_impl< true, true, false, 4 >, 649, 652
- Simd256\_impl< true, true, false, 8 >, 665
- Simd256\_impl< true, true, true, 2 >, 677
- Simd256\_impl< true, true, true, 4 >, 687, 693
- Simd256\_impl< true, true, true, 8 >, 702
- Simd512\_impl< true, true, false, 8 >, 717
- Simd512\_impl< true, true, true, 8 >, 730
- FOR1D
  - parallel.h, 1041
- FOR2D
  - parallel.h, 1042
- FORBLOCK1D
  - parallel.h, 1041
- FORBLOCK2D
  - parallel.h, 1042
- ForceCheck\_charpoly
  - FFPACK, 310
- ForceCheck\_Det
  - FFPACK, 309
- ForceCheck\_fgemm
  - FFLAS, 78
- ForceCheck\_ftsm
  - FFLAS, 78
- ForceCheck\_invert
  - FFPACK, 309
- ForceCheck\_PLUQ
  - FFPACK, 309
- ForStrategy1D
  - ForStrategy1D< blocksize\_t, Cut, Param >, 459
- ForStrategy1D< blocksize\_t, Cut, Param >, 459
  - begin, 460
  - blockindex, 460
  - build, 459
  - changeBS, 461
  - current, 460
  - end, 460
  - firstBlockSize, 460
  - ForStrategy1D, 459
  - ibeg, 460
  - iend, 460
  - initialize, 460
  - isTerminated, 460
  - lastBlockSize, 461
  - numBlock, 461
  - numblocks, 460
  - operator++, 460
- ForStrategy2D
  - ForStrategy2D< blocksize\_t, Cut, Param >, 462
- ForStrategy2D< blocksize\_t, Cut, Param >, 461
  - \_ibeg, 463
  - \_iend, 463
  - \_jbeg, 463
  - \_jend, 463
  - blockindex, 463
  - BLOCKS, 464
  - changeCBS, 464
  - changeRBS, 464
  - colblockindex, 463
  - colBlockSize, 463
  - colnumblocks, 462
  - current, 464
  - ForStrategy2D, 462
  - ibegin, 462
  - iend, 462
  - initialize, 462
  - isTerminated, 462
  - jbegin, 462
  - jend, 462
  - lastCBS, 464
  - lastRBS, 464
  - numColBlock, 464
  - numRowBlock, 464
  - operator<<, 463
  - operator++, 462
  - rowblockindex, 463
  - rowBlockSize, 463
  - rownumblocks, 462
- frand
  - FFLAS, 137, 140
- freduce
  - FFLAS, 116–118, 120, 162, 163, 173, 174
  - FFLAS::details, 209
- freduce\_1\_modular\_double
  - fflas\_c.h, 955
  - fflas\_lvl1.C, 977
- freduce\_2\_modular\_double
  - fflas\_c.h, 960
  - fflas\_lvl2.C, 982
- freduce\_constoverride
  - FFLAS, 117, 119
- freducein\_1\_modular\_double
  - fflas\_c.h, 955
  - fflas\_lvl1.C, 976
- freducein\_2\_modular\_double
  - fflas\_c.h, 960
  - fflas\_lvl2.C, 982
- freivalds
  - FFLAS, 120
- fscal
  - FFLAS, 122–126, 167, 176
  - FFLAS::details, 210
- fscal\_1\_modular\_double
  - fflas\_c.h, 957
  - fflas\_lvl1.C, 978
- fscal\_2\_modular\_double
  - fflas\_c.h, 961
  - fflas\_lvl2.C, 983

- fscalin
  - FFLAS, [121](#), [123–126](#), [167](#), [176](#)
  - FFLAS::details, [210](#)
- fscalin\_1\_modular\_double
  - fflas\_c.h, [957](#)
  - fflas\_lvl1.C, [978](#)
- fscalin\_2\_modular\_double
  - fflas\_c.h, [960](#)
  - fflas\_lvl2.C, [983](#)
- fspmm
  - FFLAS, [150](#)
  - FFLAS::sparse\_details, [236–238](#)
  - FFLAS::sparse\_details\_impl, [252](#), [253](#), [260](#), [261](#), [267](#), [271](#), [272](#), [281](#)
- fspmm\_dispatch
  - FFLAS::sparse\_details, [236](#)
- fspmm\_mone
  - FFLAS::sparse\_details\_impl, [254](#), [262](#), [272](#), [273](#)
- fspmm\_mone\_simd\_aligned
  - FFLAS::sparse\_details\_impl, [255](#), [262](#), [274](#)
- fspmm\_mone\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, [255](#), [263](#), [274](#)
- fspmm\_one
  - FFLAS::sparse\_details\_impl, [254](#), [261](#), [272](#), [273](#)
- fspmm\_one\_simd\_aligned
  - FFLAS::sparse\_details\_impl, [254](#), [262](#), [273](#)
- fspmm\_one\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, [254](#), [262](#), [273](#)
- fspmm\_simd\_aligned
  - FFLAS::sparse\_details\_impl, [253](#), [261](#)
- fspmm\_simd\_unaligned
  - FFLAS::sparse\_details\_impl, [253](#), [261](#)
- fspmv
  - FFLAS, [150](#), [158](#)
  - FFLAS::sparse\_details, [233–235](#), [243](#), [244](#)
  - FFLAS::sparse\_details\_impl, [255](#), [256](#), [263](#), [267](#), [268](#), [274](#), [275](#), [277](#), [278](#), [281–284](#)
- fspmv\_dispatch
  - FFLAS::sparse\_details, [233](#)
- fspmv\_mone
  - FFLAS::sparse\_details\_impl, [256](#), [264](#), [275](#), [278](#), [279](#), [285](#)
- fspmv\_mone\_simd
  - FFLAS::sparse\_details\_impl, [279](#), [285](#)
- fspmv\_one
  - FFLAS::sparse\_details\_impl, [256](#), [264](#), [275](#), [278](#), [284](#), [285](#)
- fspmv\_one\_simd
  - FFLAS::sparse\_details\_impl, [279](#), [285](#)
- fspmv\_simd
  - FFLAS::sparse\_details\_impl, [277](#), [278](#), [284](#)
- fsquare
  - FFLAS, [99–101](#), [187](#)
- fsquare\_3\_modular\_double
  - fflas\_c.h, [964](#)
  - fflas\_lvl3.C, [987](#)
- fsquareCommon
  - FFLAS::Protected, [223](#)
- fsub
  - FFLAS, [85](#), [89](#), [170](#), [178](#)
- fsub\_1\_modular\_double
  - fflas\_c.h, [958](#)
  - fflas\_lvl1.C, [979](#)
- fsub\_2\_modular\_double
  - fflas\_c.h, [962](#)
  - fflas\_lvl2.C, [984](#)
- fsubin
  - FFLAS, [85](#), [90](#), [179](#)
- fsubin\_1\_modular\_double
  - fflas\_c.h, [958](#)
  - fflas\_lvl1.C, [980](#)
- fsubin\_2\_modular\_double
  - fflas\_c.h, [962](#)
  - fflas\_lvl2.C, [984](#)
- fswap
  - FFLAS, [139](#), [169](#)
- fswap\_1\_modular\_double
  - fflas\_c.h, [957](#)
  - fflas\_lvl1.C, [979](#)
- fsyr2k
  - FFLAS, [126](#)
- fsyrk
  - FFLAS, [127–129](#)
- fsyrk.C, [771](#)
  - CUBE, [772](#)
  - GFOPS, [772](#)
  - main, [772](#)
  - Timer, [772](#)
- fsytrf
  - FFPACK, [318](#), [319](#)
- fsytrf.C, [772](#)
  - CUBE, [773](#)
  - GFOPS, [773](#)
  - main, [773](#)
  - Timer, [773](#)
- fsytrf\_BC\_Crout
  - FFPACK, [353](#)
- fsytrf\_BC\_RL
  - FFPACK, [353](#)
- fsytrf\_LOW\_RPM\_BC\_Crout
  - FFPACK, [353](#)
- fsytrf\_nonunit
  - FFPACK, [319](#), [354](#)
- fsytrf\_RPM
  - FFPACK, [355](#)
- fsytrf\_UP\_RPM
  - FFPACK, [354](#)
- fsytrf\_UP\_RPM\_BC\_Crout
  - FFPACK, [354](#)
- fsytrf\_UP\_RPM\_BC\_RL
  - FFPACK, [353](#)
- ftmm
  - FFLAS, [130](#), [131](#), [184](#)
- ftmm\_3\_modular\_double
  - fflas\_c.h, [963](#)
  - fflas\_lvl3.C, [987](#)

- ftmmLeftLowerNoTransNonUnit< Element >, 464
- ftmmLeftLowerNoTransUnit< Element >, 464
- ftmmLeftLowerTransNonUnit< Element >, 465
- ftmmLeftLowerTransUnit< Element >, 465
- ftmmLeftUpperNoTransNonUnit< Element >, 465
- ftmmLeftUpperNoTransUnit< Element >, 465
- ftmmLeftUpperTransNonUnit< Element >, 465
- ftmmLeftUpperTransUnit< Element >, 465
- ftmmRightLowerNoTransNonUnit< Element >, 465
- ftmmRightLowerNoTransUnit< Element >, 465
- ftmmRightLowerTransNonUnit< Element >, 466
- ftmmRightLowerTransUnit< Element >, 466
- ftmmRightUpperNoTransNonUnit< Element >, 466
- ftmmRightUpperNoTransUnit< Element >, 466
- ftmmRightUpperTransNonUnit< Element >, 466
- ftmmRightUpperTransUnit< Element >, 466
- ftmv
  - FFLAS, 146
- ftsm
  - FFLAS, 132, 133, 146, 149, 150, 183
- ftsm\_3\_modular\_double
  - fflas\_c.h, 963
  - fflas\_lvl3.C, 986
- ftsmLeftLowerNoTransNonUnit< Element >, 466
- ftsmLeftLowerNoTransUnit< Element >, 466
- ftsmLeftLowerTransNonUnit< Element >, 467
- ftsmLeftLowerTransUnit< Element >, 467
- ftsmLeftUpperNoTransNonUnit< Element >, 467
- ftsmLeftUpperNoTransUnit< Element >, 467
- ftsmLeftUpperTransNonUnit< Element >, 467
- ftsmLeftUpperTransUnit< Element >, 467
- ftsmRightLowerNoTransNonUnit< Element >, 468
- ftsmRightLowerNoTransUnit< Element >, 468
- ftsmRightLowerTransNonUnit< Element >, 468
- ftsmRightLowerTransUnit< Element >, 468
- ftsmRightUpperNoTransNonUnit< Element >, 468
- ftsmRightUpperNoTransUnit< Element >, 468
- ftsmRightUpperTransNonUnit< Element >, 468
- ftsmRightUpperTransUnit< Element >, 468
- ftssyr2k
  - FFPACK, 317
- ftstr
  - FFPACK, 317
- ftsv
  - FFLAS, 134, 182
- ftsv\_2\_modular\_double
  - fflas\_c.h, 963
  - fflas\_lvl2.C, 985
- fttri
  - FFPACK, 316, 368
- fttri.C, 773
  - CUBE, 774
  - GFOPS, 774
  - main, 774
  - TTimer, 774
- fttri\_modular\_double
  - ffpack.C, 995
  - ffpack\_c.h, 1018
- fttrm
  - FFPACK, 316, 369
- fttrm\_modular\_double
  - ffpack.C, 995
  - ffpack\_c.h, 1019
- fzero
  - FFLAS, 137, 140, 165, 171
- fzero\_1\_modular\_double
  - fflas\_c.h, 956
  - fflas\_lvl1.C, 977
- fzero\_2\_modular\_double
  - fflas\_c.h, 959
  - fflas\_lvl2.C, 981
- gather
  - Simd128\_impl< true, true, false, 2 >, 570, 573
  - Simd128\_impl< true, true, false, 4 >, 580, 583
  - Simd128\_impl< true, true, false, 8 >, 589, 592
  - Simd128\_impl< true, true, true, 2 >, 600
  - Simd128\_impl< true, true, true, 4 >, 608
  - Simd128\_impl< true, true, true, 8 >, 617
  - Simd256\_impl< true, false, true, 8 >, 628
  - Simd256\_impl< true, true, false, 2 >, 636, 639
  - Simd256\_impl< true, true, false, 4 >, 647, 650, 653
  - Simd256\_impl< true, true, false, 8 >, 663, 666
  - Simd256\_impl< true, true, true, 2 >, 673
  - Simd256\_impl< true, true, true, 4 >, 683, 690
  - Simd256\_impl< true, true, true, 8 >, 698
  - Simd512\_impl< true, false, true, 8 >, 708
  - Simd512\_impl< true, true, false, 8 >, 715, 719
  - Simd512\_impl< true, true, true, 8 >, 727
- GaussJordan
  - FFPACK::Protected, 397
- GCC\_VERSION
  - fflas-ffpack-config.h, 826
- gebp
  - FFLAS::details, 213
- genData
  - benchmark-fgemv.C, 792
- generate\_random\_vector
  - test-simd.C, 1115, 1116
- GenericTag, 469
- get
  - Simd128\_impl< true, true, false, 8 >, 592
  - Simd128\_impl< true, true, true, 8 >, 617
  - Simd256\_impl< true, true, false, 8 >, 666
  - Simd256\_impl< true, true, true, 8 >, 698
- getDataType
  - FFLAS, 157
- getEchelonForm
  - FFPACK, 345
- getEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, 378, 379
- getEchelonForm\_modular\_double
  - ffpack.C, 1007
  - ffpack\_c.h, 1029
- getEchelonFormin\_modular\_double
  - ffpack.C, 1008

- ffpack\_c.h, [1029](#)
- getEchelonTransform
  - FFPACK, [346](#)
- getEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [379](#)
- getEchelonTransform\_modular\_double
  - ffpack.C, [1008](#)
  - ffpack\_c.h, [1029](#)
- getListArgs
  - args-parser.h, [1049](#)
- getReducedEchelonForm
  - FFPACK, [347](#), [348](#)
- getReducedEchelonForm< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [379](#), [380](#)
- getReducedEchelonForm\_modular\_double
  - ffpack.C, [1008](#)
  - ffpack\_c.h, [1030](#)
- getReducedEchelonFormin\_modular\_double
  - ffpack.C, [1009](#)
  - ffpack\_c.h, [1030](#)
- getReducedEchelonTransform
  - FFPACK, [348](#)
- getReducedEchelonTransform< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [380](#)
- getReducedEchelonTransform\_modular\_double
  - ffpack.C, [1009](#)
  - ffpack\_c.h, [1030](#)
- getSeed
  - FFLAS, [198](#)
- getStat
  - FFLAS, [159](#)
- getTLBSize
  - FFLAS, [198](#)
- getTriangular
  - FFPACK, [343](#), [344](#)
- getTriangular< FFLAS\_FIELD< FFLAS\_ELT > >
  - FFPACK, [378](#)
- getTriangular\_modular\_double
  - ffpack.C, [1007](#)
  - ffpack\_c.h, [1028](#)
- getTriangularin\_modular\_double
  - ffpack.C, [1007](#)
  - ffpack\_c.h, [1028](#)
- getTridiagonal
  - FFPACK, [355](#)
- gf2ModularBalanced
  - regression-check.C, [1065](#)
- GFOPS
  - arithprog.C, [769](#)
  - autotune/charpoly.C, [770](#)
  - autotune/pluq.C, [775](#)
  - fsyrk.C, [772](#)
  - fsytrf.C, [773](#)
  - fttri.C, [774](#)
  - winograd.C, [776](#)
- Givaro, [403](#)
- gmp.C, [1060](#)
  - main, [1060](#)
- GRAIN
  - benchmark-fgemm-rns.C, [788](#)
- Grain, [469](#)
- greater
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [560](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [565](#)
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [581](#)
  - Simd128\_impl< true, true, false, 8 >, [590](#)
  - Simd128\_impl< true, true, true, 2 >, [604](#)
  - Simd128\_impl< true, true, true, 4 >, [613](#)
  - Simd128\_impl< true, true, true, 8 >, [621](#)
  - Simd256\_impl< true, false, true, 8 >, [631](#)
  - Simd256\_impl< true, true, false, 2 >, [637](#)
  - Simd256\_impl< true, true, false, 4 >, [648](#), [651](#)
  - Simd256\_impl< true, true, false, 8 >, [664](#)
  - Simd256\_impl< true, true, true, 2 >, [678](#)
  - Simd256\_impl< true, true, true, 4 >, [688](#), [694](#)
  - Simd256\_impl< true, true, true, 8 >, [703](#)
  - Simd512\_impl< true, false, true, 8 >, [711](#)
  - Simd512\_impl< true, true, false, 8 >, [716](#)
  - Simd512\_impl< true, true, true, 8 >, [731](#)
- greater\_eq
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [560](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [565](#)
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [581](#)
  - Simd128\_impl< true, true, false, 8 >, [590](#)
  - Simd128\_impl< true, true, true, 2 >, [604](#)
  - Simd128\_impl< true, true, true, 4 >, [613](#)
  - Simd128\_impl< true, true, true, 8 >, [621](#)
  - Simd256\_impl< true, false, true, 8 >, [631](#)
  - Simd256\_impl< true, true, false, 2 >, [637](#)
  - Simd256\_impl< true, true, false, 4 >, [648](#), [651](#)
  - Simd256\_impl< true, true, false, 8 >, [664](#)
  - Simd256\_impl< true, true, true, 2 >, [678](#)
  - Simd256\_impl< true, true, true, 4 >, [688](#), [694](#)
  - Simd256\_impl< true, true, true, 8 >, [703](#)
  - Simd512\_impl< true, false, true, 8 >, [712](#)
  - Simd512\_impl< true, true, false, 8 >, [717](#)
  - Simd512\_impl< true, true, true, 8 >, [731](#)
- hadd
  - Simd256\_impl< true, false, true, 8 >, [632](#)
  - Simd512\_impl< true, false, true, 8 >, [712](#)
- hadd\_to\_scal
  - Simd128\_impl< true, true, false, 2 >, [572](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [592](#)
  - Simd128\_impl< true, true, true, 2 >, [604](#)



- Simd128\_impl< true, true, true, 4 >, [613](#)
- Simd128\_impl< true, true, true, 8 >, [622](#)
- Simd256\_impl< true, false, true, 8 >, [632](#)
- Simd256\_impl< true, true, false, 2 >, [638](#)
- Simd256\_impl< true, true, false, 4 >, [649](#), [652](#)
- Simd256\_impl< true, true, false, 8 >, [665](#)
- Simd256\_impl< true, true, true, 2 >, [678](#)
- Simd256\_impl< true, true, true, 4 >, [689](#), [694](#)
- Simd256\_impl< true, true, true, 8 >, [703](#)
- Simd512\_impl< true, false, true, 8 >, [712](#)
- Simd512\_impl< true, true, false, 8 >, [718](#)
- Simd512\_impl< true, true, true, 8 >, [732](#)
- half\_t
  - Simd256\_impl< true, true, false, 2 >, [635](#)
  - Simd256\_impl< true, true, false, 4 >, [646](#), [647](#)
  - Simd256\_impl< true, true, false, 8 >, [663](#)
  - Simd256\_impl< true, true, true, 2 >, [672](#)
  - Simd256\_impl< true, true, true, 4 >, [682](#), [683](#)
  - Simd256\_impl< true, true, true, 8 >, [697](#)
  - Simd512\_impl< true, true, false, 8 >, [715](#)
  - Simd512\_impl< true, true, true, 8 >, [726](#)
- has\_equal
  - FFLAS, [80](#)
- has\_minus
  - FFLAS, [79](#)
- has\_minus\_eq
  - FFLAS, [80](#)
- has\_minus\_eq\_impl< C >, [469](#)
  - value, [469](#)
- has\_minus\_impl< C >, [469](#)
  - value, [470](#)
- has\_mul
  - FFLAS, [80](#)
- has\_mul\_eq
  - FFLAS, [80](#)
- has\_mul\_eq\_impl< C >, [470](#)
  - value, [470](#)
- has\_mul\_impl< C >, [470](#)
  - value, [470](#)
- has\_operation< T >, [470](#)
  - value, [471](#)
- has\_plus
  - FFLAS, [79](#)
- has\_plus\_eq
  - FFLAS, [80](#)
- has\_plus\_eq\_impl< C >, [471](#)
  - value, [471](#)
- has\_plus\_impl< C >, [471](#)
  - value, [471](#)
- hasAVX512ER
  - instrset.h, [1063](#)
  - instrset\_detect.cpp, [1064](#)
- hasF16C
  - instrset\_detect.cpp, [1063](#)
- hasFMA3
  - instrset.h, [1062](#)
  - instrset\_detect.cpp, [1063](#)
- hasFMA4
  - instrset.h, [1062](#)
  - instrset\_detect.cpp, [1063](#)
- hasXOP
  - instrset.h, [1063](#)
  - instrset\_detect.cpp, [1063](#)
- HAVE\_BLAS
  - config.h, [804](#)
- HAVE\_CBLAS
  - config.h, [804](#)
- HAVE\_CXX11
  - config.h, [804](#)
- HAVE\_DLFCN\_H
  - config.h, [805](#)
- HAVE\_FLOAT\_H
  - config.h, [805](#)
- HAVE\_INT128
  - config.h, [805](#)
- HAVE\_INTTYPES\_H
  - config.h, [805](#)
- HAVE\_LAPACK
  - config.h, [805](#)
- HAVE\_LIMITS\_H
  - config.h, [805](#)
- HAVE\_LITTLE\_ENDIAN
  - config.h, [805](#)
- HAVE\_PTHREAD\_H
  - config.h, [805](#)
- HAVE\_STDDEF\_H
  - config.h, [805](#)
- HAVE\_STDINT\_H
  - config.h, [805](#)
- HAVE\_STDIO\_H
  - config.h, [805](#)
- HAVE\_STDLIB\_H
  - config.h, [805](#)
- HAVE\_STRING\_H
  - config.h, [806](#)
- HAVE\_STRINGS\_H
  - config.h, [806](#)
- HAVE\_SYS\_STAT\_H
  - config.h, [806](#)
- HAVE\_SYS\_TIME\_H
  - config.h, [806](#)
- HAVE\_SYS\_TYPES\_H
  - config.h, [806](#)
- HAVE\_UNISTD\_H
  - config.h, [806](#)
- HelperFlag, [471](#)
  - aut, [472](#)
  - coo, [472](#)
  - csr, [472](#)
  - ell, [472](#)
  - none, [472](#)
  - pm1, [472](#)
- HelperMod
  - HelperMod< Field, ElementCategories::MachineIntTag >, [473](#)

HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecisionTag  
 >, 474  
 HelperMod< Field, FFLAS::ElementCategories::FixedPrecisionTag  
 >, 474  
 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag  
 >, 475  
 HelperMod< Field, ElementCategories::MachineIntTag  
 >, 472  
 HelperMod, 473  
 invp, 473  
 max, 473  
 min, 473  
 p, 473  
 pow50rem, 473  
 HelperMod< Field, ElementTraits >, 472  
 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecisionTag  
 >, 474  
 HelperMod, 474  
 p, 474  
 HelperMod< Field, FFLAS::ElementCategories::FixedPrecisionTag  
 >, 474  
 HelperMod, 474  
 p, 475  
 HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag  
 >, 475  
 HelperMod, 475  
 invp, 475  
 max, 476  
 min, 475  
 p, 475  
 helpString  
 Argument, 407  
 HYB\_ZO  
 FFLAS, 83  
 hyb\_zo.h, 907  
 hyb\_zo\_pspmm.inl, 907  
 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmm\_INL, 908  
 hyb\_zo\_pspmv.inl, 908  
 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_pspmv\_INL, 908  
 hyb\_zo\_spm.inl, 909  
 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spm\_INL, 909  
 hyb\_zo\_spmv.inl, 909  
 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_spmv\_INL, 909  
 hyb\_zo\_utils.inl, 910  
 \_\_FFLASFFPACK\_fflas\_sparse\_HYB\_ZO\_utils\_INL, 910  
 Hybrid, 476  
 Hybrid\_KGF\_LUK\_MinPoly  
 FFPACK::Protected, 400  
 ibeg  
 ForStrategy1D< blocksize\_t, Cut, Param >, 460  
 ibegin  
 ForStrategy2D< blocksize\_t, Cut, Param >, 462  
 idamax\_  
 iend  
 ForStrategy1D< blocksize\_t, Cut, Param >, 460  
 ForStrategy2D< blocksize\_t, Cut, Param >, 462  
 igebb14  
 FFLAS::details, 211  
 igebb21  
 FFLAS::details, 212  
 igebb24  
 FFLAS::details, 211  
 igebb41  
 FFLAS::details, 211  
 igebb44  
 FFLAS::details, 211  
 igebp  
 FFLAS::details, 212  
 igemm  
 FFLAS::Protected, 228  
 igemm.doxy, 860  
 igemm.h, 860  
 igemm.inl, 861  
 FFLASFFPACK\_fflas\_igemm\_igemm\_INL, 861  
 igemm\_  
 FFLAS, 134  
 igemm\_colmajor  
 FFLAS::Protected, 228  
 igemm\_kernels.h, 862  
 igemm\_kernels.inl, 862  
 \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_kernels\_INL, 863  
 igemm\_tools.h, 863  
 igemm\_tools.inl, 863  
 \_\_FFLASFFPACK\_fflas\_igemm\_igemm\_tools\_INL, 864  
 inl\_range  
 flimits.h, 1055, 1056  
 index\_t  
 fflas\_sparse.h, 886  
 parallel.h, 1040  
 InfNorm  
 FFLAS, 84  
 Info, 476, 477  
 begin, 477, 478  
 Info, 476, 478  
 operator=, 477, 478  
 perm, 477, 478  
 size, 477, 478  
 init  
 FieldSimd< \_Field >, 446  
 rns\_double, 529–531  
 rns\_double\_extended, 542, 543  
 RNSInteger< RNS >, 547  
 RNSIntegerMod< RNS >, 551, 552  
 init\_transpose  
 rns\_double, 530  
 init\_y



- FFLAS::sparse\_details, [232](#), [233](#)
- initA
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [500](#)
- initB
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [500](#)
- initC
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [500](#)
- initialize
  - ForStrategy1D< blocksize\_t, Cut, Param >, [460](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [462](#)
- initOut
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [500](#)
- INLINE
  - fflas\_simd.h, [874](#)
- INST\_OR\_DECL
  - fflas\_L1\_inst.C, [965](#)
  - fflas\_L1\_inst.h, [966](#)
  - fflas\_L2\_inst.C, [969](#)
  - fflas\_L2\_inst.h, [970](#)
  - fflas\_L3\_inst.C, [973](#)
  - fflas\_L3\_inst.h, [974](#)
  - ffpack\_inst.C, [1031](#)
  - ffpack\_inst.h, [1032](#)
- INSTRSET
  - instrset.h, [1061](#)
- instrset.h, [1060](#)
  - const\_int, [1061](#)
  - const\_uint, [1061](#)
  - hasAVX512ER, [1063](#)
  - hasFMA3, [1062](#)
  - hasFMA4, [1062](#)
  - hasXOP, [1063](#)
  - INSTRSET, [1061](#)
  - instrset\_detect, [1062](#)
  - INSTRSET\_H, [1061](#)
  - int16\_t, [1062](#)
  - int32\_t, [1062](#)
  - int64\_t, [1062](#)
  - int8\_t, [1061](#)
  - intptr\_t, [1062](#)
  - uint16\_t, [1062](#)
  - uint32\_t, [1062](#)
  - uint64\_t, [1062](#)
  - uint8\_t, [1062](#)
- instrset\_detect
  - instrset.h, [1062](#)
  - instrset\_detect.cpp, [1063](#)
- instrset\_detect.cpp, [1063](#)
  - hasAVX512ER, [1064](#)
  - hasF16C, [1063](#)
  - hasFMA3, [1063](#)
  - hasFMA4, [1063](#)
  - hasXOP, [1063](#)
  - instrset\_detect, [1063](#)
- INSTRSET\_H
  - instrset.h, [1061](#)
- int16\_t
  - instrset.h, [1062](#)
- int32\_t
  - instrset.h, [1062](#)
- int64\_t
  - instrset.h, [1062](#)
- int8\_t
  - instrset.h, [1061](#)
- integer
  - rns\_double, [528](#)
  - rns\_double\_extended, [541](#)
  - RNSInteger< RNS >, [546](#)
  - RNSIntegerMod< RNS >, [550](#)
  - test-simd.C, [1114](#)
- Interfaces, [51](#)
- interfaces.doxy, [951](#)
- intptr\_t
  - instrset.h, [1062](#)
- inv
  - RNSIntegerMod< RNS >, [553](#)
- Invert
  - FFPACK, [327](#), [328](#), [371](#)
- Invert2
  - FFPACK, [328](#), [372](#)
- Invert2\_modular\_double
  - ffpack.C, [1003](#)
  - ffpack\_c.h, [1024](#)
- Invert\_modular\_double
  - ffpack.C, [1002](#)
  - ffpack\_c.h, [1024](#)
- Invertin\_modular\_double
  - ffpack.C, [1002](#)
  - ffpack\_c.h, [1023](#)
- invp
  - HelperMod< Field, ElementCategories::MachineIntTag >, [473](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [475](#)
- is\_simd< T >, [479](#)
  - type, [479](#)
  - value, [479](#)
- isMOne
  - RNSInteger< RNS >, [547](#)
  - RNSIntegerMod< RNS >, [551](#)
- isOdd
  - FFPACK, [381](#)
- isOne
  - RNSInteger< RNS >, [546](#)
  - RNSIntegerMod< RNS >, [551](#)
- IsSingular
  - FFPACK, [333](#), [374](#)
- IsSingular\_modular\_double
  - ffpack.C, [1003](#)
  - ffpack\_c.h, [1025](#)
- isSparseMatrix< Field, M >, [479](#)

- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO > >, [480](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >, [480](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR > >, [480](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_HYB > >, [480](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >, [481](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL > >, [481](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd > >, [481](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >, [482](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >, [482](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::HYB\_ZO > >, [482](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL > >, [483](#)
- isSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >, [483](#)
- isSparseMatrixMKLFormat< F, M >, [483](#)
- isSparseMatrixSimdFormat< F, M >, [483](#)
- isTerminated
  - ForStrategy1D< blocksize\_t, Cut, Param >, [460](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [462](#)
- isZero
  - RNSInteger< RNS >, [547](#)
  - RNSIntegerMod< RNS >, [551](#)
- isZOSparseMatrix< F, M >, [484](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::COO\_ZO > >, [484](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::CSR\_ZO > >, [484](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_simd\_ZO > >, [485](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::ELL\_ZO > >, [485](#)
- isZOSparseMatrix< Field, Sparse< Field, SparseMatrix\_t::SELL\_ZO > >, [485](#)
- Iterative, [485](#)
- jbegin
  - ForStrategy2D< blocksize\_t, Cut, Param >, [462](#)
- jend
  - ForStrategy2D< blocksize\_t, Cut, Param >, [462](#)
- kaapi\_routines.inl, [1039](#)
- \_\_FFLASFFPACK\_KAAPI\_ROUTINES\_INL, [1039](#)
- KellerGehrig
  - FFPACK::Protected, [398](#)
- KGFast
  - FFPACK::Protected, [398](#)
- KGFast\_generalized
  - FFPACK::Protected, [398](#)
- kmax
  - Sparse< \_Field, SparseMatrix\_t::COO >, [738](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [741](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [742](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [746](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [748](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [749](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [751](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [752](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [754](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [756](#)
- KrylovElim
  - FFPACK, [373](#)
- KrylovElim\_modular\_double
  - ffpack.C, [1003](#)
  - ffpack\_c.h, [1024](#)
- lapack.C, [1064](#)
- \_\_FFLASFFPACK\_CONFIGURATION, [1064](#)
- \_\_FFLASFFPACK\_HAVE\_LAPACK, [1064](#)
- main, [1064](#)
- LAPACKPerm2MathPerm
  - FFPACK, [310](#)
  - ffpack.C, [991](#)
  - ffpack\_c.h, [1015](#)
- lastBlockSize
  - ForStrategy1D< blocksize\_t, Cut, Param >, [461](#)
- lastCBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [464](#)
- lastRBS
  - ForStrategy2D< blocksize\_t, Cut, Param >, [464](#)
- launch\_fger
  - test-fger.C, [1081](#)
- launch\_fger\_dispatch
  - test-fger.C, [1082](#)
- launch\_MM
  - test-fgemm.C, [1077](#)
- launch\_MM\_dispatch
  - test-fgemm-check.C, [1075](#)
  - test-fgemm.C, [1078](#)
- launch\_MV
  - test-fgemv.C, [1079](#)
- launch\_MV\_dispatch
  - test-fgemv.C, [1080](#)
- launch\_test
  - test-charpoly.C, [1067](#)
  - test-lu.C, [1104](#)
- launch\_wino
  - benchmark-wino.C, [803](#)
- LazyTag, [486](#)
- ld
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [746](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [747](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [749](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [751](#)

- LeadingSubmatrixRankProfiles
  - FFPACK, [340](#)
  - ffpack.C, [1006](#)
  - ffpack\_c.h, [1027](#)
- lesser
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [560](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [565](#)
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [581](#)
  - Simd128\_impl< true, true, false, 8 >, [590](#)
  - Simd128\_impl< true, true, true, 2 >, [604](#)
  - Simd128\_impl< true, true, true, 4 >, [613](#)
  - Simd128\_impl< true, true, true, 8 >, [621](#)
  - Simd256\_impl< true, false, true, 8 >, [631](#)
  - Simd256\_impl< true, true, false, 2 >, [637](#)
  - Simd256\_impl< true, true, false, 4 >, [648](#), [651](#)
  - Simd256\_impl< true, true, false, 8 >, [664](#)
  - Simd256\_impl< true, true, true, 2 >, [678](#)
  - Simd256\_impl< true, true, true, 4 >, [688](#), [694](#)
  - Simd256\_impl< true, true, true, 8 >, [703](#)
  - Simd512\_impl< true, false, true, 8 >, [711](#)
  - Simd512\_impl< true, true, false, 8 >, [716](#)
  - Simd512\_impl< true, true, true, 8 >, [731](#)
- lesser\_eq
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [560](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [565](#)
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [581](#)
  - Simd128\_impl< true, true, false, 8 >, [591](#)
  - Simd128\_impl< true, true, true, 2 >, [604](#)
  - Simd128\_impl< true, true, true, 4 >, [613](#)
  - Simd128\_impl< true, true, true, 8 >, [621](#)
  - Simd256\_impl< true, false, true, 8 >, [631](#)
  - Simd256\_impl< true, true, false, 2 >, [637](#)
  - Simd256\_impl< true, true, false, 4 >, [648](#), [651](#)
  - Simd256\_impl< true, true, false, 8 >, [664](#)
  - Simd256\_impl< true, true, true, 2 >, [678](#)
  - Simd256\_impl< true, true, true, 4 >, [688](#), [694](#)
  - Simd256\_impl< true, true, true, 8 >, [703](#)
  - Simd512\_impl< true, false, true, 8 >, [711](#)
  - Simd512\_impl< true, true, false, 8 >, [717](#)
  - Simd512\_impl< true, true, true, 8 >, [731](#)
- limits< char >, [486](#)
  - digits, [486](#)
  - max, [486](#)
  - min, [486](#)
  - T, [486](#)
- limits< double >, [487](#)
  - digits, [487](#)
  - max, [487](#)
  - min, [487](#)
  - T, [487](#)
- limits< float >, [487](#)
  - digits, [488](#)
  - max, [488](#)
  - min, [488](#)
  - T, [488](#)
- limits< Givaro::Integer >, [488](#)
  - max, [488](#)
  - min, [488](#)
  - T, [488](#)
- limits< int >, [489](#)
  - digits, [489](#)
  - max, [489](#)
  - min, [489](#)
  - T, [489](#)
- limits< long >, [489](#)
  - digits, [490](#)
  - max, [490](#)
  - min, [490](#)
  - T, [490](#)
- limits< long long >, [490](#)
  - digits, [491](#)
  - max, [491](#)
  - min, [491](#)
  - T, [490](#)
- limits< Reclnt::rint< K > >, [491](#)
  - max, [491](#)
  - min, [491](#)
  - T, [491](#)
- limits< Reclnt::ruint< K > >, [492](#)
  - max, [492](#)
  - min, [492](#)
  - T, [492](#)
- limits< short int >, [492](#)
  - digits, [493](#)
  - max, [493](#)
  - min, [493](#)
  - T, [492](#)
- limits< signed char >, [493](#)
  - digits, [493](#)
  - max, [493](#)
  - min, [493](#)
  - T, [493](#)
- limits< T >, [486](#)
- limits< unsigned char >, [494](#)
  - digits, [494](#)
  - max, [494](#)
  - min, [494](#)
  - T, [494](#)
- limits< unsigned int >, [494](#)
  - digits, [495](#)
  - max, [495](#)
  - min, [495](#)
  - T, [495](#)
- limits< unsigned long >, [495](#)
  - digits, [496](#)
  - max, [496](#)
  - min, [496](#)
  - T, [495](#)

- limits< unsigned long long >, [496](#)
  - digits, [496](#)
  - max, [496](#)
  - min, [496](#)
  - T, [496](#)
- limits< unsigned short int >, [497](#)
  - digits, [497](#)
  - max, [497](#)
  - min, [497](#)
  - T, [497](#)
- load
  - Simd128\_impl< true, true, false, 2 >, [570](#), [573](#)
  - Simd128\_impl< true, true, false, 4 >, [580](#), [583](#)
  - Simd128\_impl< true, true, false, 8 >, [590](#), [592](#)
  - Simd128\_impl< true, true, true, 2 >, [600](#)
  - Simd128\_impl< true, true, true, 4 >, [608](#)
  - Simd128\_impl< true, true, true, 8 >, [617](#)
  - Simd256\_impl< true, false, true, 8 >, [628](#)
  - Simd256\_impl< true, true, false, 2 >, [636](#), [639](#)
  - Simd256\_impl< true, true, false, 4 >, [647](#), [650](#), [653](#)
  - Simd256\_impl< true, true, false, 8 >, [663](#), [666](#)
  - Simd256\_impl< true, true, true, 2 >, [673](#)
  - Simd256\_impl< true, true, true, 4 >, [683](#), [690](#)
  - Simd256\_impl< true, true, true, 8 >, [698](#)
  - Simd512\_impl< true, false, true, 8 >, [708](#)
  - Simd512\_impl< true, true, false, 8 >, [716](#), [719](#)
  - Simd512\_impl< true, true, true, 8 >, [727](#)
- loadu
  - Simd128\_impl< true, true, false, 2 >, [570](#), [573](#)
  - Simd128\_impl< true, true, false, 4 >, [580](#), [583](#)
  - Simd128\_impl< true, true, false, 8 >, [590](#), [593](#)
  - Simd128\_impl< true, true, true, 2 >, [600](#)
  - Simd128\_impl< true, true, true, 4 >, [609](#)
  - Simd128\_impl< true, true, true, 8 >, [617](#)
  - Simd256\_impl< true, false, true, 8 >, [628](#)
  - Simd256\_impl< true, true, false, 2 >, [636](#), [639](#)
  - Simd256\_impl< true, true, false, 4 >, [647](#), [650](#), [653](#)
  - Simd256\_impl< true, true, false, 8 >, [663](#), [666](#)
  - Simd256\_impl< true, true, true, 2 >, [673](#)
  - Simd256\_impl< true, true, true, 4 >, [684](#), [690](#)
  - Simd256\_impl< true, true, true, 8 >, [698](#)
  - Simd512\_impl< true, false, true, 8 >, [708](#)
  - Simd512\_impl< true, true, false, 8 >, [716](#), [719](#)
  - Simd512\_impl< true, true, true, 8 >, [727](#)
- LQUPtoInverseOfFullRankMinor
  - FFPACK, [349](#), [380](#)
- LT\_OBJDIR
  - config.h, [806](#)
- LUdivine
  - FFPACK, [321](#), [356](#), [369](#)
- LUdivine\_construct
  - FFPACK::Protected, [397](#), [402](#)
- LUdivine\_gauss
  - FFPACK, [355](#), [370](#)
- LUdivine\_gauss\_modular\_double
  - ffpack\_c.h, [1020](#)
- LUdivine\_modular\_double
  - ffpack.C, [996](#)
  - ffpack\_c.h, [1019](#)
- LUdivine\_small
  - FFPACK, [355](#), [369](#)
- LUdivine\_small\_modular\_double
  - ffpack\_c.h, [1019](#)
- LUKrylov
  - FFPACK::Protected, [399](#)
- LUKrylov\_KGFast
  - FFPACK::Protected, [400](#)
- m
  - Sparse< \_Field, SparseMatrix\_t::COO >, [738](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [739](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR >, [741](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [743](#)
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [744](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL >, [746](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [747](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [749](#)
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [751](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [752](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [754](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [756](#)
- MachineFloatTag, [497](#)
- MachineIntTag, [498](#)
- main
  - 101-fgemv.C, [1118](#)
  - 2x2-fgemv.C, [1118](#)
  - 2x2-ftsrv.C, [1119](#)
  - 2x2-pluq.C, [1119](#)
  - arithprog.C, [770](#)
  - autotune/charpoly.C, [771](#)
  - autotune/pluq.C, [775](#)
  - benchmark-charpoly-mp.C, [777](#)
  - benchmark-charpoly.C, [778](#)
  - benchmark-checkers.C, [779](#)
  - benchmark-dgemm.C, [780](#)
  - benchmark-dgetrf.C, [781](#)
  - benchmark-dgetri.C, [782](#)
  - benchmark-dsytrf.C, [783](#)
  - benchmark-dtrsm.C, [783](#)
  - benchmark-dtrtri.C, [784](#)
  - benchmark-fadd-lvl2.C, [785](#)
  - benchmark-fdot.C, [786](#)
  - benchmark-fgemm-mp.C, [787](#)
  - benchmark-fgemm-rns.C, [789](#)
  - benchmark-fgemm.C, [790](#)
  - benchmark-fgemv-mp.C, [791](#)
  - benchmark-fgemv.C, [794](#)
  - benchmark-fgesv.C, [795](#)
  - benchmark-fsyrr.C, [796](#)
  - benchmark-fsytrf.C, [797](#)
  - benchmark-ftsrm-mp.C, [797](#)
  - benchmark-ftsrm.C, [798](#)
  - benchmark-ftsrv.C, [798](#)

- benchmark-ctrtri.C, [799](#)
- benchmark-inverse.C, [800](#)
- benchmark-lqmp-mp.C, [800](#)
- benchmark-lqmp.C, [801](#)
- benchmark-pluq.C, [802](#)
- benchmark-wino.C, [803](#)
- cblas.C, [1058](#)
- clapack.C, [1059](#)
- cuda.C, [1059](#)
- det.C, [812](#)
- examples/charpoly.C, [771](#)
- examples/pluq.C, [776](#)
- fblas.C, [1060](#)
- fflas-101\_1.C, [1120](#)
- fflas-101\_3.C, [1120](#)
- fflas\_101.C, [1120](#)
- fflas\_101\_lvl1.C, [1121](#)
- ffpack-fgesv.C, [1121](#)
- ffpack-solve.C, [1122](#)
- fsyrk.C, [772](#)
- fsytrf.C, [773](#)
- ctrtri.C, [774](#)
- gmp.C, [1060](#)
- lapack.C, [1064](#)
- matmul.C, [812](#)
- rank.C, [813](#)
- regression-check.C, [1065](#)
- solve.C, [813](#)
- test-charpoly-check.C, [1066](#)
- test-charpoly.C, [1067](#)
- test-compressQ.C, [1068](#)
- test-det-check.C, [1069](#)
- test-det.C, [1070](#)
- test-echelon.C, [1072](#)
- test-fadd.C, [1073](#)
- test-fdot.C, [1075](#)
- test-fgemm-check.C, [1076](#)
- test-fgemm.C, [1078](#)
- test-fgemv.C, [1080](#)
- test-fger.C, [1082](#)
- test-fgesv.C, [1083](#)
- test-finit.C, [1084](#)
- test-fscal.C, [1086](#)
- test-fsyr2k.C, [1087](#)
- test-fsyrk.C, [1089](#)
- test-fsytrf.C, [1090](#)
- test-ctrmm.C, [1091](#)
- test-ctrmv.C, [1093](#)
- test-ctrsm-check.C, [1093](#)
- test-ctrsm.C, [1095](#)
- test-ctrssyr2k.C, [1096](#)
- test-ctrstr.C, [1097](#)
- test-ctrsv.C, [1098](#)
- test-ctrtri.C, [1099](#)
- test-interfaces-c.c, [1100](#)
- test-invert-check.C, [1100](#)
- test-io.C, [1101](#)
- test-lu.C, [1105](#)
- test-maxdelayeddim.C, [1106](#)
- test-minpoly.C, [1107](#)
- test-multifile2.C, [1108](#)
- test-nullspace.C, [1109](#)
- test-permutations.C, [1110](#)
- test-pluq-check.C, [1111](#)
- test-rankprofiles.C, [1112](#)
- test-rpm.C, [1113](#)
- test-simd.C, [1117](#)
- test-solve.C, [1117](#)
- winograd.C, [777](#)
- mainpage.doxy, [812](#)
- mask\_high
  - Simd128\_impl< true, true, false, 8 >, [595](#)
  - Simd128\_impl< true, true, true, 8 >, [622](#)
  - Simd256\_impl< true, true, false, 8 >, [669](#)
  - Simd256\_impl< true, true, true, 8 >, [703](#)
  - Simd512\_impl< true, true, false, 8 >, [722](#)
  - Simd512\_impl< true, true, true, 8 >, [732](#)
- mask\_t
  - read\_sparse.h, [911](#)
- maskstore
  - Simd512\_impl< true, true, false, 8 >, [716](#), [719](#)
  - Simd512\_impl< true, true, true, 8 >, [727](#)
- MatF2MatD\_Triangular
  - FFLAS::Protected, [229](#)
- MatF2MatFI\_Triangular
  - FFLAS::Protected, [229](#)
- MathPerm2LAPACKPerm
  - FFPACK, [310](#)
  - ffpack.C, [991](#)
  - ffpack\_c.h, [1015](#)
- Matio.h, [1056](#)
  - read\_field, [1056](#)
  - write\_field, [1056](#)
- matmul.C, [812](#)
  - main, [812](#)
- matmul.doxy, [841](#)
- Matrix Multiplication Algorithms, [50](#)
- MatrixApplyS
  - FFPACK, [358](#), [359](#)
- MatrixApplyS\_modular\_double
  - ffpack.C, [991](#)
  - ffpack\_c.h, [1015](#)
- MatrixApplyT
  - FFPACK, [360](#), [361](#)
- MatrixApplyT\_modular\_double
  - ffpack.C, [992](#)
  - ffpack\_c.h, [1015](#)
- MatVecMinPoly
  - FFPACK, [331](#), [373](#)
  - FFPACK::Protected, [400](#)
- max
  - HelperMod< Field, ElementCategories::MachineIntTag >, [473](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [476](#)
  - limits< char >, [486](#)

- limits< double >, 487
- limits< float >, 488
- limits< Givaro::Integer >, 488
- limits< int >, 489
- limits< long >, 490
- limits< long long >, 491
- limits< RecInt::rint< K > >, 491
- limits< RecInt::ruint< K > >, 492
- limits< short int >, 493
- limits< signed char >, 493
- limits< unsigned char >, 494
- limits< unsigned int >, 495
- limits< unsigned long >, 496
- limits< unsigned long long >, 496
- limits< unsigned short int >, 497
- max3
  - FFLAS, 84
- max4
  - FFLAS, 84
- MAX\_THREADS
  - parallel.h, 1040
- MAX\_WITH\_SIZE\_T
  - test-maxdelayeddim.C, 1106
- maxCol
  - StatsMatrix, 760
- maxColDifference
  - StatsMatrix, 760
- MaxDelayedDim
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 500
- maxElement
  - RNSIntegerMod< RNS >, 551
- maxFieldElt
  - FFPACK, 394
- maxFieldElt< Givaro::ZRing< Givaro::Integer > >
  - FFPACK, 394
- maxpy
  - FieldSimd< \_Field >, 449
- maxpyin
  - FieldSimd< \_Field >, 450
- maxRow
  - StatsMatrix, 760
- maxrow
  - Sparse< \_Field, SparseMatrix\_t::COO >, 738
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, 740
  - Sparse< \_Field, SparseMatrix\_t::CSR >, 741
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 743
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 745
  - Sparse< \_Field, SparseMatrix\_t::ELL >, 746
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, 748
  - Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, 750
  - Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, 751
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 753
  - Sparse< \_Field, SparseMatrix\_t::SELL >, 754
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 757
- maxRowDifference
  - StatsMatrix, 761
- MaxStorableValue
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, 503
- MG\_DEFAULT
  - benchmark-fgemm-mp.C, 787
  - benchmark-fgemv-mp.C, 790
- min
  - HelperMod< Field, ElementCategories::MachineIntTag >, 473
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, 475
  - limits< char >, 486
  - limits< double >, 487
  - limits< float >, 488
  - limits< Givaro::Integer >, 488
  - limits< int >, 489
  - limits< long >, 490
  - limits< long long >, 491
  - limits< RecInt::rint< K > >, 491
  - limits< RecInt::ruint< K > >, 492
  - limits< short int >, 493
  - limits< signed char >, 493
  - limits< unsigned char >, 494
  - limits< unsigned int >, 495
  - limits< unsigned long >, 496
  - limits< unsigned long long >, 496
  - limits< unsigned short int >, 497
- min3
  - FFLAS, 84
- min4
  - FFLAS, 84
- min\_types
  - FFLAS::Protected, 226, 227
- minCol
  - StatsMatrix, 760
- minColDifference
  - StatsMatrix, 760
- minElement
  - RNSIntegerMod< RNS >, 551
- MinPoly
  - FFPACK, 331, 373
- minRow
  - StatsMatrix, 760
- minRowDifference
  - StatsMatrix, 761
- MKL\_CONFIG, 403
- MKLSparseMatrixFormat
  - FFLAS, 79
- MMHelper
  - MMHelper< FFPACK::RNSInteger< E >, Algo-Trait, ModeCategories::DefaultTag, ParSeq-Trait >, 504
  - MMHelper< FFPACK::RNSIntegerMod< E >, Algo-Trait, ModeCategories::DefaultTag, ParSeq-Trait >, 506
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, 508



- MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<Aunfit, 501  
   ElementCategories::RNSElementTag >, Bmax, 502  
   ParSeqTrait >, 509, 510 Bmin, 502
- MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, Bunfit, 501  
   ParSeqTrait >, 512
- MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-  
   Trait >, 499, 500
- MMHelper< FFPACK::RNSInteger< E >, AlgoTrait,  
   ModeCategories::DefaultTag, ParSeqTrait >,  
   503
- MMHelper, 504
- normA, 505
- normB, 505
- operator<<, 504
- parseq, 505
- recLevel, 505
- Self\_t, 503
- setNorm, 504
- MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait,  
   ModeCategories::DefaultTag, ParSeqTrait >,  
   505
- MMHelper, 506
- normA, 507
- normB, 507
- operator<<, 507
- parseq, 507
- recLevel, 507
- Self\_t, 506
- setNorm, 506
- MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<mod  
   Dest >, ParSeqTrait >, 507
- MMHelper, 508
- operator<<, 508
- parseq, 508
- recLevel, 508
- Self\_t, 508
- MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo<  
   ElementCategories::RNSElementTag >,  
   ParSeqTrait >, 509
- MMHelper, 509, 510
- normA, 510
- normB, 510
- operator<<, 510
- parseq, 511
- recLevel, 511
- Self\_t, 509
- setNorm, 510
- MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag,  
   ParSeqTrait >, 511
- MMHelper, 512
- operator<<, 512
- parseq, 512
- recLevel, 512
- Self\_t, 511
- MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >,  
   498
- Amax, 502
- Amin, 502
- checkA, 501
- checkB, 501
- checkOut, 501
- Cmax, 502
- Cmin, 502
- DelayedField, 499
- delayedField, 503
- DelayedField\_t, 499
- DFElt, 499
- FieldMax, 502
- FieldMin, 502
- initA, 500
- initB, 500
- initC, 500
- initOut, 500
- MaxDelayedDim, 500
- MaxStorableValue, 503
- MMHelper, 499, 500
- operator<<, 501
- Outmax, 502
- Outmin, 502
- parseq, 503
- recLevel, 502
- Self\_t, 499
- setOutBounds, 501
- FieldSimd< \_Field >, 448
- Simd128\_impl< true, true, false, 2 >, 576
- Simd128\_impl< true, true, false, 4 >, 586
- Simd128\_impl< true, true, false, 8 >, 596
- Simd128\_impl< true, true, true, 2 >, 604
- Simd128\_impl< true, true, true, 4 >, 613
- Simd128\_impl< true, true, true, 8 >, 622
- Simd256\_impl< true, false, true, 8 >, 632
- Simd256\_impl< true, true, false, 2 >, 642
- Simd256\_impl< true, true, false, 4 >, 659
- Simd256\_impl< true, true, false, 8 >, 670
- Simd256\_impl< true, true, true, 2 >, 678
- Simd256\_impl< true, true, true, 4 >, 689, 694
- Simd256\_impl< true, true, true, 8 >, 704
- Simd512\_impl< true, true, false, 8 >, 722
- Simd512\_impl< true, true, true, 8 >, 732
- MODE
- parallel.h, 1043
- ModeTraits< Field >, 513
- value, 513
- ModeTraits< Givaro::Modular< Element, Compute >  
   >, 513
- value, 513
- ModeTraits< Givaro::Modular< Givaro::Integer, Com-  
   pute > >, 513
- value, 514
- ModeTraits< Givaro::Modular< int16\_t, Compute > >,  
   514

- value, [514](#)
- ModeTraits< Givaro::Modular< int32\_t, Compute > >, [514](#)
  - value, [514](#)
- ModeTraits< Givaro::Modular< int8\_t, Compute > >, [514](#)
  - value, [515](#)
- ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >, [515](#)
  - value, [515](#)
- ModeTraits< Givaro::Modular< uint16\_t, Compute > >, [515](#)
  - value, [515](#)
- ModeTraits< Givaro::Modular< uint32\_t, Compute > >, [516](#)
  - value, [516](#)
- ModeTraits< Givaro::Modular< uint8\_t, Compute > >, [516](#)
  - value, [516](#)
- ModeTraits< Givaro::ModularBalanced< Element > >, [516](#)
  - value, [516](#)
- ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >, [517](#)
  - value, [517](#)
- ModeTraits< Givaro::ModularBalanced< int16\_t > >, [517](#)
  - value, [517](#)
- ModeTraits< Givaro::ModularBalanced< int32\_t > >, [517](#)
  - value, [518](#)
- ModeTraits< Givaro::ModularBalanced< int8\_t > >, [518](#)
  - value, [518](#)
- ModeTraits< Givaro::Montgomery< T > >, [518](#)
  - value, [518](#)
- ModeTraits< Givaro::ZRing< double > >, [518](#)
  - value, [519](#)
- ModeTraits< Givaro::ZRing< float > >, [519](#)
  - value, [519](#)
- ModeTraits< Givaro::ZRing< Givaro::Integer > >, [519](#)
  - value, [519](#)
- ModField
  - rns\_double, [528](#)
  - rns\_double\_extended, [541](#)
  - RNSIntegerMod< RNS >, [550](#)
- modp
  - FFLAS::vectorised, [291](#)
  - FFLAS::vectorised::unswitch, [292](#)
- ModularBalanced< T >, [519](#)
- ModularTag, [520](#)
- mOne
  - RNSInteger< RNS >, [548](#)
  - RNSIntegerMod< RNS >, [555](#)
- mone
  - FFLAS, [83](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [753](#)
- MonotonicApplyP
  - FFPACK, [312](#)
- MonotonicCompress
  - FFPACK, [356](#)
- MonotonicCompressCycles
  - FFPACK, [357](#)
- MonotonicCompressMorePivots
  - FFPACK, [357](#)
- MonotonicExpand
  - FFPACK, [357](#)
- Montgomery< T >, [520](#)
- mul
  - FieldSimd< \_Field >, [448](#)
  - RNSIntegerMod< RNS >, [553](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [558](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [562](#)
  - Simd128\_impl< true, true, false, 2 >, [575](#)
  - Simd128\_impl< true, true, false, 4 >, [585](#)
  - Simd128\_impl< true, true, false, 8 >, [594](#)
  - Simd128\_impl< true, true, true, 2 >, [602](#)
  - Simd128\_impl< true, true, true, 4 >, [610](#)
  - Simd128\_impl< true, true, true, 8 >, [619](#)
  - Simd256\_impl< true, false, true, 8 >, [630](#)
  - Simd256\_impl< true, true, false, 2 >, [641](#)
  - Simd256\_impl< true, true, false, 4 >, [657](#)
  - Simd256\_impl< true, true, false, 8 >, [668](#)
  - Simd256\_impl< true, true, true, 2 >, [676](#)
  - Simd256\_impl< true, true, true, 4 >, [686](#), [691](#)
  - Simd256\_impl< true, true, true, 8 >, [701](#)
  - Simd512\_impl< true, false, true, 8 >, [710](#)
  - Simd512\_impl< true, true, false, 8 >, [721](#)
  - Simd512\_impl< true, true, true, 8 >, [729](#)
- mul\_r
  - FieldSimd< \_Field >, [448](#)
- mulhi
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [563](#)
  - Simd128\_impl< true, true, false, 2 >, [571](#)
  - Simd128\_impl< true, true, false, 4 >, [581](#)
  - Simd128\_impl< true, true, true, 2 >, [602](#)
  - Simd128\_impl< true, true, true, 4 >, [611](#)
  - Simd256\_impl< true, true, false, 2 >, [637](#)
  - Simd256\_impl< true, true, false, 4 >, [648](#), [651](#)
  - Simd256\_impl< true, true, true, 2 >, [676](#)
  - Simd256\_impl< true, true, true, 4 >, [686](#), [692](#)
- mulhi\_fast
  - Simd128\_impl< true, true, false, 8 >, [596](#)
  - Simd128\_impl< true, true, true, 8 >, [622](#)
  - Simd256\_impl< true, true, false, 8 >, [670](#)
  - Simd256\_impl< true, true, true, 8 >, [703](#)
  - Simd512\_impl< true, true, false, 8 >, [722](#)
  - Simd512\_impl< true, true, true, 8 >, [732](#)
- mulin
  - FieldSimd< \_Field >, [448](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type



- [>](#), [559](#)
- [Simd256\\_impl< true, false, true, 8 >](#), [630](#)
- [Simd512\\_impl< true, false, true, 8 >](#), [710](#)
- mullo**
  - [ScalFunctions< Element, typename enable\\_if< is\\_integral< Element >::value >::type >](#), [562](#)
  - [Simd128\\_impl< true, true, false, 2 >](#), [575](#)
  - [Simd128\\_impl< true, true, false, 4 >](#), [584](#)
  - [Simd128\\_impl< true, true, false, 8 >](#), [591](#)
  - [Simd128\\_impl< true, true, true, 2 >](#), [602](#)
  - [Simd128\\_impl< true, true, true, 4 >](#), [610](#)
  - [Simd128\\_impl< true, true, true, 8 >](#), [619](#)
  - [Simd256\\_impl< true, true, false, 2 >](#), [641](#)
  - [Simd256\\_impl< true, true, false, 4 >](#), [656](#), [657](#)
  - [Simd256\\_impl< true, true, false, 8 >](#), [664](#)
  - [Simd256\\_impl< true, true, true, 2 >](#), [676](#)
  - [Simd256\\_impl< true, true, true, 4 >](#), [686](#), [691](#)
  - [Simd256\\_impl< true, true, true, 8 >](#), [701](#)
  - [Simd512\\_impl< true, true, false, 8 >](#), [717](#)
  - [Simd512\\_impl< true, true, true, 8 >](#), [729](#)
- mulx**
  - [ScalFunctions< Element, typename enable\\_if< is\\_integral< Element >::value >::type >](#), [563](#)
  - [Simd128\\_impl< true, true, false, 2 >](#), [571](#)
  - [Simd128\\_impl< true, true, false, 4 >](#), [581](#)
  - [Simd128\\_impl< true, true, false, 8 >](#), [591](#)
  - [Simd128\\_impl< true, true, true, 2 >](#), [602](#)
  - [Simd128\\_impl< true, true, true, 4 >](#), [611](#)
  - [Simd128\\_impl< true, true, true, 8 >](#), [619](#)
  - [Simd256\\_impl< true, true, false, 2 >](#), [637](#)
  - [Simd256\\_impl< true, true, false, 4 >](#), [648](#), [651](#)
  - [Simd256\\_impl< true, true, false, 8 >](#), [665](#)
  - [Simd256\\_impl< true, true, true, 2 >](#), [676](#)
  - [Simd256\\_impl< true, true, true, 4 >](#), [686](#), [692](#)
  - [Simd256\\_impl< true, true, true, 8 >](#), [701](#)
  - [Simd512\\_impl< true, true, false, 8 >](#), [717](#)
  - [Simd512\\_impl< true, true, true, 8 >](#), [729](#)
- mvcnt**
  - [test-lu.C](#), [1106](#)
- n**
  - [Sparse< \\_Field, SparseMatrix\\_t::COO >](#), [738](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::COO\\_ZO >](#), [739](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::CSR >](#), [741](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::CSR\\_HYB >](#), [743](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::CSR\\_ZO >](#), [744](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::ELL >](#), [746](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd >](#), [747](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#), [749](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_ZO >](#), [751](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::HYB\\_ZO >](#), [753](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::SELL >](#), [754](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::SELL\\_ZO >](#), [756](#)
- nChunks**
  - [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd >](#), [748](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#), [750](#)
- [Sparse< \\_Field, SparseMatrix\\_t::SELL >](#), [755](#)
- [Sparse< \\_Field, SparseMatrix\\_t::SELL\\_ZO >](#), [757](#)
- nDenseCols**
  - [StatsMatrix](#), [761](#)
- nDenseRows**
  - [StatsMatrix](#), [761](#)
- [need\\_field\\_characteristic< Field >](#), [520](#)
- [value](#), [520](#)
- [need\\_field\\_characteristic< Givaro::Modular< Field >](#), [520](#)
- [value](#), [520](#)
- [need\\_field\\_characteristic< Givaro::ModularBalanced< Field >](#), [520](#)
- [value](#), [521](#)
- NeedDoublePreAddReduction**
  - [FFLAS::Protected](#), [222](#)
- NeedPreAddReduction**
  - [FFLAS::Protected](#), [221](#)
- NeedPreSubReduction**
  - [FFLAS::Protected](#), [221](#)
- neg**
  - [RNSIntegerMod< RNS >](#), [553](#)
- nElements**
  - [Sparse< \\_Field, SparseMatrix\\_t::COO >](#), [738](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::COO\\_ZO >](#), [740](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::CSR >](#), [741](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::CSR\\_HYB >](#), [743](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::CSR\\_ZO >](#), [744](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::ELL >](#), [746](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd >](#), [748](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_simd\\_ZO >](#), [749](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::ELL\\_ZO >](#), [751](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::HYB\\_ZO >](#), [753](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::SELL >](#), [755](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::SELL\\_ZO >](#), [757](#)
- nEmptyCols**
  - [StatsMatrix](#), [761](#)
- nEmptyColsEnd**
  - [StatsMatrix](#), [761](#)
- nEmptyRows**
  - [StatsMatrix](#), [761](#)
- newD**
  - [FFPACK::Protected](#), [401](#)
- NEWWINO**
  - [fgemm\\_winograd.inl](#), [841](#)
- nMOnes**
  - [Sparse< \\_Field, SparseMatrix\\_t::CSR\\_HYB >](#), [743](#)
  - [StatsMatrix](#), [759](#)
- nnz**
  - [Sparse< \\_Field, SparseMatrix\\_t::COO >](#), [738](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::COO\\_ZO >](#), [739](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::CSR >](#), [741](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::CSR\\_HYB >](#), [743](#)
  - [Sparse< \\_Field, SparseMatrix\\_t::CSR\\_ZO >](#), [744](#)

- Sparse< \_Field, SparseMatrix\_t::ELL >, 746
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, 748
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, 749
- Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, 751
- Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 753
- Sparse< \_Field, SparseMatrix\_t::SELL >, 755
- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 757
- StatsMatrix, 760
- none
  - HelperFlag, 472
- nOnes
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 743
  - StatsMatrix, 759
- NonZeroRandomMatrix
  - FFPACK, 381, 382
- normA
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 505
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 507
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, 510
- normB
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 505
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 507
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, 510
- NORML\_MOD
  - fflas\_simd.h, 874
- NoSimd< T >, 521
  - compliant, 521
  - scalar\_t, 521
  - type\_string, 521
  - valid, 521
  - vect\_size, 522
  - vect\_t, 521
- NoSimdSparseMatrix
  - FFLAS, 79
- NOSPLIT
  - parallel.h, 1045
- nOthers
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 743
  - StatsMatrix, 759
- NotMKLSparseMatrixFormat
  - FFLAS, 79
- NotZOSparseMatrix
  - FFLAS, 79
- NullSpaceBasis
  - FFPACK, 336, 376
- NullSpaceBasis\_modular\_double
  - ffpack.C, 1005
  - ffpack\_c.h, 1026
- NUM\_THREADS
  - parallel.h, 1040
- NUMARGS
  - parallel.h, 1043
- number\_kind
  - FFLAS, 83
- numBlock
  - ForStrategy1D< blocksize\_t, Cut, Param >, 461
- numblocks
  - ForStrategy1D< blocksize\_t, Cut, Param >, 460
- numColBlock
  - ForStrategy2D< blocksize\_t, Cut, Param >, 464
- numRowBlock
  - ForStrategy2D< blocksize\_t, Cut, Param >, 464
- numthreads
  - Parallel< C, P >, 522
  - Sequential, 566
- one
  - FFLAS, 83
  - RNSInteger< RNS >, 548
  - RNSIntegerMod< RNS >, 555
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, 753
- OPENBLAS\_NUM\_THREADS
  - config.h, 806
- operator!=
  - rns\_double\_elt\_cstptr, 536
  - rns\_double\_elt\_ptr, 539
- operator<
  - rns\_double\_elt\_cstptr, 536
  - rns\_double\_elt\_ptr, 539
- operator<<
  - Compose< H1, H2 >, 422
  - FFLAS, 156
  - ForStrategy2D< blocksize\_t, Cut, Param >, 463
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 504
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 507
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, 508
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, 510
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 512
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 501
  - Parallel< C, P >, 523
  - Sequential, 566
  - test-fsytrf.C, 1089
- operator\*

- rns\_double\_elt\_cstptr, [535](#)
- rns\_double\_elt\_ptr, [538](#)
- operator()
  - callLUdivine\_small< double >, [410](#)
  - callLUdivine\_small< Element >, [410](#)
  - callLUdivine\_small< float >, [411](#)
  - Failure, [442](#)
  - readMyMachineType< Field, mpz\_t >, [526](#)
  - readMyMachineType< Field, T >, [526](#)
  - RNSInteger< RNS >::RandIter, [524](#)
  - RNSIntegerMod< RNS >::RandIter, [525](#)
  - rnsRandIter< RNS >, [556](#)
  - tfn\_minus, [764](#)
  - tfn\_minus\_eq, [764](#)
  - tfn\_mul, [764](#)
  - tfn\_mul\_eq, [765](#)
  - tfn\_plus, [765](#)
  - tfn\_plus\_eq, [766](#)
- operator+
  - rns\_double\_elt\_cstptr, [536](#)
  - rns\_double\_elt\_ptr, [539](#)
- operator++
  - ForStrategy1D< blocksize\_t, Cut, Param >, [460](#)
  - ForStrategy2D< blocksize\_t, Cut, Param >, [462](#)
  - rns\_double\_elt\_cstptr, [536](#)
  - rns\_double\_elt\_ptr, [539](#)
- operator+=
  - rns\_double\_elt\_cstptr, [536](#)
  - rns\_double\_elt\_ptr, [539](#)
- operator-
  - rns\_double\_elt\_cstptr, [536](#)
  - rns\_double\_elt\_ptr, [539](#)
- operator--
  - rns\_double\_elt\_cstptr, [536](#)
  - rns\_double\_elt\_ptr, [539](#)
- operator-=
  - rns\_double\_elt\_cstptr, [536](#)
  - rns\_double\_elt\_ptr, [539](#)
- operator=
  - Coo< Field >, [430](#), [431](#)
  - Coo< ValT, IdxT >, [429](#), [432](#)
  - FieldSimd< \_Field >, [446](#)
  - Info, [477](#), [478](#)
  - rns\_double\_elt\_cstptr, [536](#)
  - rns\_double\_elt\_ptr, [539](#)
- operator&
  - rns\_double\_elt, [533](#)
  - rns\_double\_elt\_cstptr, [535](#), [536](#)
  - rns\_double\_elt\_ptr, [538](#), [539](#)
- operator[]
  - rns\_double\_elt\_cstptr, [535](#)
  - rns\_double\_elt\_ptr, [538](#)
- other
  - FFLAS, [83](#)
  - rns\_double\_elt\_cstptr, [537](#)
  - rns\_double\_elt\_ptr, [540](#)
- Outmax
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [502](#)
- Outmin
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeq-Trait >, [502](#)
- p
  - HelperMod< Field, ElementCategories::MachineIntTag >, [473](#)
  - HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag >, [474](#)
  - HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag >, [475](#)
  - HelperMod< Field, FFLAS::ElementCategories::MachineFloatTag >, [475](#)
- pack\_lhs
  - FFLAS::details, [213](#)
- pack\_rhs
  - FFLAS::details, [213](#)
- PACKAGE
  - config.h, [806](#)
- PACKAGE\_BUGREPORT
  - config.h, [806](#)
- PACKAGE\_NAME
  - config.h, [806](#)
- PACKAGE\_STRING
  - config.h, [806](#)
- PACKAGE\_TARNAME
  - config.h, [807](#)
- PACKAGE\_URL
  - config.h, [807](#)
- PACKAGE\_VERSION
  - config.h, [807](#)
- PAR\_BLOCK
  - parallel.h, [1040](#)
- Parallel
  - Parallel< C, P >, [522](#)
- Parallel< C, P >, [522](#)
  - Cut, [522](#)
  - numthreads, [522](#)
  - operator<<, [523](#)
  - Parallel, [522](#)
  - Param, [522](#)
  - set\_numthreads, [523](#)
- parallel.h, [1039](#)
  - \_\_FFLASFFPACK\_SEQUENTIAL, [1040](#)
  - BARRIER, [1040](#)
  - BEGIN\_PARALLEL\_MAIN, [1041](#)
  - CHECK\_DEPENDENCIES, [1040](#)
  - COMMA, [1043](#)
  - CONSTREFERENCE, [1041](#)
  - END\_PARALLEL\_MAIN, [1041](#)
  - FOR1D, [1041](#)
  - FOR2D, [1042](#)
  - FORBLOCK1D, [1041](#)
  - FORBLOCK2D, [1042](#)
  - index\_t, [1040](#)
  - MAX\_THREADS, [1040](#)
  - MODE, [1043](#)

- NOSPLIT, [1045](#)
- NUM\_THREADS, [1040](#)
- NUMARGS, [1043](#)
- PAR\_BLOCK, [1040](#)
- PARFOR1D, [1042](#)
- PARFOR2D, [1043](#)
- PARFORBLOCK1D, [1042](#)
- PARFORBLOCK2D, [1043](#)
- PP\_ARG\_N, [1043](#)
- PP\_NARG\_, [1043](#)
- PP\_RSEQ\_N, [1045](#)
- READ, [1041](#)
- READWRITE, [1041](#)
- RETURNPARAM, [1043](#)
- splitt, [1045](#)
- SPLITTER, [1045](#)
- splitting\_0, [1045](#)
- splitting\_1, [1045](#)
- splitting\_2, [1045](#)
- splitting\_3, [1045](#)
- SYNCH\_GROUP, [1040](#)
- TASK, [1040](#)
- VALUE, [1041](#)
- WAIT, [1040](#)
- WRITE, [1041](#)
- Param
  - Parallel< C, P >, [522](#)
- PARFOR1D
  - parallel.h, [1042](#)
- PARFOR2D
  - parallel.h, [1043](#)
- PARFORBLOCK1D
  - parallel.h, [1042](#)
- PARFORBLOCK2D
  - parallel.h, [1043](#)
- parseArguments
  - FFLAS, [194](#)
- parseq
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [505](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [507](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, [508](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [511](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [512](#)
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [503](#)
  - TRSMHelper< ReclterTrait, ParSeqTrait >, [767](#)
- PBASECASE\_K
  - ffpack\_ppluq.inl, [943](#)
- pColumnEchelonForm
  - FFPACK, [323](#)
- pColumnEchelonForm\_modular\_double
  - ffpack.C, [999](#)
- pColumnEchelonForm\_modular\_float
  - ffpack.C, [1000](#)
- pColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [1001](#)
- pColumnRankProfile
  - FFPACK, [339](#)
- pDet
  - FFPACK, [334](#)
- perm
  - Info, [477](#), [478](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL >, [755](#)
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, [757](#)
- PermApplyS
  - FFPACK, [359](#)
- PermApplyS\_double
  - ffpack.C, [992](#)
  - ffpack\_c.h, [1015](#)
- PermApplyT
  - FFPACK, [361](#)
- PermApplyT\_double
  - ffpack.C, [992](#)
  - ffpack\_c.h, [1016](#)
- pfadd
  - FFLAS, [86](#)
- pfaddin
  - FFLAS, [87](#)
- pfgemm
  - FFLAS, [147](#), [190–192](#)
- pfgemm\_1D\_rec
  - FFLAS, [147](#)
- pfgemm\_2D\_rec
  - FFLAS, [148](#)
- pfgemm\_3D\_rec
  - FFLAS, [148](#)
- pfgemm\_3D\_rec2
  - FFLAS, [149](#)
- pfgemm\_variants.inl, [1046](#)
- pfgemv.inl, [1047](#)
- pfrand
  - FFLAS, [190](#)
- pfreduce
  - FFLAS, [118](#)
- pfspmm
  - FFLAS::sparse\_details, [239–241](#)
  - FFLAS::sparse\_details\_impl, [257](#), [264–266](#), [268](#), [269](#), [279](#), [280](#)
  - pfspmm\_dispatch
  - pfspmm\_mone
    - FFLAS::sparse\_details\_impl, [258](#)
  - pfspmm\_one
    - FFLAS::sparse\_details\_impl, [257](#), [258](#)
  - pfspmm\_zo
    - FFLAS::sparse\_details\_impl, [269](#), [270](#)
- pfspmv
  - FFLAS::sparse\_details, [242](#), [243](#)

- FFLAS::sparse\_details\_impl, [258](#), [259](#), [266](#), [270](#), [276](#), [280](#), [282](#)
- pfspmv\_mone
  - FFLAS::sparse\_details\_impl, [259](#), [260](#), [271](#), [276](#), [277](#), [283](#)
- pfspmv\_one
  - FFLAS::sparse\_details\_impl, [259](#), [260](#), [271](#), [276](#), [277](#), [283](#)
- pfspmv\_task
  - FFLAS::sparse\_details\_impl, [259](#)
- pfsb
  - FFLAS, [86](#)
- pfsbin
  - FFLAS, [87](#)
- pfzero
  - FFLAS, [190](#)
- PLUQ
  - FFPACK, [320](#), [321](#), [365](#), [369](#)
- pluq.C, [774](#), [775](#)
- PLUQ\_basecaseCrout
  - FFPACK, [364](#)
- PLUQ\_basecaseV2
  - FFPACK, [364](#)
- PLUQ\_basecaseV3
  - FFPACK, [364](#)
- PLUQ\_modular\_double
  - ffpack.C, [995](#)
  - ffpack\_c.h, [1019](#)
- PLUQtoEchelonPermutation
  - FFPACK, [349](#)
  - ffpack.C, [1009](#)
  - ffpack\_c.h, [1030](#)
- pm1
  - HelperFlag, [472](#)
- pMMH
  - TRSMHelper< ReclterTrait, ParSeqTrait >, [767](#)
- pow50rem
  - HelperMod< Field, ElementCategories::MachineIntTag >, [473](#)
- PP\_ARG\_N
  - parallel.h, [1043](#)
- PP\_NARG\_
  - parallel.h, [1043](#)
- PP\_RSEQ\_N
  - parallel.h, [1045](#)
- pPLUQ
  - FFPACK, [320](#)
- pRank
  - FFPACK, [332](#)
- preamble
  - FFLAS, [195](#)
- precompute\_cst
  - rns\_double, [529](#)
  - rns\_double\_extended, [542](#)
- pReducedColumnEchelonForm
  - FFPACK, [325](#)
- pReducedColumnEchelonForm\_modular\_double
  - ffpack.C, [1000](#)
- pReducedColumnEchelonForm\_modular\_float
  - ffpack.C, [1001](#)
- pReducedColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, [1002](#)
- pReducedRowEchelonForm
  - FFPACK, [326](#)
- pReducedRowEchelonForm\_modular\_double
  - ffpack.C, [1000](#)
- pReducedRowEchelonForm\_modular\_float
  - ffpack.C, [1001](#)
- pReducedRowEchelonForm\_modular\_int32\_t
  - ffpack.C, [1002](#)
- prefetch
  - FFLAS, [197](#)
- print
  - Failure, [443](#)
- printHelpMessage
  - args-parser.h, [1048](#)
- printPolynomial
  - test-charpoly-check.C, [1066](#)
- printvect
  - test-compressQ.C, [1068](#)
- pRowEchelonForm
  - FFPACK, [324](#)
- pRowEchelonForm\_modular\_double
  - ffpack.C, [999](#)
- pRowEchelonForm\_modular\_float
  - ffpack.C, [1000](#)
- pRowEchelonForm\_modular\_int32\_t
  - ffpack.C, [1001](#)
- pRowRankProfile
  - FFPACK, [338](#)
- PSeq
  - benchmark-fgemm-rns.C, [789](#)
- pSolve
  - FFPACK, [336](#)
- PTRSM\_HYBRID\_THRESHOLD
  - fflas\_ptrsm.inl, [873](#)
- PURE
  - fflas\_simd.h, [874](#)
- queryCacheSizes
  - FFLAS, [198](#)
- queryL1CacheSize
  - FFLAS, [198](#)
- queryTopLevelCacheSize
  - FFLAS, [198](#)
- RandInt
  - FFPACK, [385](#)
- RandIter
  - RNSInteger< RNS >::RandIter, [523](#)
  - RNSIntegerMod< RNS >::RandIter, [524](#)
- random
  - RNSInteger< RNS >::RandIter, [523](#), [524](#)
  - RNSIntegerMod< RNS >::RandIter, [525](#)
  - rnsRandIter< RNS >, [556](#)
- RandomIndexSubset
  - FFPACK, [387](#)

- RandomKrylovPrecond
  - FFPACK::Protected, 399
- RandomMatrix
  - FFPACK, 382, 383
- RandomMatrixWithDet
  - FFPACK, 393
- RandomMatrixWithRank
  - FFPACK, 385, 386
- RandomMatrixWithRankandRandomRPM
  - FFPACK, 391
- RandomMatrixWithRankandRPM
  - FFPACK, 388, 389
- RandomNullSpaceVector
  - FFPACK, 336, 350, 376
- RandomNullSpaceVector\_modular\_double
  - ffpack.C, 1005
  - ffpack\_c.h, 1026
- RandomPermutation
  - FFPACK, 387
- RandomRankProfileMatrix
  - FFPACK, 387
- RandomSymmetricMatrix
  - FFPACK, 385
- RandomSymmetricMatrixWithRankandRandomRPM
  - FFPACK, 392
- RandomSymmetricMatrixWithRankandRPM
  - FFPACK, 390
- RandomSymmetricRankProfileMatrix
  - FFPACK, 388
- RandomTriangularMatrix
  - FFPACK, 383, 384
- Rank
  - FFPACK, 332, 333, 374
- rank.C, 813
  - main, 813
- Rank\_modular\_double
  - ffpack.C, 1003
  - ffpack\_c.h, 1025
- RankProfileFromLU
  - FFPACK, 339
  - ffpack.C, 1006
  - ffpack\_c.h, 1027
- READ
  - parallel.h, 1041
- read\_field
  - Matio.h, 1056
- read\_sparse.h, 910
  - DNS\_BIN\_VER, 911
  - mask\_t, 911
- readDnsFormat
  - FFLAS, 158
- readMachineType
  - FFLAS, 157
- ReadMatrix
  - FFLAS, 195
- readMyMachineType< Field, mpz\_t >, 526
  - Element, 526
  - Element\_ptr, 526
  - operator(), 526
- readMyMachineType< Field, T >, 525
  - Element, 525
  - Element\_ptr, 526
  - operator(), 526
- readOrRandomMatrixWithRankAndRandomRPM
  - test-nullspace.C, 1109
- readSmsFormat
  - FFLAS, 156
- readSprFormat
  - FFLAS, 157
- READWRITE
  - parallel.h, 1041
- Rec\_Initialize
  - benchmark-pluq.C, 802
- RecInt, 403
- recLevel
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 505
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 507
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, 508
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, 511
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 512
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 502
- Recursive, 527
- reduce
  - FFLAS::vectorised, 289–291
  - rns\_double, 530
  - rns\_double\_extended, 543
  - RNSInteger< RNS >, 547
  - RNSIntegerMod< RNS >, 552
- reduce\_modp
  - RNSIntegerMod< RNS >, 553, 554
- reduce\_modp\_rnsmajor
  - RNSIntegerMod< RNS >, 554
- ReducedColumnEchelonForm
  - FFPACK, 324, 325, 371
- ReducedColumnEchelonForm\_modular\_double
  - ffpack.C, 997
  - ffpack\_c.h, 1021
- ReducedColumnEchelonForm\_modular\_float
  - ffpack.C, 998
  - ffpack\_c.h, 1022
- ReducedColumnEchelonForm\_modular\_int32\_t
  - ffpack.C, 999
  - ffpack\_c.h, 1022
- ReducedRowEchelonForm
  - FFPACK, 326, 327, 370
- ReducedRowEchelonForm2\_modular\_double
  - ffpack\_c.h, 1023

ReducedRowEchelonForm\_modular\_double  
     ffpack.C, 997  
     ffpack\_c.h, 1022  
 ReducedRowEchelonForm\_modular\_float  
     ffpack.C, 998  
     ffpack\_c.h, 1022  
 ReducedRowEchelonForm\_modular\_int32\_t  
     ffpack.C, 999  
     ffpack\_c.h, 1023  
 REF\_modular\_double  
     ffpack\_c.h, 1023  
 REGISTER\_TYPE\_NAME  
     test-simd.C, 1114, 1115  
 regression-check.C, 1064  
     check1, 1065  
     check2, 1065  
     check3, 1065  
     check4, 1065  
     checkZeroDimCharpoly, 1065  
     checkZeroDimMinPoly, 1065  
     gf2ModularBalanced, 1065  
     main, 1065  
 RETURNPARAM  
     parallel.h, 1043  
 ring  
     RNSInteger< RNS >::RandIter, 524  
     RNSIntegerMod< RNS >::RandIter, 525  
     rnsRandIter< RNS >, 556  
 rint< K >, 527  
 RNS, 51  
     benchmark-fgemm-rns.C, 788  
 rns  
     RNSInteger< RNS >, 546  
     RNSIntegerMod< RNS >, 550  
 rns-double-elt.h, 947  
 rns-double-recint.inl, 948  
     \_\_FFLASFFPACK\_field\_rns\_double\_recint\_INL,  
         948  
 rns-double.h, 948  
     ROUND\_DOWN, 949  
 rns-double.inl, 949  
     \_\_FFLASFFPACK\_field\_rns\_double\_INL, 949  
 rns-integer-mod.h, 949  
 rns-integer.h, 950  
 rns.h, 951  
 rns.inl, 951  
     \_\_FFLASFFPACK\_field\_rns\_INL, 951  
 rns\_double, 527  
     \_M, 532  
     \_MMi, 532  
     \_Mi, 532  
     \_basis, 531  
     \_basisMax, 531  
     \_crt\_in, 532  
     \_crt\_out, 532  
     \_field\_rns, 531  
     \_invbasis, 531  
     \_ldm, 532  
     \_mi\_sum, 532  
     \_negbasis, 531  
     \_pbits, 532  
     \_size, 532  
     BasisElement, 528  
     ConstElement\_ptr, 529  
     convert, 530, 531  
     convert\_transpose, 530  
     Element, 528  
     Element\_ptr, 528  
     init, 529–531  
     init\_transpose, 530  
     integer, 528  
     ModField, 528  
     precompute\_cst, 529  
     reduce, 530  
     rns\_double, 529  
 rns\_double\_elt, 532  
     \_alloc, 534  
     \_ptr, 534  
     \_stride, 534  
     ~rns\_double\_elt, 533  
     operator&, 533  
     rns\_double\_elt, 533  
 rns\_double\_elt\_cstptr, 534  
     \_alloc, 537  
     \_ptr, 537  
     \_stride, 537  
     operator!=, 536  
     operator<, 536  
     operator\*, 535  
     operator+, 536  
     operator++, 536  
     operator+=", 536  
     operator-, 536  
     operator--, 536  
     operator=, 536  
     operator=, 536  
     operator&, 535, 536  
     operator[], 535  
     other, 537  
     rns\_double\_elt\_cstptr, 535  
 rns\_double\_elt\_ptr, 537  
     \_alloc, 540  
     \_ptr, 540  
     \_stride, 540  
     operator!=, 539  
     operator<, 539  
     operator\*, 538  
     operator+, 539  
     operator++, 539  
     operator+=", 539  
     operator-, 539  
     operator--, 539  
     operator=, 539  
     operator=, 539  
     operator&, 538, 539  
     operator[], 538



- other, 540
  - rns\_double\_elt\_ptr, 538
- rns\_double\_extended, 540
  - \_M, 544
  - \_MMi, 544
  - \_Mi, 544
  - \_basis, 543
  - \_basisMax, 543
  - \_crt\_in, 544
  - \_crt\_out, 544
  - \_field\_rns, 544
  - \_invbasis, 543
  - \_ldm, 544
  - \_negbasis, 543
  - \_pbits, 544
  - \_size, 544
- BasisElement, 541
- ConstElement\_ptr, 541
- convert, 542, 543
- Element, 541
- Element\_ptr, 541
- init, 542, 543
- integer, 541
- ModField, 541
- precompute\_cst, 542
- reduce, 543
- rns\_double\_extended, 541, 542
- RNSElementTag, 544
- RNSInteger
  - RNSInteger< RNS >, 546
- RNSInteger< RNS >, 545
  - \_rns, 548
  - assign, 548
  - BasisElement, 546
  - cardinality, 547
  - characteristic, 547
  - ConstElement\_ptr, 546
  - convert, 547
  - Element, 546
  - Element\_ptr, 546
  - init, 547
  - integer, 546
  - isMOne, 547
  - isOne, 546
  - isZero, 547
  - mOne, 548
  - one, 548
  - reduce, 547
  - rns, 546
  - RNSInteger, 546
  - size, 546
  - write, 548
  - zero, 548
- RNSInteger< RNS >::RandIter, 523
  - operator(), 524
  - RandIter, 523
  - random, 523, 524
  - ring, 524
- RNSIntegerMod
  - RNSIntegerMod< RNS >, 550
- RNSIntegerMod< RNS >, 548
  - \_F, 554
  - \_Mi\_modp\_rns, 554
  - \_RNSdelayed, 555
  - \_iM\_modp\_rns, 554
  - \_p, 554
  - \_rns, 554
  - add, 552
  - areEqual, 553
  - assign, 552
  - axpyin, 553
  - BasisElement, 550
  - cardinality, 551
  - characteristic, 551
  - ConstElement\_ptr, 550
  - convert, 552
  - delayed, 550
  - Element, 550
  - Element\_ptr, 550
  - init, 551, 552
  - integer, 550
  - inv, 553
  - isMOne, 551
  - isOne, 551
  - isZero, 551
  - maxElement, 551
  - minElement, 551
  - ModField, 550
  - mOne, 555
  - mul, 553
  - neg, 553
  - one, 555
  - reduce, 552
  - reduce\_modp, 553, 554
  - reduce\_modp\_rnsmajor, 554
  - rns, 550
  - RNSIntegerMod, 550
  - size, 551
  - sub, 552
  - write, 553
  - write\_matrix, 553
  - write\_matrix\_long, 554
  - zero, 555
- RNSIntegerMod< RNS >::RandIter, 524
  - operator(), 525
  - RandIter, 524
  - random, 525
  - ring, 525
- RNSModulus
  - FFLAS::CuttingStrategy, 206
- rnsRandIter
  - rnsRandIter< RNS >, 555
- rnsRandIter< RNS >, 555
  - operator(), 556
  - random, 556
  - ring, 556



- rnsRandIter, [555](#)
- round
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [558](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [561](#)
  - Simd128\_impl< true, true, false, 2 >, [576](#)
  - Simd128\_impl< true, true, false, 4 >, [586](#)
  - Simd128\_impl< true, true, false, 8 >, [595](#)
  - Simd128\_impl< true, true, true, 2 >, [604](#)
  - Simd128\_impl< true, true, true, 4 >, [613](#)
  - Simd128\_impl< true, true, true, 8 >, [622](#)
  - Simd256\_impl< true, false, true, 8 >, [632](#)
  - Simd256\_impl< true, true, false, 2 >, [642](#)
  - Simd256\_impl< true, true, false, 4 >, [659](#)
  - Simd256\_impl< true, true, false, 8 >, [669](#)
  - Simd256\_impl< true, true, true, 2 >, [678](#)
  - Simd256\_impl< true, true, true, 4 >, [689](#), [694](#)
  - Simd256\_impl< true, true, true, 8 >, [703](#)
  - Simd512\_impl< true, false, true, 8 >, [712](#)
  - Simd512\_impl< true, true, false, 8 >, [722](#)
  - Simd512\_impl< true, true, true, 8 >, [732](#)
- ROUND\_DOWN
  - fflas\_sparse.h, [886](#)
  - rns-double.h, [949](#)
- Row, [556](#)
- row
  - Coo< Field >, [431](#)
  - Coo< ValT, IdxT >, [429](#), [432](#)
  - Sparse< \_Field, SparseMatrix\_t::COO >, [737](#)
  - Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [739](#)
- rowblockindex
  - ForStrategy2D< blocksize\_t, Cut, Param >, [463](#)
- rowBlockSize
  - ForStrategy2D< blocksize\_t, Cut, Param >, [463](#)
- rowdim
  - StatsMatrix, [759](#)
- RowEchelonForm
  - FFPACK, [323](#), [324](#), [370](#)
- RowEchelonForm\_modular\_double
  - ffpack.C, [996](#)
  - ffpack\_c.h, [1020](#)
- RowEchelonForm\_modular\_float
  - ffpack.C, [997](#)
  - ffpack\_c.h, [1021](#)
- RowEchelonForm\_modular\_int32\_t
  - ffpack.C, [998](#)
  - ffpack\_c.h, [1021](#)
- rownumblocks
  - ForStrategy2D< blocksize\_t, Cut, Param >, [462](#)
- RowRankProfile
  - FFPACK, [337](#), [338](#), [376](#)
- RowRankProfile\_modular\_double
  - ffpack.C, [1005](#)
  - ffpack\_c.h, [1026](#)
- RowRankProfileSubmatrix
  - FFPACK, [342](#), [377](#)
- RowRankProfileSubmatrix\_modular\_double
  - ffpack.C, [1006](#)
  - ffpack\_c.h, [1028](#)
- RowRankProfileSubmatrixIndices
  - FFPACK, [340](#), [377](#)
- RowRankProfileSubmatrixIndices\_modular\_double
  - ffpack.C, [1006](#)
  - ffpack\_c.h, [1027](#)
- ruint< K >, [556](#)
- run\_with\_field
  - benchmark-charpoly.C, [778](#)
  - benchmark-fdot.C, [786](#)
  - test-charpoly.C, [1067](#)
  - test-echelon.C, [1072](#)
  - test-fdot.C, [1074](#)
  - test-fgemm-check.C, [1076](#)
  - test-fgemm.C, [1078](#)
  - test-fgemv.C, [1080](#)
  - test-fger.C, [1082](#)
  - test-fgesv.C, [1083](#)
  - test-finit.C, [1084](#)
  - test-fsyr2k.C, [1087](#)
  - test-fsyrk.C, [1089](#)
  - test-fsytrf.C, [1090](#)
  - test-ftrmm.C, [1091](#)
  - test-ftrmv.C, [1093](#)
  - test-ftrsm.C, [1095](#)
  - test-ftrssyr2k.C, [1096](#)
  - test-ftrstr.C, [1097](#)
  - test-ftrsv.C, [1098](#)
  - test-ftrtri.C, [1099](#)
  - test-io.C, [1101](#)
  - test-lu.C, [1105](#)
  - test-minpoly.C, [1107](#)
  - test-nullspace.C, [1109](#)
  - test-rankprofiles.C, [1112](#)
  - test-solve.C, [1117](#)
- run\_with\_Integer
  - test-fdot.C, [1075](#)
- saxpy\_
  - config-blas.h, [821](#)
- ScalAndReduce
  - FFLAS::Protected, [222](#)
- scalar\_t
  - FieldSimd< \_Field >, [445](#)
  - NoSimd< T >, [521](#)
  - Simd128\_impl< true, true, false, 2 >, [569](#)
  - Simd128\_impl< true, true, false, 4 >, [579](#)
  - Simd128\_impl< true, true, false, 8 >, [589](#)
  - Simd128\_impl< true, true, true, 2 >, [599](#)
  - Simd128\_impl< true, true, true, 4 >, [608](#)
  - Simd128\_impl< true, true, true, 8 >, [616](#)
  - Simd256\_impl< true, false, true, 8 >, [627](#)
  - Simd256\_impl< true, true, false, 2 >, [635](#)
  - Simd256\_impl< true, true, false, 4 >, [646](#)
  - Simd256\_impl< true, true, false, 8 >, [662](#)
  - Simd256\_impl< true, true, true, 2 >, [672](#)
  - Simd256\_impl< true, true, true, 4 >, [682](#), [683](#)

- Simd256\_impl< true, true, true, 8 >, [697](#)
- Simd512\_impl< true, false, true, 8 >, [707](#)
- Simd512\_impl< true, true, false, 8 >, [715](#)
- Simd512\_impl< true, true, true, 8 >, [726](#)
- ScalFunctions< Element, Enable >, [556](#)
- ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [557](#)
  - add, [558](#)
  - addin, [558](#)
  - ceil, [558](#)
  - div, [559](#)
  - eq, [560](#)
  - floor, [558](#)
  - fmadd, [559](#)
  - fmaddin, [559](#)
  - fmsub, [559](#)
  - fmsubin, [559](#)
  - fnmadd, [559](#)
  - fnmaddin, [559](#)
  - greater, [560](#)
  - greater\_eq, [560](#)
  - lesser, [560](#)
  - lesser\_eq, [560](#)
  - mul, [558](#)
  - mulin, [559](#)
  - round, [558](#)
  - sub, [558](#)
  - subin, [558](#)
  - vand, [557](#)
  - vandnot, [558](#)
  - vor, [557](#)
  - vxor, [557](#)
  - zero, [557](#)
- ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [560](#)
  - add, [562](#)
  - addin, [562](#)
  - eq, [565](#)
  - fmadd, [563](#)
  - fmaddin, [563](#)
  - fmaddx, [563](#)
  - fmaddxin, [563](#)
  - fmsub, [563](#)
  - fmsubin, [563](#)
  - fmsubx, [564](#)
  - fmsubxin, [564](#)
  - fnmadd, [564](#)
  - fnmaddin, [564](#)
  - fnmaddx, [564](#)
  - fnmaddxin, [564](#)
  - greater, [565](#)
  - greater\_eq, [565](#)
  - lesser, [565](#)
  - lesser\_eq, [565](#)
  - mul, [562](#)
  - mulhi, [563](#)
  - mullo, [562](#)
  - mulx, [563](#)
  - round, [561](#)
  - sll, [565](#)
  - sra, [564](#), [565](#)
  - srl, [565](#)
  - sub, [562](#)
  - subin, [562](#)
  - vand, [561](#)
  - vandnot, [562](#)
  - vor, [561](#)
  - vxor, [562](#)
  - zero, [561](#)
- scalp
  - FFLAS::vectorised, [291](#)
  - FFLAS::vectorised::unswitch, [293](#)
- schedule\_bini.inl, [842](#)
  - \_\_FFLASFFPACK\_fgemm\_bini\_INL, [842](#)
- schedule\_winograd.inl, [842](#)
  - \_\_FFLASFFPACK\_fgemm\_winograd\_INL, [843](#)
- schedule\_winograd\_acc.inl, [843](#)
  - \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_INL, [843](#)
- schedule\_winograd\_acc\_ip.inl, [844](#)
  - \_\_FFLASFFPACK\_fgemm\_winograd\_acc\_ip\_INL, [844](#)
- schedule\_winograd\_ip.inl, [844](#)
  - \_\_FFLASFFPACK\_fgemm\_winograd\_ip\_INL, [845](#)
- scopy\_
  - config-blas.h, [823](#)
- sdot\_
  - config-blas.h, [821](#)
- second\_component
  - Compose< H1, H2 >, [422](#)
- Self
  - Coo< ValT, IdxT >, [428](#), [432](#)
- Self\_t
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [503](#)
  - MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [506](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< Dest >, ParSeqTrait >, [508](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, [509](#)
  - MMHelper< Field, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, [511](#)
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, [499](#)
  - Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [752](#)
- SELL
  - FFLAS, [83](#)
- sell.h, [911](#)
- sell\_pspmv.inl, [912](#)
  - \_\_FFLASFFPACK\_fflas\_sparse\_sell\_pspmv\_INL,

- 912
- sell\_spmv.inl, 912
  - \_\_FFLASFFPACK\_fflas\_sparse\_sell\_spmv\_INL, 913
- sell\_utils.inl, 913
  - \_\_FFLASFFPACK\_fflas\_sparse\_sell\_utils\_INL, 914
- SELL\_ZO
  - FFLAS, 83
- Sequential, 566
  - numthreads, 566
  - operator<<, 566
  - Sequential, 566
- set
  - Simd128\_impl< true, true, false, 2 >, 570, 573
  - Simd128\_impl< true, true, false, 4 >, 580, 582
  - Simd128\_impl< true, true, false, 8 >, 589, 592
  - Simd128\_impl< true, true, true, 2 >, 599
  - Simd128\_impl< true, true, true, 4 >, 608
  - Simd128\_impl< true, true, true, 8 >, 617
  - Simd256\_impl< true, false, true, 8 >, 628
  - Simd256\_impl< true, true, false, 2 >, 635, 638
  - Simd256\_impl< true, true, false, 4 >, 647, 650, 653
  - Simd256\_impl< true, true, false, 8 >, 663, 666
  - Simd256\_impl< true, true, true, 2 >, 673
  - Simd256\_impl< true, true, true, 4 >, 683, 689
  - Simd256\_impl< true, true, true, 8 >, 698
  - Simd512\_impl< true, false, true, 8 >, 708
  - Simd512\_impl< true, true, false, 8 >, 715, 718
  - Simd512\_impl< true, true, true, 8 >, 726
- set1
  - Simd128\_impl< true, true, false, 2 >, 570, 573
  - Simd128\_impl< true, true, false, 4 >, 580, 582
  - Simd128\_impl< true, true, false, 8 >, 589, 592
  - Simd128\_impl< true, true, true, 2 >, 599
  - Simd128\_impl< true, true, true, 4 >, 608
  - Simd128\_impl< true, true, true, 8 >, 617
  - Simd256\_impl< true, false, true, 8 >, 628
  - Simd256\_impl< true, true, false, 2 >, 635, 638
  - Simd256\_impl< true, true, false, 4 >, 647, 649, 653
  - Simd256\_impl< true, true, false, 8 >, 663, 666
  - Simd256\_impl< true, true, true, 2 >, 673
  - Simd256\_impl< true, true, true, 4 >, 683, 689
  - Simd256\_impl< true, true, true, 8 >, 698
  - Simd512\_impl< true, false, true, 8 >, 708
  - Simd512\_impl< true, true, false, 8 >, 715, 718
  - Simd512\_impl< true, true, true, 8 >, 726
- set\_numthreads
  - Parallel< C, P >, 523
- setErrorStream
  - Failure, 442
- setNorm
  - MMHelper< FFPACK::RNSInteger< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 504
- MMHelper< FFPACK::RNSIntegerMod< E >, AlgoTrait, ModeCategories::DefaultTag, ParSeqTrait >, 506
- MMHelper< Field, AlgoTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeqTrait >, 510
- setOutBounds
  - MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >, 501
- sgemm\_
  - config-blas.h, 825
- sgemv\_
  - config-blas.h, 822
- sger\_
  - config-blas.h, 823
- shuffle
  - Simd128\_impl< true, true, false, 2 >, 574
  - Simd128\_impl< true, true, false, 4 >, 583
  - Simd128\_impl< true, true, false, 8 >, 593
  - Simd128\_impl< true, true, true, 2 >, 601
  - Simd128\_impl< true, true, true, 4 >, 609
  - Simd128\_impl< true, true, true, 8 >, 618
  - Simd256\_impl< true, true, false, 2 >, 640
  - Simd256\_impl< true, true, false, 4 >, 655
  - Simd256\_impl< true, true, false, 8 >, 667
  - Simd256\_impl< true, true, true, 2 >, 674
  - Simd256\_impl< true, true, true, 4 >, 684, 691
  - Simd256\_impl< true, true, true, 8 >, 699
  - Simd512\_impl< true, false, true, 8 >, 709
  - Simd512\_impl< true, true, false, 8 >, 720
  - Simd512\_impl< true, true, true, 8 >, 728
- shuffle\_twice
  - Simd256\_impl< true, true, false, 4 >, 654
  - Simd256\_impl< true, true, true, 4 >, 684, 691
- sigma
  - Sparse< \_Field, SparseMatrix\_t::SELL >, 754
  - Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 757
- signbits
  - Simd128\_impl< true, true, false, 8 >, 596
  - Simd128\_impl< true, true, true, 8 >, 622
  - Simd256\_impl< true, true, false, 8 >, 670
  - Simd256\_impl< true, true, true, 8 >, 704
  - Simd512\_impl< true, true, false, 8 >, 723
  - Simd512\_impl< true, true, true, 8 >, 732
- Simd
  - fflas\_simd.h, 875
- simd
  - FieldSimd< \_Field >, 445
- SIMD wrapper, 50
- simd.dox, 875
- Simd128
  - simd128.inl, 875
- simd128.inl, 875
  - \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_INL, 875
  - Simd128, 875
- simd128\_double.inl, 875

- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_double\_INLunpackhi, 574
  - 876
- simd128\_float.inl, 876
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_float\_INL, 876
- Simd128\_impl< ArithType, Int, Signed, Size >, 567
- Simd128\_impl< true, false, true, 4 >, 567
  - type\_string, 567
- Simd128\_impl< true, false, true, 8 >, 567
  - type\_string, 567
- Simd128\_impl< true, true, false, 2 >, 568
  - add, 574
  - addin, 574
  - alignment, 577
  - blend, 574
  - compliant, 572
  - eq, 576
  - fmadd, 575
  - fmaddin, 575
  - fmaddx, 571
  - fmaddxin, 572
  - fmsub, 575
  - fmsubin, 576
  - fmsubx, 572
  - fmsubxin, 572
  - fnmadd, 575
  - fnmaddin, 575
  - fnmaddx, 572
  - fnmaddxin, 572
  - gather, 570, 573
  - greater, 571
  - greater\_eq, 571
  - hadd\_to\_scal, 572
  - lesser, 571
  - lesser\_eq, 571
  - load, 570, 573
  - loadu, 570, 573
  - mod, 576
  - mul, 575
  - mulhi, 571
  - mullo, 575
  - mulx, 571
  - round, 576
  - scalar\_t, 569
  - set, 570, 573
  - set1, 570, 573
  - shuffle, 574
  - sll, 574
  - sll128, 576
  - sra, 571
  - srl, 574
  - srl128, 576
  - store, 570, 573
  - storeu, 570, 573
  - stream, 570, 573
  - sub, 574
  - subin, 575
  - type\_string, 576
  - unpacklo, 574
  - valid, 572
  - vand, 577
  - vandnot, 577
  - vect\_size, 577
  - vect\_t, 569
  - vor, 577
  - vxor, 577
  - zero, 576
- Simd128\_impl< true, true, false, 2 >::Converter, 422
  - t, 422
  - v, 422
- Simd128\_impl< true, true, false, 4 >, 577
  - add, 584
  - addin, 584
  - alignment, 587
  - blend, 584
  - compliant, 582
  - eq, 585
  - fmadd, 585
  - fmaddin, 585
  - fmaddx, 581
  - fmaddxin, 581
  - fmsub, 585
  - fmsubin, 585
  - fmsubx, 582
  - fmsubxin, 582
  - fnmadd, 585
  - fnmaddin, 585
  - fnmaddx, 582
  - fnmaddxin, 582
  - gather, 580, 583
  - greater, 581
  - greater\_eq, 581
  - hadd\_to\_scal, 582
  - lesser, 581
  - lesser\_eq, 581
  - load, 580, 583
  - loadu, 580, 583
  - mod, 586
  - mul, 585
  - mulhi, 581
  - mullo, 584
  - mulx, 581
  - round, 586
  - scalar\_t, 579
  - set, 580, 582
  - set1, 580, 582
  - shuffle, 583
  - sll, 583
  - sll128, 586
  - sra, 580
  - srl, 583
  - srl128, 586
  - store, 580, 583
  - storeu, 580, 583
  - stream, 580, 583

- sub, [584](#)
- subin, [584](#)
- type\_string, [586](#)
- unpackhi, [584](#)
- unpacklo, [584](#)
- valid, [582](#)
- vand, [586](#)
- vandnot, [587](#)
- vect\_size, [587](#)
- vect\_t, [579](#)
- vor, [586](#)
- vxor, [587](#)
- zero, [586](#)
- Simd128\_impl< true, true, false, 4 >::Converter, [423](#)
  - t, [423](#)
  - v, [423](#)
- Simd128\_impl< true, true, false, 8 >, [587](#)
  - add, [594](#)
  - addin, [594](#)
  - alignment, [597](#)
  - blend, [594](#)
  - compliant, [592](#)
  - eq, [595](#)
  - fmadd, [594](#)
  - fmaddin, [594](#)
  - fmaddx, [591](#)
  - fmaddxin, [591](#)
  - fmsub, [595](#)
  - fmsubin, [595](#)
  - fmsubx, [591](#)
  - fmsubxin, [592](#), [595](#)
  - fnmadd, [595](#)
  - fnmaddin, [595](#)
  - fnmaddx, [591](#)
  - fnmaddxin, [591](#)
  - gather, [589](#), [592](#)
  - get, [592](#)
  - greater, [590](#)
  - greater\_eq, [590](#)
  - hadd\_to\_scal, [592](#)
  - lesser, [590](#)
  - lesser\_eq, [591](#)
  - load, [590](#), [592](#)
  - loadu, [590](#), [593](#)
  - mask\_high, [595](#)
  - mod, [596](#)
  - mul, [594](#)
  - mulhi\_fast, [596](#)
  - mullo, [591](#)
  - mulx, [591](#)
  - round, [595](#)
  - scalar\_t, [589](#)
  - set, [589](#), [592](#)
  - set1, [589](#), [592](#)
  - shuffle, [593](#)
  - signbits, [596](#)
  - sll, [593](#)
  - sll128, [596](#)
  - sra, [590](#)
  - srl, [593](#)
  - srl128, [596](#)
  - store, [590](#), [593](#)
  - storeu, [590](#), [593](#)
  - stream, [590](#), [593](#)
  - sub, [594](#)
  - subin, [594](#)
  - type\_string, [596](#)
  - unpackhi, [593](#)
  - unpacklo, [593](#)
  - valid, [592](#)
  - vand, [596](#)
  - vandnot, [597](#)
  - vect\_size, [597](#)
  - vect\_t, [589](#)
  - vor, [596](#)
  - vxor, [597](#)
  - zero, [596](#)
- Simd128\_impl< true, true, false, 8 >::Converter, [423](#)
  - t, [423](#)
  - v, [423](#)
- Simd128\_impl< true, true, true, 2 >, [597](#)
  - add, [601](#)
  - addin, [601](#)
  - alignment, [606](#)
  - blend, [601](#)
  - compliant, [599](#)
  - eq, [604](#)
  - fmadd, [602](#)
  - fmaddin, [602](#)
  - fmaddx, [602](#)
  - fmaddxin, [602](#)
  - fmsub, [603](#)
  - fmsubin, [603](#)
  - fmsubx, [603](#)
  - fmsubxin, [604](#)
  - fnmadd, [603](#)
  - fnmaddin, [603](#)
  - fnmaddx, [603](#)
  - fnmaddxin, [603](#)
  - gather, [600](#)
  - greater, [604](#)
  - greater\_eq, [604](#)
  - hadd\_to\_scal, [604](#)
  - lesser, [604](#)
  - lesser\_eq, [604](#)
  - load, [600](#)
  - loadu, [600](#)
  - mod, [604](#)
  - mul, [602](#)
  - mulhi, [602](#)
  - mullo, [602](#)
  - mulx, [602](#)
  - round, [604](#)
  - scalar\_t, [599](#)
  - set, [599](#)
  - set1, [599](#)

- shuffle, [601](#)
- sll, [600](#)
- sll128, [605](#)
- sra, [601](#)
- srl, [600](#)
- srl128, [605](#)
- store, [600](#)
- storeu, [600](#)
- stream, [600](#)
- sub, [601](#)
- subin, [601](#)
- type\_string, [605](#)
- unpackhi, [601](#)
- unpacklo, [601](#)
- valid, [599](#)
- vand, [605](#)
- vandnot, [605](#)
- vect\_size, [606](#)
- vect\_t, [599](#)
- vor, [605](#)
- vxor, [605](#)
- zero, [605](#)
- Simd128\_impl< true, true, true, 2 >::Converter, [423](#)
- t, [424](#)
- v, [424](#)
- Simd128\_impl< true, true, true, 4 >, [606](#)
- add, [610](#)
- addin, [610](#)
- alignment, [614](#)
- blend, [610](#)
- compliant, [608](#)
- eq, [612](#)
- fmadd, [611](#)
- fmaddin, [611](#)
- fmaddx, [611](#)
- fmaddxin, [611](#)
- fmsub, [612](#)
- fmsubin, [612](#)
- fmsubx, [612](#)
- fmsubxin, [612](#)
- fnmadd, [611](#)
- fnmaddin, [611](#)
- fnmaddx, [612](#)
- fnmaddxin, [612](#)
- gather, [608](#)
- greater, [613](#)
- greater\_eq, [613](#)
- hadd\_to\_scal, [613](#)
- lesser, [613](#)
- lesser\_eq, [613](#)
- load, [608](#)
- loadu, [609](#)
- mod, [613](#)
- mul, [610](#)
- mulhi, [611](#)
- mullo, [610](#)
- mulx, [611](#)
- round, [613](#)
- scalar\_t, [608](#)
- set, [608](#)
- set1, [608](#)
- shuffle, [609](#)
- sll, [609](#)
- sll128, [614](#)
- sra, [609](#)
- srl, [609](#)
- srl128, [614](#)
- store, [609](#)
- storeu, [609](#)
- stream, [609](#)
- sub, [610](#)
- subin, [610](#)
- type\_string, [613](#)
- unpackhi, [610](#)
- unpacklo, [609](#)
- valid, [608](#)
- vand, [614](#)
- vandnot, [614](#)
- vect\_size, [614](#)
- vect\_t, [608](#)
- vor, [614](#)
- vxor, [614](#)
- zero, [613](#)
- Simd128\_impl< true, true, true, 4 >::Converter, [424](#)
- t, [424](#)
- v, [424](#)
- Simd128\_impl< true, true, true, 8 >, [615](#)
- add, [618](#)
- addin, [619](#)
- alignment, [623](#)
- blend, [618](#)
- compliant, [617](#)
- eq, [621](#)
- fmadd, [619](#)
- fmaddin, [619](#)
- fmaddx, [620](#)
- fmaddxin, [620](#)
- fmsub, [620](#)
- fmsubin, [621](#)
- fmsubx, [621](#)
- fmsubxin, [621](#)
- fnmadd, [620](#)
- fnmaddin, [620](#)
- fnmaddx, [620](#)
- fnmaddxin, [620](#)
- gather, [617](#)
- get, [617](#)
- greater, [621](#)
- greater\_eq, [621](#)
- hadd\_to\_scal, [622](#)
- lesser, [621](#)
- lesser\_eq, [621](#)
- load, [617](#)
- loadu, [617](#)
- mask\_high, [622](#)
- mod, [622](#)

- mul, [619](#)
- mulhi\_fast, [622](#)
- mullo, [619](#)
- mulx, [619](#)
- round, [622](#)
- scalar\_t, [616](#)
- set, [617](#)
- set1, [617](#)
- shuffle, [618](#)
- signbits, [622](#)
- sll, [618](#)
- sll128, [622](#)
- sra, [618](#)
- srl, [618](#)
- srl128, [623](#)
- store, [617](#)
- storeu, [617](#)
- stream, [618](#)
- sub, [619](#)
- subin, [619](#)
- type\_string, [622](#)
- unpackhi, [618](#)
- unpacklo, [618](#)
- valid, [617](#)
- vand, [623](#)
- vandnot, [623](#)
- vect\_size, [623](#)
- vect\_t, [616](#)
- vor, [623](#)
- vxor, [623](#)
- zero, [622](#)
- Simd128\_impl< true, true, true, 8 >::Converter, [424](#)
- t, [424](#)
- v, [424](#)
- simd128\_int16.inl, [876](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int16\_INL, [876](#)
- simd128\_int32.inl, [876](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int32\_INL, [877](#)
- simd128\_int64.inl, [877](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd128\_int64\_INL, [877](#)
- vect\_t, [877](#)
- Simd128fp\_base, [623](#)
- type\_string, [624](#)
- Simd128i\_base, [624](#)
- sll128, [625](#)
- srl128, [625](#)
- type\_string, [625](#)
- vand, [625](#)
- vandnot, [625](#)
- vect\_t, [624](#)
- vor, [625](#)
- vxor, [625](#)
- zero, [625](#)
- Simd256
- simd256.inl, [878](#)
- simd256.inl, [877](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_INL, [878](#)
- Simd256, [878](#)
- simd256\_double.inl, [878](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_double\_INL, [878](#)
- simd256\_float.inl, [878](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_float\_INL, [878](#)
- Simd256\_impl< ArithType, Int, Signed, Size >, [626](#)
- Simd256\_impl< true, false, true, 4 >, [626](#)
- Simd256\_impl< true, false, true, 8 >, [626](#)
- add, [629](#)
- addin, [629](#)
- alignment, [633](#)
- blend, [629](#)
- blendv, [629](#)
- ceil, [632](#)
- compliant, [628](#)
- div, [630](#)
- eq, [631](#)
- floor, [632](#)
- fmadd, [630](#)
- fmaddin, [630](#)
- fmsub, [631](#)
- fmsubin, [631](#)
- fnmadd, [630](#)
- fnmaddin, [630](#)
- gather, [628](#)
- greater, [631](#)
- greater\_eq, [631](#)
- hadd, [632](#)
- hadd\_to\_scal, [632](#)
- lesser, [631](#)
- lesser\_eq, [631](#)
- load, [628](#)
- loadu, [628](#)
- mod, [632](#)
- mul, [630](#)
- mulin, [630](#)
- round, [632](#)
- scalar\_t, [627](#)
- set, [628](#)
- set1, [628](#)
- store, [628](#)
- storeu, [628](#)
- stream, [629](#)
- sub, [629](#)
- subin, [630](#)
- unpackhi\_twice, [629](#)
- unpacklo\_twice, [629](#)
- valid, [627](#)
- vand, [631](#)
- vandnot, [632](#)
- vect\_size, [633](#)
- vect\_t, [627](#)
- vor, [632](#)

- vxor, [632](#)
- zero, [628](#)
- Simd256\_impl< true, true, false, 2 >, [633](#)
  - add, [641](#)
  - addin, [641](#)
  - alignment, [643](#)
  - blend\_twice, [640](#)
  - compliant, [638](#)
  - eq, [642](#)
  - fmadd, [641](#)
  - fmaddin, [641](#)
  - fmaddx, [637](#)
  - fmaddxin, [637](#)
  - fmsub, [642](#)
  - fmsubin, [642](#)
  - fmsubx, [638](#)
  - fmsubxin, [638](#)
  - fnmadd, [641](#)
  - fnmaddin, [642](#)
  - fnmaddx, [638](#)
  - fnmaddxin, [638](#)
  - gather, [636](#), [639](#)
  - greater, [637](#)
  - greater\_eq, [637](#)
  - hadd\_to\_scal, [638](#)
  - half\_t, [635](#)
  - lesser, [637](#)
  - lesser\_eq, [637](#)
  - load, [636](#), [639](#)
  - loadu, [636](#), [639](#)
  - mod, [642](#)
  - mul, [641](#)
  - mulhi, [637](#)
  - mullo, [641](#)
  - mulx, [637](#)
  - round, [642](#)
  - scalar\_t, [635](#)
  - set, [635](#), [638](#)
  - set1, [635](#), [638](#)
  - shuffle, [640](#)
  - simdHalf, [635](#)
  - sll, [639](#)
  - sra, [636](#)
  - srl, [640](#)
  - store, [636](#), [639](#)
  - storeu, [636](#), [639](#)
  - stream, [636](#), [639](#)
  - sub, [641](#)
  - subin, [641](#)
  - type\_string, [642](#)
  - unpackhi, [640](#)
  - unpackhi\_twice, [640](#)
  - unpacklo, [640](#)
  - unpacklo\_twice, [640](#)
  - unpacklohi, [640](#)
  - valid, [638](#)
  - vect\_size, [643](#)
  - vect\_t, [635](#)
- zero, [643](#)
- Simd256\_impl< true, true, false, 2 >::Converter, [425](#)
  - t, [425](#)
  - v, [425](#)
- Simd256\_impl< true, true, false, 4 >, [643](#)
  - add, [656](#)
  - addin, [656](#)
  - alignment, [660](#)
  - blend, [655](#)
  - compliant, [652](#)
  - eq, [659](#)
  - fmadd, [657](#)
  - fmaddin, [657](#)
  - fmaddx, [649](#), [651](#)
  - fmaddxin, [649](#), [651](#)
  - fmsub, [658](#)
  - fmsubin, [658](#)
  - fmsubx, [649](#), [652](#)
  - fmsubxin, [649](#), [652](#)
  - fnmadd, [657](#), [658](#)
  - fnmaddin, [658](#)
  - fnmaddx, [649](#), [652](#)
  - fnmaddxin, [649](#), [652](#)
  - gather, [647](#), [650](#), [653](#)
  - greater, [648](#), [651](#)
  - greater\_eq, [648](#), [651](#)
  - hadd\_to\_scal, [649](#), [652](#)
  - half\_t, [646](#), [647](#)
  - lesser, [648](#), [651](#)
  - lesser\_eq, [648](#), [651](#)
  - load, [647](#), [650](#), [653](#)
  - loadu, [647](#), [650](#), [653](#)
  - mod, [659](#)
  - mul, [657](#)
  - mulhi, [648](#), [651](#)
  - mullo, [656](#), [657](#)
  - mulx, [648](#), [651](#)
  - round, [659](#)
  - scalar\_t, [646](#)
  - set, [647](#), [650](#), [653](#)
  - set1, [647](#), [649](#), [653](#)
  - shuffle, [655](#)
  - shuffle\_twice, [654](#)
  - simdHalf, [646](#)
  - sll, [654](#)
  - sra, [648](#), [650](#)
  - srl, [654](#)
  - store, [647](#), [650](#), [654](#)
  - storeu, [647](#), [650](#), [654](#)
  - stream, [648](#), [650](#), [654](#)
  - sub, [656](#)
  - subin, [656](#)
  - type\_string, [659](#)
  - unpackhi, [655](#)
  - unpackhi\_twice, [655](#)
  - unpacklo, [655](#)
  - unpacklo\_twice, [655](#)
  - unpacklohi, [655](#)



- valid, [652](#)
- vand, [660](#)
- vandnot, [660](#)
- vect\_size, [660](#)
- vect\_t, [646](#)
- vor, [660](#)
- vxor, [660](#)
- zero, [660](#)
- Simd256\_impl< true, true, false, 4 >::Converter, [425](#)
  - t, [425](#)
  - v, [425](#)
- Simd256\_impl< true, true, false, 8 >, [661](#)
  - add, [668](#)
  - addin, [668](#)
  - alignment, [670](#)
  - blend, [668](#)
  - compliant, [666](#)
  - eq, [669](#)
  - fmadd, [668](#)
  - fmaddin, [669](#)
  - fmaddx, [665](#)
  - fmaddxin, [665](#)
  - fmsub, [669](#)
  - fmsubin, [669](#)
  - fmsubx, [665](#)
  - fmsubxin, [665](#)
  - fnmadd, [669](#)
  - fnmaddin, [669](#)
  - fnmaddx, [665](#)
  - fnmaddxin, [665](#)
  - gather, [663](#), [666](#)
  - get, [666](#)
  - greater, [664](#)
  - greater\_eq, [664](#)
  - hadd\_to\_scal, [665](#)
  - half\_t, [663](#)
  - lesser, [664](#)
  - lesser\_eq, [664](#)
  - load, [663](#), [666](#)
  - loadu, [663](#), [666](#)
  - mask\_high, [669](#)
  - mod, [670](#)
  - mul, [668](#)
  - mulhi\_fast, [670](#)
  - mullo, [664](#)
  - mulx, [665](#)
  - round, [669](#)
  - scalar\_t, [662](#)
  - set, [663](#), [666](#)
  - set1, [663](#), [666](#)
  - shuffle, [667](#)
  - signbits, [670](#)
  - simdHalf, [663](#)
  - sll, [667](#)
  - sra, [664](#)
  - srl, [667](#)
  - store, [663](#), [666](#)
  - storeu, [664](#), [667](#)
  - stream, [664](#), [667](#)
  - sub, [668](#)
  - subin, [668](#)
  - type\_string, [670](#)
  - unpackhi, [667](#)
  - unpackhi\_twice, [667](#)
  - unpacklo, [667](#)
  - unpacklo\_twice, [667](#)
  - unpacklohi, [668](#)
  - valid, [666](#)
  - vect\_size, [670](#)
  - vect\_t, [663](#)
  - zero, [670](#)
- Simd256\_impl< true, true, false, 8 >::Converter, [425](#)
  - t, [426](#)
  - v, [425](#)
- Simd256\_impl< true, true, true, 2 >, [671](#)
  - add, [675](#)
  - addin, [675](#)
  - alignment, [679](#)
  - blend\_twice, [675](#)
  - compliant, [673](#)
  - eq, [678](#)
  - fmadd, [676](#)
  - fmaddin, [676](#)
  - fmaddx, [676](#)
  - fmaddxin, [676](#)
  - fmsub, [677](#)
  - fmsubin, [677](#)
  - fmsubx, [677](#)
  - fmsubxin, [678](#)
  - fnmadd, [677](#)
  - fnmaddin, [677](#)
  - fnmaddx, [677](#)
  - fnmaddxin, [677](#)
  - gather, [673](#)
  - greater, [678](#)
  - greater\_eq, [678](#)
  - hadd\_to\_scal, [678](#)
  - half\_t, [672](#)
  - lesser, [678](#)
  - lesser\_eq, [678](#)
  - load, [673](#)
  - loadu, [673](#)
  - mod, [678](#)
  - mul, [676](#)
  - mulhi, [676](#)
  - mullo, [676](#)
  - mulx, [676](#)
  - round, [678](#)
  - scalar\_t, [672](#)
  - set, [673](#)
  - set1, [673](#)
  - shuffle, [674](#)
  - simdHalf, [672](#)
  - sll, [674](#)
  - sra, [674](#)
  - srl, [674](#)

- store, [674](#)
- storeu, [674](#)
- stream, [674](#)
- sub, [675](#)
- subin, [675](#)
- type\_string, [679](#)
- unpackhi, [675](#)
- unpackhi\_twice, [674](#)
- unpacklo, [675](#)
- unpacklo\_twice, [674](#)
- unpacklohi, [675](#)
- valid, [673](#)
- vect\_size, [679](#)
- vect\_t, [672](#)
- zero, [679](#)
- Simd256\_impl< true, true, true, 2 >::Converter, [426](#)
- t, [426](#)
- v, [426](#)
- Simd256\_impl< true, true, true, 4 >, [679](#)
- add, [685](#), [691](#)
- addin, [685](#), [691](#)
- alignment, [695](#)
- blend, [685](#)
- compliant, [683](#), [689](#)
- eq, [688](#), [693](#)
- fmadd, [686](#), [692](#)
- fmaddin, [686](#), [692](#)
- fmaddx, [687](#), [692](#)
- fmaddxin, [687](#), [692](#)
- fmsub, [687](#), [693](#)
- fmsubin, [688](#), [693](#)
- fmsubx, [688](#), [693](#)
- fmsubxin, [688](#), [693](#)
- fnmadd, [687](#), [692](#)
- fnmaddin, [687](#), [692](#)
- fnmaddx, [687](#), [693](#)
- fnmaddxin, [687](#), [693](#)
- gather, [683](#), [690](#)
- greater, [688](#), [694](#)
- greater\_eq, [688](#), [694](#)
- hadd\_to\_scal, [689](#), [694](#)
- half\_t, [682](#), [683](#)
- lesser, [688](#), [694](#)
- lesser\_eq, [688](#), [694](#)
- load, [683](#), [690](#)
- loadu, [684](#), [690](#)
- mod, [689](#), [694](#)
- mul, [686](#), [691](#)
- mulhi, [686](#), [692](#)
- mullo, [686](#), [691](#)
- mulx, [686](#), [692](#)
- round, [689](#), [694](#)
- scalar\_t, [682](#), [683](#)
- set, [683](#), [689](#)
- set1, [683](#), [689](#)
- shuffle, [684](#), [691](#)
- shuffle\_twice, [684](#), [691](#)
- simdHalf, [682](#), [683](#)
- sll, [684](#), [690](#)
- sra, [684](#), [691](#)
- srl, [684](#), [690](#)
- store, [684](#), [690](#)
- storeu, [684](#), [690](#)
- stream, [684](#), [690](#)
- sub, [686](#), [691](#)
- subin, [686](#), [691](#)
- type\_string, [694](#), [695](#)
- unpackhi, [685](#)
- unpackhi\_twice, [685](#)
- unpacklo, [685](#)
- unpacklo\_twice, [685](#)
- unpacklohi, [685](#)
- valid, [683](#), [689](#)
- vand, [695](#)
- vandnot, [695](#)
- vect\_size, [695](#)
- vect\_t, [682](#)
- vor, [695](#)
- vxor, [695](#)
- zero, [694](#), [695](#)
- Simd256\_impl< true, true, true, 4 >::Converter, [426](#)
- t, [426](#)
- v, [426](#)
- Simd256\_impl< true, true, true, 8 >, [696](#)
- add, [700](#)
- addin, [700](#)
- alignment, [704](#)
- blend, [700](#)
- compliant, [698](#)
- eq, [703](#)
- fmadd, [701](#)
- fmaddin, [701](#)
- fmaddx, [701](#)
- fmaddxin, [701](#)
- fmsub, [702](#)
- fmsubin, [702](#)
- fmsubx, [702](#)
- fmsubxin, [702](#)
- fnmadd, [701](#)
- fnmaddin, [702](#)
- fnmaddx, [702](#)
- fnmaddxin, [702](#)
- gather, [698](#)
- get, [698](#)
- greater, [703](#)
- greater\_eq, [703](#)
- hadd\_to\_scal, [703](#)
- half\_t, [697](#)
- lesser, [703](#)
- lesser\_eq, [703](#)
- load, [698](#)
- loadu, [698](#)
- mask\_high, [703](#)
- mod, [704](#)
- mul, [701](#)
- mulhi\_fast, [703](#)

- mullo, [701](#)
- mulx, [701](#)
- round, [703](#)
- scalar\_t, [697](#)
- set, [698](#)
- set1, [698](#)
- shuffle, [699](#)
- signbits, [704](#)
- simdHalf, [698](#)
- sll, [699](#)
- sra, [699](#)
- srl, [699](#)
- store, [699](#)
- storeu, [699](#)
- stream, [699](#)
- sub, [700](#)
- subin, [700](#)
- type\_string, [704](#)
- unpackhi, [700](#)
- unpackhi\_twice, [699](#)
- unpacklo, [700](#)
- unpacklo\_twice, [699](#)
- unpacklohi, [700](#)
- valid, [698](#)
- vect\_size, [704](#)
- vect\_t, [697](#)
- zero, [704](#)
- Simd256\_impl< true, true, true, 8 >::Converter, [427](#)
- t, [427](#)
- v, [427](#)
- simd256\_int16.inl, [879](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int16\_INL, [879](#)
- simd256\_int32.inl, [879](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int32\_INL, [879](#)
- simd256\_int64.inl, [879](#)
- \_\_FFLASFFPACK\_fflas\_ffpack\_utils\_simd256\_int64\_INL, [880](#)
- vect\_t, [880](#)
- Simd256fp\_base, [704](#)
- Simd256i\_base, [705](#)
- type\_string, [705](#)
- vect\_t, [705](#)
- zero, [705](#)
- Simd512
- simd512.inl, [880](#)
- simd512.inl, [880](#)
- \_\_FFLASFFPACK\_simd512\_INL, [880](#)
- Simd512, [880](#)
- simd512\_double.inl, [880](#)
- \_\_FFLASFFPACK\_simd512\_double\_INL, [881](#)
- simd512\_float.inl, [881](#)
- \_\_FFLASFFPACK\_simd512\_float\_INL, [881](#)
- Simd512\_impl< ArithType, Int, Signed, Size >, [706](#)
- Simd512\_impl< true, false, true, 4 >, [706](#)
- type\_string, [706](#)
- Simd512\_impl< true, false, true, 8 >, [706](#)
- add, [710](#)
- addin, [710](#)
- alignment, [712](#)
- blend, [709](#)
- blendv, [709](#)
- ceil, [712](#)
- compliant, [708](#)
- div, [710](#)
- eq, [711](#)
- floor, [712](#)
- fmadd, [710](#)
- fmaddin, [710](#)
- fmsub, [711](#)
- fmsubin, [711](#)
- fnmadd, [711](#)
- fnmaddin, [711](#)
- gather, [708](#)
- greater, [711](#)
- greater\_eq, [712](#)
- hadd, [712](#)
- hadd\_to\_scal, [712](#)
- lesser, [711](#)
- lesser\_eq, [711](#)
- load, [708](#)
- loadu, [708](#)
- mul, [710](#)
- mulin, [710](#)
- round, [712](#)
- scalar\_t, [707](#)
- set, [708](#)
- set1, [708](#)
- shuffle, [709](#)
- store, [709](#)
- storeu, [709](#)
- stream, [709](#)
- sub, [710](#)
- subin, [710](#)
- type\_string, [712](#)
- unpackhi\_twice, [709](#)
- unpacklo\_twice, [709](#)
- valid, [708](#)
- vect\_size, [712](#)
- vect\_t, [707](#)
- zero, [708](#)
- Simd512\_impl< true, true, false, 8 >, [713](#)
- add, [720](#)
- addin, [721](#)
- alignment, [723](#)
- blend, [720](#)
- compliant, [718](#)
- eq, [722](#)
- fmadd, [721](#)
- fmaddin, [721](#)
- fmaddx, [717](#)
- fmaddxin, [717](#)
- fmsub, [722](#)
- fmsubin, [722](#)
- fmsubx, [718](#)

- fmsubxin, [718](#)
- fnmadd, [721](#)
- fnmaddin, [721](#)
- fnmaddx, [717](#)
- fnmaddxin, [717](#)
- gather, [715](#), [719](#)
- greater, [716](#)
- greater\_eq, [717](#)
- hadd\_to\_scal, [718](#)
- half\_t, [715](#)
- lesser, [716](#)
- lesser\_eq, [717](#)
- load, [716](#), [719](#)
- loadu, [716](#), [719](#)
- mask\_high, [722](#)
- maskstore, [716](#), [719](#)
- mod, [722](#)
- mul, [721](#)
- mulhi\_fast, [722](#)
- mullo, [717](#)
- mulx, [717](#)
- round, [722](#)
- scalar\_t, [715](#)
- set, [715](#), [718](#)
- set1, [715](#), [718](#)
- shuffle, [720](#)
- signbits, [723](#)
- simdHalf, [715](#)
- sll, [719](#)
- sra, [716](#)
- srl, [720](#)
- store, [716](#), [719](#)
- storeu, [716](#), [719](#)
- stream, [716](#), [719](#)
- sub, [721](#)
- subin, [721](#)
- type\_string, [723](#)
- unpackhi, [720](#)
- unpackhi\_twice, [720](#)
- unpacklo, [720](#)
- unpacklo\_twice, [720](#)
- unpacklohi, [720](#)
- valid, [718](#)
- vand, [723](#)
- vandnot, [723](#)
- vect\_size, [723](#)
- vect\_t, [715](#)
- vor, [723](#)
- vxor, [723](#)
- zero, [723](#)
- Simd512\_impl< true, true, false, 8 >::Converter, [427](#)
  - t, [427](#)
  - v, [427](#)
- Simd512\_impl< true, true, true, 8 >, [724](#)
  - add, [729](#)
  - addin, [729](#)
  - alignment, [733](#)
  - blend, [728](#)
  - compliant, [726](#)
  - eq, [731](#)
  - fmadd, [729](#)
  - fmaddin, [730](#)
  - fmaddx, [730](#)
  - fmaddxin, [730](#)
  - fmsub, [730](#)
  - fmsubin, [731](#)
  - fmsubx, [731](#)
  - fmsubxin, [731](#)
  - fnmadd, [730](#)
  - fnmaddin, [730](#)
  - fnmaddx, [730](#)
  - fnmaddxin, [730](#)
  - gather, [727](#)
  - greater, [731](#)
  - greater\_eq, [731](#)
  - hadd\_to\_scal, [732](#)
  - half\_t, [726](#)
  - lesser, [731](#)
  - lesser\_eq, [731](#)
  - load, [727](#)
  - loadu, [727](#)
  - mask\_high, [732](#)
  - maskstore, [727](#)
  - mod, [732](#)
  - mul, [729](#)
  - mulhi\_fast, [732](#)
  - mullo, [729](#)
  - mulx, [729](#)
  - round, [732](#)
  - scalar\_t, [726](#)
  - set, [726](#)
  - set1, [726](#)
  - shuffle, [728](#)
  - signbits, [732](#)
  - simdHalf, [726](#)
  - sll, [727](#)
  - sra, [728](#)
  - srl, [728](#)
  - store, [727](#)
  - storeu, [727](#)
  - stream, [727](#)
  - sub, [729](#)
  - subin, [729](#)
  - type\_string, [732](#)
  - unpackhi, [728](#)
  - unpackhi\_twice, [728](#)
  - unpacklo, [728](#)
  - unpacklo\_twice, [728](#)
  - unpacklohi, [728](#)
  - valid, [726](#)
  - vand, [733](#)
  - vandnot, [733](#)
  - vect\_size, [733](#)
  - vect\_t, [726](#)
  - vor, [732](#)
  - vxor, [733](#)

- zero, [732](#)
- Simd512\_impl< true, true, true, 8 >::Converter, [427](#)
- t, [428](#)
- v, [427](#)
- simd512\_int32.inl, [881](#)
- \_\_FFLASFFPACK\_simd512\_int32\_INL, [881](#)
- simd512\_int64.inl, [882](#)
- \_simd512\_int64\_INL, [882](#)
- vect\_t, [882](#)
- Simd512fp\_base, [733](#)
- type\_string, [734](#)
- Simd512i\_base, [734](#)
- type\_string, [734](#)
- vand, [735](#)
- vandnot, [735](#)
- vect\_t, [734](#)
- vor, [734](#)
- vxor, [735](#)
- zero, [734](#)
- SIMD\_INT
- fflas\_simd.h, [874](#)
- simd\_modular.inl, [882](#)
- SimdChooser< T, bool, bool >, [735](#)
- SimdChooser< T, false, b >, [735](#)
- value, [735](#)
- SimdChooser< T, true, false >, [735](#)
- value, [736](#)
- SimdChooser< T, true, true >, [736](#)
- value, [736](#)
- simdHalf
- Simd256\_impl< true, true, false, 2 >, [635](#)
- Simd256\_impl< true, true, false, 4 >, [646](#)
- Simd256\_impl< true, true, false, 8 >, [663](#)
- Simd256\_impl< true, true, true, 2 >, [672](#)
- Simd256\_impl< true, true, true, 4 >, [682](#), [683](#)
- Simd256\_impl< true, true, true, 8 >, [698](#)
- Simd512\_impl< true, true, false, 8 >, [715](#)
- Simd512\_impl< true, true, true, 8 >, [726](#)
- SimdSparseMatrix
- FFLAS, [79](#)
- simdToType< T >, [736](#)
- Single, [736](#)
- size
- Info, [477](#), [478](#)
- RNSInteger< RNS >, [546](#)
- RNSIntegerMod< RNS >, [551](#)
- SIZEOF\_\_INT64
- config.h, [807](#)
- SIZEOF\_CHAR
- config.h, [807](#)
- SIZEOF\_INT
- config.h, [807](#)
- SIZEOF\_LONG
- config.h, [807](#)
- SIZEOF\_LONG\_LONG
- config.h, [807](#)
- SIZEOF\_SHORT
- config.h, [807](#)
- sll
- ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [565](#)
- Simd128\_impl< true, true, false, 2 >, [574](#)
- Simd128\_impl< true, true, false, 4 >, [583](#)
- Simd128\_impl< true, true, false, 8 >, [593](#)
- Simd128\_impl< true, true, true, 2 >, [600](#)
- Simd128\_impl< true, true, true, 4 >, [609](#)
- Simd128\_impl< true, true, true, 8 >, [618](#)
- Simd256\_impl< true, true, false, 2 >, [639](#)
- Simd256\_impl< true, true, false, 4 >, [654](#)
- Simd256\_impl< true, true, false, 8 >, [667](#)
- Simd256\_impl< true, true, true, 2 >, [674](#)
- Simd256\_impl< true, true, true, 4 >, [684](#), [690](#)
- Simd256\_impl< true, true, true, 8 >, [699](#)
- Simd512\_impl< true, true, false, 8 >, [719](#)
- Simd512\_impl< true, true, true, 8 >, [727](#)
- sll128
- Simd128\_impl< true, true, false, 2 >, [576](#)
- Simd128\_impl< true, true, false, 4 >, [586](#)
- Simd128\_impl< true, true, false, 8 >, [596](#)
- Simd128\_impl< true, true, true, 2 >, [605](#)
- Simd128\_impl< true, true, true, 4 >, [614](#)
- Simd128\_impl< true, true, true, 8 >, [622](#)
- Simd128i\_base, [625](#)
- Solve
- FFPACK, [335](#), [375](#)
- solve.C, [813](#)
- main, [813](#)
- Solve\_modular\_double
- ffpack.C, [1004](#)
- ffpack\_c.h, [1025](#)
- solveLB
- FFPACK, [351](#), [375](#)
- solveLB2
- FFPACK, [351](#), [375](#)
- solveLB2\_modular\_double
- ffpack.C, [1004](#)
- ffpack\_c.h, [1026](#)
- solveLB\_modular\_double
- ffpack.C, [1004](#)
- ffpack\_c.h, [1025](#)
- Sparse< \_Field, SparseMatrix\_t::COO >, [737](#)
- col, [737](#)
- dat, [737](#)
- delayed, [737](#)
- Field, [737](#)
- kmax, [738](#)
- m, [738](#)
- maxrow, [738](#)
- n, [738](#)
- nElements, [738](#)
- nnz, [738](#)
- row, [737](#)
- Sparse< \_Field, SparseMatrix\_t::COO\_ZO >, [738](#)
- col, [739](#)
- cst, [739](#)
- dat, [739](#)

- delayed, [739](#)
  - Field, [739](#)
  - kmax, [739](#)
  - m, [739](#)
  - maxrow, [740](#)
  - n, [739](#)
  - nElements, [740](#)
  - nnz, [739](#)
  - row, [739](#)
- Sparse< \_Field, SparseMatrix\_t::CSR >, [740](#)
  - col, [741](#)
  - dat, [741](#)
  - delayed, [741](#)
  - Field, [740](#)
  - kmax, [741](#)
  - m, [741](#)
  - maxrow, [741](#)
  - n, [741](#)
  - nElements, [741](#)
  - nnz, [741](#)
  - st, [741](#)
  - stend, [741](#)
- Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, [742](#)
  - col, [742](#)
  - dat, [742](#)
  - delayed, [742](#)
  - Field, [742](#)
  - kmax, [742](#)
  - m, [743](#)
  - maxrow, [743](#)
  - n, [743](#)
  - nElements, [743](#)
  - nMOnes, [743](#)
  - nnz, [743](#)
  - nOnes, [743](#)
  - nOthers, [743](#)
  - st, [742](#)
- Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, [743](#)
  - col, [745](#)
  - cst, [744](#)
  - dat, [745](#)
  - delayed, [744](#)
  - Field, [744](#)
  - kmax, [744](#)
  - m, [744](#)
  - maxrow, [745](#)
  - n, [744](#)
  - nElements, [744](#)
  - nnz, [744](#)
  - st, [745](#)
  - stend, [745](#)
- Sparse< \_Field, SparseMatrix\_t::ELL >, [745](#)
  - col, [746](#)
  - dat, [746](#)
  - delayed, [746](#)
  - Field, [746](#)
  - kmax, [746](#)
  - ld, [746](#)
  - m, [746](#)
  - maxrow, [746](#)
  - n, [746](#)
  - nElements, [746](#)
  - nnz, [746](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd >, [747](#)
  - chunk, [747](#)
  - col, [748](#)
  - dat, [748](#)
  - delayed, [747](#)
  - kmax, [748](#)
  - ld, [747](#)
  - m, [747](#)
  - maxrow, [748](#)
  - n, [747](#)
  - nChunks, [748](#)
  - nElements, [748](#)
  - nnz, [748](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_simd\_ZO >, [748](#)
  - chunk, [749](#)
  - col, [750](#)
  - cst, [749](#)
  - dat, [750](#)
  - delayed, [749](#)
  - kmax, [749](#)
  - ld, [749](#)
  - m, [749](#)
  - maxrow, [750](#)
  - n, [749](#)
  - nChunks, [750](#)
  - nElements, [749](#)
  - nnz, [749](#)
- Sparse< \_Field, SparseMatrix\_t::ELL\_ZO >, [750](#)
  - col, [751](#)
  - cst, [751](#)
  - dat, [751](#)
  - delayed, [751](#)
  - Field, [751](#)
  - kmax, [751](#)
  - ld, [751](#)
  - m, [751](#)
  - maxrow, [751](#)
  - n, [751](#)
  - nElements, [751](#)
  - nnz, [751](#)
- Sparse< \_Field, SparseMatrix\_t::HYB\_ZO >, [752](#)
  - dat, [753](#)
  - delayed, [752](#)
  - Field, [752](#)
  - kmax, [752](#)
  - m, [752](#)
  - maxrow, [753](#)
  - mone, [753](#)
  - n, [753](#)
  - nElements, [753](#)
  - nnz, [753](#)
  - one, [753](#)
  - Self\_t, [752](#)

- Sparse< \_Field, SparseMatrix\_t::SELL >, 753
  - chunk, 754
  - chunkSize, 755
  - col, 755
  - dat, 755
  - delayed, 754
  - Field, 754
  - kmax, 754
  - m, 754
  - maxrow, 754
  - n, 754
  - nChunks, 755
  - nElements, 755
  - nnz, 755
  - perm, 755
  - sigma, 754
  - st, 755
- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 755
  - chunk, 756
  - chunkSize, 757
  - col, 757
  - cst, 756
  - dat, 757
  - delayed, 756
  - Field, 756
  - kmax, 756
  - m, 756
  - maxrow, 757
  - n, 756
  - nChunks, 757
  - nElements, 757
  - nnz, 757
  - perm, 757
  - sigma, 757
  - st, 757
- Sparse< Field, SparseMatrix\_t, IdxT, PtrT >, 736
- sparse\_delete
  - FFLAS, 151, 152, 154–156, 158
- sparse\_init
  - FFLAS, 151–156, 159
- sparse\_matrix\_traits.h, 914
- sparse\_print
  - FFLAS, 152, 156, 159
- SparseMatrix\_t
  - FFLAS, 83
- SpecRankProfile
  - FFPACK, 374
- SpecRankProfile\_modular\_double
  - ffpack.C, 1003
  - ffpack\_c.h, 1024
- splitt
  - parallel.h, 1045
- SPLITTER
  - parallel.h, 1045
- splitting\_0
  - parallel.h, 1045
- splitting\_1
  - parallel.h, 1045
- splitting\_2
  - parallel.h, 1045
- splitting\_3
  - parallel.h, 1045
- SpMat< Field, flag >, 757
  - \_coo, 758
  - \_csr, 758
  - \_ell, 758
- sra
  - ScalFunctions< Element, typename enable\_if<
    - is\_integral< Element >::value >::type >, 564, 565
 Simd128\_impl< true, true, false, 2 >, 571
 Simd128\_impl< true, true, false, 4 >, 580
 Simd128\_impl< true, true, false, 8 >, 590
 Simd128\_impl< true, true, true, 2 >, 601
 Simd128\_impl< true, true, true, 4 >, 609
 Simd128\_impl< true, true, true, 8 >, 618
 Simd256\_impl< true, true, false, 2 >, 636
 Simd256\_impl< true, true, false, 4 >, 648, 650
 Simd256\_impl< true, true, false, 8 >, 664
 Simd256\_impl< true, true, true, 2 >, 674
 Simd256\_impl< true, true, true, 4 >, 684, 691
 Simd256\_impl< true, true, true, 8 >, 699
 Simd512\_impl< true, true, false, 8 >, 716
 Simd512\_impl< true, true, true, 8 >, 728
 >, 564, 565
 Simd128\_impl< true, true, false, 2 >, 571
 Simd128\_impl< true, true, false, 4 >, 580
 Simd128\_impl< true, true, false, 8 >, 590
 Simd128\_impl< true, true, true, 2 >, 601
 Simd128\_impl< true, true, true, 4 >, 609
 Simd128\_impl< true, true, true, 8 >, 618
 Simd256\_impl< true, true, false, 2 >, 636
 Simd256\_impl< true, true, false, 4 >, 648, 650
 Simd256\_impl< true, true, false, 8 >, 664
 Simd256\_impl< true, true, true, 2 >, 674
 Simd256\_impl< true, true, true, 4 >, 684, 691
 Simd256\_impl< true, true, true, 8 >, 699
 Simd512\_impl< true, true, false, 8 >, 716
 Simd512\_impl< true, true, true, 8 >, 728
 >, 564, 565
- srl
  - ScalFunctions< Element, typename enable\_if<
    - is\_integral< Element >::value >::type >, 565
 Simd128\_impl< true, true, false, 2 >, 574
 Simd128\_impl< true, true, false, 4 >, 583
 Simd128\_impl< true, true, false, 8 >, 593
 Simd128\_impl< true, true, true, 2 >, 600
 Simd128\_impl< true, true, true, 4 >, 609
 Simd128\_impl< true, true, true, 8 >, 618
 Simd256\_impl< true, true, false, 2 >, 640
 Simd256\_impl< true, true, false, 4 >, 654
 Simd256\_impl< true, true, false, 8 >, 667
 Simd256\_impl< true, true, true, 2 >, 674
 Simd256\_impl< true, true, true, 4 >, 684, 690
 Simd256\_impl< true, true, true, 8 >, 699
 Simd512\_impl< true, true, false, 8 >, 720
 Simd512\_impl< true, true, true, 8 >, 728
 >, 565
 Simd128\_impl< true, true, false, 2 >, 574
 Simd128\_impl< true, true, false, 4 >, 583
 Simd128\_impl< true, true, false, 8 >, 593
 Simd128\_impl< true, true, true, 2 >, 600
 Simd128\_impl< true, true, true, 4 >, 609
 Simd128\_impl< true, true, true, 8 >, 618
 Simd256\_impl< true, true, false, 2 >, 640
 Simd256\_impl< true, true, false, 4 >, 654
 Simd256\_impl< true, true, false, 8 >, 667
 Simd256\_impl< true, true, true, 2 >, 674
 Simd256\_impl< true, true, true, 4 >, 684, 690
 Simd256\_impl< true, true, true, 8 >, 699
 Simd512\_impl< true, true, false, 8 >, 720
 Simd512\_impl< true, true, true, 8 >, 728
 >, 565
- srl128
  - Simd128\_impl< true, true, false, 2 >, 576
  - Simd128\_impl< true, true, false, 4 >, 586
  - Simd128\_impl< true, true, false, 8 >, 596
  - Simd128\_impl< true, true, true, 2 >, 605
  - Simd128\_impl< true, true, true, 4 >, 614
  - Simd128\_impl< true, true, true, 8 >, 623
  - Simd128i\_base, 625
- sscal\_
  - config-blas.h, 824
- st
  - Sparse< \_Field, SparseMatrix\_t::CSR >, 741
  - Sparse< \_Field, SparseMatrix\_t::CSR\_HYB >, 742
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 745

- Sparse< \_Field, SparseMatrix\_t::SELL >, 755
- Sparse< \_Field, SparseMatrix\_t::SELL\_ZO >, 757
- Static\_error\_check
  - Static\_error\_check< bool >, 758
- Static\_error\_check< bool >, 758
  - Static\_error\_check, 758
- Static\_error\_check< false >, 758
- StatsMatrix, 758
  - averageCol, 760
  - averageColDifference, 760
  - averageRow, 760
  - averageRowDifference, 761
  - coldim, 759
  - denseCols, 761
  - denseRows, 761
  - deviationCol, 760
  - deviationColDifference, 761
  - deviationRow, 760
  - deviationRowDifference, 761
  - maxCol, 760
  - maxColDifference, 760
  - maxRow, 760
  - maxRowDifference, 761
  - minCol, 760
  - minColDifference, 760
  - minRow, 760
  - minRowDifference, 761
  - nDenseCols, 761
  - nDenseRows, 761
  - nEmptyCols, 761
  - nEmptyColsEnd, 761
  - nEmptyRows, 761
  - nMOnes, 759
  - nnz, 760
  - nOnes, 759
  - nOthers, 759
  - rowdim, 759
- STD\_RECINT\_SIZE
  - benchmark-fgemm-mp.C, 787
  - benchmark-fgemv-mp.C, 790
- STDC\_HEADERS
  - config.h, 807
- stend
  - Sparse< \_Field, SparseMatrix\_t::CSR >, 741
  - Sparse< \_Field, SparseMatrix\_t::CSR\_ZO >, 745
- store
  - Simd128\_impl< true, true, false, 2 >, 570, 573
  - Simd128\_impl< true, true, false, 4 >, 580, 583
  - Simd128\_impl< true, true, false, 8 >, 590, 593
  - Simd128\_impl< true, true, true, 2 >, 600
  - Simd128\_impl< true, true, true, 4 >, 609
  - Simd128\_impl< true, true, true, 8 >, 617
  - Simd256\_impl< true, false, true, 8 >, 628
  - Simd256\_impl< true, true, false, 2 >, 636, 639
  - Simd256\_impl< true, true, false, 4 >, 647, 650, 654
  - Simd256\_impl< true, true, false, 8 >, 663, 666
  - Simd256\_impl< true, true, true, 2 >, 674
- Simd256\_impl< true, true, true, 4 >, 684, 690
- Simd256\_impl< true, true, true, 8 >, 699
- Simd512\_impl< true, false, true, 8 >, 709
- Simd512\_impl< true, true, false, 8 >, 716, 719
- Simd512\_impl< true, true, true, 8 >, 727
- storeu
  - Simd128\_impl< true, true, false, 2 >, 570, 573
  - Simd128\_impl< true, true, false, 4 >, 580, 583
  - Simd128\_impl< true, true, false, 8 >, 590, 593
  - Simd128\_impl< true, true, true, 2 >, 600
  - Simd128\_impl< true, true, true, 4 >, 609
  - Simd128\_impl< true, true, true, 8 >, 617
  - Simd256\_impl< true, false, true, 8 >, 628
  - Simd256\_impl< true, true, false, 2 >, 636, 639
  - Simd256\_impl< true, true, false, 4 >, 647, 650, 654
  - Simd256\_impl< true, true, false, 8 >, 664, 667
  - Simd256\_impl< true, true, true, 2 >, 674
  - Simd256\_impl< true, true, true, 4 >, 684, 690
  - Simd256\_impl< true, true, true, 8 >, 699
  - Simd512\_impl< true, false, true, 8 >, 709
  - Simd512\_impl< true, true, false, 8 >, 716, 719
  - Simd512\_impl< true, true, true, 8 >, 727
- stream
  - Simd128\_impl< true, true, false, 2 >, 570, 573
  - Simd128\_impl< true, true, false, 4 >, 580, 583
  - Simd128\_impl< true, true, false, 8 >, 590, 593
  - Simd128\_impl< true, true, true, 2 >, 600
  - Simd128\_impl< true, true, true, 4 >, 609
  - Simd128\_impl< true, true, true, 8 >, 618
  - Simd256\_impl< true, false, true, 8 >, 629
  - Simd256\_impl< true, true, false, 2 >, 636, 639
  - Simd256\_impl< true, true, false, 4 >, 648, 650, 654
  - Simd256\_impl< true, true, false, 8 >, 664, 667
  - Simd256\_impl< true, true, true, 2 >, 674
  - Simd256\_impl< true, true, true, 4 >, 684, 690
  - Simd256\_impl< true, true, true, 8 >, 699
  - Simd512\_impl< true, false, true, 8 >, 709
  - Simd512\_impl< true, true, false, 8 >, 716, 719
  - Simd512\_impl< true, true, true, 8 >, 727
- strmm\_
  - config-blas.h, 825
- strsm\_
  - config-blas.h, 824
- sub
  - FFLAS::vectorised, 289
  - FieldSimd< \_Field >, 447
  - RNSIntegerMod< RNS >, 552
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, 558
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, 562
  - Simd128\_impl< true, true, false, 2 >, 574
  - Simd128\_impl< true, true, false, 4 >, 584
  - Simd128\_impl< true, true, false, 8 >, 594
  - Simd128\_impl< true, true, true, 2 >, 601



- Simd128\_impl< true, true, true, 4 >, [610](#)
- Simd128\_impl< true, true, true, 8 >, [619](#)
- Simd256\_impl< true, false, true, 8 >, [629](#)
- Simd256\_impl< true, true, false, 2 >, [641](#)
- Simd256\_impl< true, true, false, 4 >, [656](#)
- Simd256\_impl< true, true, false, 8 >, [668](#)
- Simd256\_impl< true, true, true, 2 >, [675](#)
- Simd256\_impl< true, true, true, 4 >, [686](#), [691](#)
- Simd256\_impl< true, true, true, 8 >, [700](#)
- Simd512\_impl< true, false, true, 8 >, [710](#)
- Simd512\_impl< true, true, false, 8 >, [721](#)
- Simd512\_impl< true, true, true, 8 >, [729](#)
- sub\_r
  - FieldSimd< \_Field >, [447](#)
- subin
  - FieldSimd< \_Field >, [447](#)
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [558](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [562](#)
  - Simd128\_impl< true, true, false, 2 >, [575](#)
  - Simd128\_impl< true, true, false, 4 >, [584](#)
  - Simd128\_impl< true, true, false, 8 >, [594](#)
  - Simd128\_impl< true, true, true, 2 >, [601](#)
  - Simd128\_impl< true, true, true, 4 >, [610](#)
  - Simd128\_impl< true, true, true, 8 >, [619](#)
  - Simd256\_impl< true, false, true, 8 >, [630](#)
  - Simd256\_impl< true, true, false, 2 >, [641](#)
  - Simd256\_impl< true, true, false, 4 >, [656](#)
  - Simd256\_impl< true, true, false, 8 >, [668](#)
  - Simd256\_impl< true, true, true, 2 >, [675](#)
  - Simd256\_impl< true, true, true, 4 >, [686](#), [691](#)
  - Simd256\_impl< true, true, true, 8 >, [700](#)
  - Simd512\_impl< true, false, true, 8 >, [710](#)
  - Simd512\_impl< true, true, false, 8 >, [721](#)
  - Simd512\_impl< true, true, true, 8 >, [729](#)
- subin\_r
  - FieldSimd< \_Field >, [447](#)
- subp
  - FFLAS::vectorised, [288](#)
- support\_fast\_mod< double >, [762](#)
- support\_fast\_mod< float >, [762](#)
- support\_fast\_mod< int64\_t >, [762](#)
- support\_fast\_mod< T >, [762](#)
- support\_simd< T >, [763](#)
- support\_simd\_add< T >, [763](#)
- support\_simd\_mod< T >, [763](#)
- swapval
  - FFPACK, [388](#)
- SYNCH\_GROUP
  - parallel.h, [1040](#)
- SysTimer
  - FFLAS, [81](#)
- T
  - limits< char >, [486](#)
  - limits< double >, [487](#)
  - limits< float >, [488](#)
  - limits< Givaro::Integer >, [488](#)
  - limits< int >, [489](#)
  - limits< long >, [490](#)
  - limits< long long >, [490](#)
  - limits< Reclnt::rint< K > >, [491](#)
  - limits< Reclnt::ruint< K > >, [492](#)
  - limits< short int >, [492](#)
  - limits< signed char >, [493](#)
  - limits< unsigned char >, [494](#)
  - limits< unsigned int >, [495](#)
  - limits< unsigned long >, [495](#)
  - limits< unsigned long long >, [496](#)
  - limits< unsigned short int >, [497](#)
- t
  - Simd128\_impl< true, true, false, 2 >::Converter, [422](#)
  - Simd128\_impl< true, true, false, 4 >::Converter, [423](#)
  - Simd128\_impl< true, true, false, 8 >::Converter, [423](#)
  - Simd128\_impl< true, true, true, 2 >::Converter, [424](#)
  - Simd128\_impl< true, true, true, 4 >::Converter, [424](#)
  - Simd128\_impl< true, true, true, 8 >::Converter, [424](#)
  - Simd256\_impl< true, true, false, 2 >::Converter, [425](#)
  - Simd256\_impl< true, true, false, 4 >::Converter, [425](#)
  - Simd256\_impl< true, true, false, 8 >::Converter, [426](#)
  - Simd256\_impl< true, true, true, 2 >::Converter, [426](#)
  - Simd256\_impl< true, true, true, 4 >::Converter, [426](#)
  - Simd256\_impl< true, true, true, 8 >::Converter, [427](#)
  - Simd512\_impl< true, true, false, 8 >::Converter, [427](#)
  - Simd512\_impl< true, true, true, 8 >::Converter, [428](#)
- TASK
  - parallel.h, [1040](#)
- tBC
  - test-lu.C, [1105](#)
  - test-permutations.C, [1110](#)
- test
  - test-maxdelayeddim.C, [1106](#)
  - test-simd.C, [1116](#), [1117](#)
- test-charpoly-check.C, [1066](#)
  - ENABLE\_CHECKER\_charpoly, [1066](#)
  - main, [1066](#)
  - printPolynomial, [1066](#)
  - TIME\_CHECKER\_CHARPOLY, [1066](#)
- test-charpoly.C, [1066](#)
  - launch\_test, [1067](#)
  - main, [1067](#)

- run\_with\_field, 1067
- test-compressQ.C, 1067
  - Field, 1068
  - main, 1068
  - printvect, 1068
- test-det-check.C, 1068
  - ENABLE\_CHECKER\_Det, 1069
  - main, 1069
  - TIME\_CHECKER\_Det, 1069
- test-det.C, 1069
  - main, 1070
  - test\_det, 1069
- test-echelon.C, 1070
  - \_\_FFLASFFPACK\_GAUSSJORDAN\_BASECASE, 1071
  - \_\_FFLASFFPACK\_PLUQ\_THRESHOLD, 1071
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1071
  - main, 1072
  - run\_with\_field, 1072
  - test\_colechelon, 1071
  - test\_redcolechelon, 1071
  - test\_redrowechelon, 1072
  - test\_rowechelon, 1071
- test-fadd.C, 1072
  - main, 1073
  - test\_fadd, 1073
  - test\_faddin, 1073
  - test\_fsub, 1073
  - test\_fsubin, 1073
- test-fdot.C, 1074
  - check\_fdot, 1074
  - ENABLE\_ALL\_CHECKINGS, 1074
  - main, 1075
  - run\_with\_field, 1074
  - run\_with\_Integer, 1075
- test-fgemm-check.C, 1075
  - ENABLE\_ALL\_CHECKINGS, 1075
  - launch\_MM\_dispatch, 1075
  - main, 1076
  - run\_with\_field, 1076
- test-fgemm.C, 1076
  - check\_MM, 1077
  - ENABLE\_CHECKER\_fgemm, 1077
  - launch\_MM, 1077
  - launch\_MM\_dispatch, 1078
  - main, 1078
  - run\_with\_field, 1078
- test-fgemv.C, 1079
  - check\_MV, 1079
  - launch\_MV, 1079
  - launch\_MV\_dispatch, 1080
  - main, 1080
  - run\_with\_field, 1080
- test-fger.C, 1080
  - check\_fger, 1081
  - launch\_fger, 1081
  - launch\_fger\_dispatch, 1082
  - main, 1082
- run\_with\_field, 1082
- TIME, 1081
- test-fgesv.C, 1082
  - main, 1083
  - run\_with\_field, 1083
  - test\_rect\_fgesv, 1083
  - test\_square\_fgesv, 1083
- test-finit.C, 1084
  - main, 1084
  - run\_with\_field, 1084
  - test\_freduce, 1084
- test-fscal.C, 1085
  - main, 1086
  - test\_fscal, 1085
  - test\_fscaln, 1085, 1086
- test-fsyr2k.C, 1086
  - check\_fsyr2k, 1087
  - ENABLE\_ALL\_CHECKINGS, 1086
  - main, 1087
  - run\_with\_field, 1087
- test-fsyrk.C, 1087
  - check\_fsyrk, 1088
  - check\_fsyrk\_bkdiag, 1088
  - check\_fsyrk\_diag, 1088
  - ENABLE\_ALL\_CHECKINGS, 1088
  - main, 1089
  - run\_with\_field, 1089
- test-fsytrf.C, 1089
  - main, 1090
  - operator<<, 1089
  - run\_with\_field, 1090
  - test\_generic\_fsytrf, 1090
  - test\_RPM\_fsytrf, 1090
- test-ftmm.C, 1090
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1091
  - check\_ftmm, 1091
  - main, 1091
  - run\_with\_field, 1091
- test-ftmv.C, 1092
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1092
  - check\_ftmv, 1092
  - ENABLE\_ALL\_CHECKINGS, 1092
  - main, 1093
  - run\_with\_field, 1093
- test-ftsm-check.C, 1093
  - ENABLE\_ALL\_CHECKINGS, 1093
  - main, 1093
- test-ftsm.C, 1094
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1094
  - check\_ftsm, 1094
  - ENABLE\_ALL\_CHECKINGS, 1094
  - main, 1095
  - run\_with\_field, 1095
- test-ftssyr2k.C, 1095
  - check\_ftssyr2k, 1096
  - ENABLE\_ALL\_CHECKINGS, 1095
  - main, 1096
  - run\_with\_field, 1096

- test-ftrstr.C, 1096
  - check\_ftrstr, 1097
  - ENABLE\_ALL\_CHECKINGS, 1097
  - main, 1097
  - run\_with\_field, 1097
- test-ftrsv.C, 1097
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1098
  - check\_ftrsv, 1098
  - ENABLE\_ALL\_CHECKINGS, 1098
  - main, 1098
  - run\_with\_field, 1098
- test-ftrtri.C, 1098
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1099
  - check\_ftrtri, 1099
  - ENABLE\_ALL\_CHECKINGS, 1099
  - main, 1099
  - run\_with\_field, 1099
- test-interfaces-c.c, 1100
  - main, 1100
- test-invert-check.C, 1100
  - ENABLE\_ALL\_CHECKINGS, 1100
  - main, 1100
- test-io.C, 1101
  - main, 1101
  - run\_with\_field, 1101
- test-lu.C, 1102
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1102
  - \_\_LUDIVINE\_CUTOFF, 1103
  - BASECASE\_K, 1102
  - launch\_test, 1104
  - main, 1105
  - mvcnt, 1106
  - run\_with\_field, 1105
  - tBC, 1105
  - test\_LUdivine, 1103
  - test\_pluq, 1104
  - tgemm, 1105
  - timtot, 1105
  - tperm, 1105
  - trest, 1105
  - ttrsm, 1105
  - verifPLUQ, 1103
- test-maxdelayeddim.C, 1106
  - main, 1106
  - MAX\_WITH\_SIZE\_T, 1106
  - test, 1106
- test-minpoly.C, 1107
  - check\_minpoly, 1107
  - main, 1107
  - run\_with\_field, 1107
- test-multifile1.C, 1108
- test-multifile2.C, 1108
  - main, 1108
- test-nullspace.C, 1108
  - checkingMessage, 1108
  - main, 1109
  - readOrRandomMatrixWithRankAndRandomRPM, 1109
  - run\_with\_field, 1109
  - test\_nullspace, 1109
- test-permutations.C, 1109
  - checkMonotonicApplyP, 1110
  - main, 1110
  - tBC, 1110
  - tgemm, 1110
  - timtot, 1111
  - tperm, 1110
  - trest, 1110
  - ttrsm, 1110
- test-pluq-check.C, 1111
  - ENABLE\_ALL\_CHECKINGS, 1111
  - main, 1111
- test-rankprofiles.C, 1111
  - \_\_FFLASFFPACK\_SEQUENTIAL, 1112
  - main, 1112
  - run\_with\_field, 1112
- test-rpm.C, 1112
  - checkRPM, 1113
  - checkSymmetricRPM, 1113
  - main, 1113
- test-simd.C, 1113
  - check\_eq, 1116
  - eval\_func\_on\_array, 1116
  - generate\_random\_vector, 1115, 1116
  - integer, 1114
  - main, 1117
  - REGISTER\_TYPE\_NAME, 1114, 1115
  - test, 1116, 1117
  - test\_impl, 1116
  - TEST\_ONE\_OP, 1114
  - test\_op, 1116
  - TypeName, 1115
- test-solve.C, 1117
  - check\_solve, 1117
  - main, 1117
  - run\_with\_field, 1117
- test-utils.h, 1056
- test\_colechelon
  - test-echelon.C, 1071
- test\_det
  - test-det.C, 1069
- test\_fadd
  - test-fadd.C, 1073
- test\_faddin
  - test-fadd.C, 1073
- test\_freduce
  - test-finit.C, 1084
- test\_fscal
  - test-fscal.C, 1085
- test\_fscaln
  - test-fscal.C, 1085, 1086
- test\_fsub
  - test-fadd.C, 1073
- test\_fsubin
  - test-fadd.C, 1073
- test\_generic\_fsytrf

- test-fsytrf.C, [1090](#)
- test\_impl
  - test-simd.C, [1116](#)
- test\_LUdivine
  - test-lu.C, [1103](#)
- test\_nullspace
  - test-nullspace.C, [1109](#)
- TEST\_ONE\_OP
  - test-simd.C, [1114](#)
- test\_op
  - test-simd.C, [1116](#)
- test\_pluq
  - test-lu.C, [1104](#)
- test\_rect\_fgesv
  - test-fgesv.C, [1083](#)
- test\_redcolechelon
  - test-echelon.C, [1071](#)
- test\_redrowechelon
  - test-echelon.C, [1072](#)
- test\_rowechelon
  - test-echelon.C, [1071](#)
- test\_RPM\_fsytrf
  - test-fsytrf.C, [1090](#)
- test\_square\_fgesv
  - test-fgesv.C, [1083](#)
- tfn\_minus, [764](#)
  - operator(), [764](#)
- tfn\_minus\_eq, [764](#)
  - operator(), [764](#)
- tfn\_mul, [764](#)
  - operator(), [764](#)
- tfn\_mul\_eq, [765](#)
  - operator(), [765](#)
- tfn\_plus, [765](#)
  - operator(), [765](#)
- tfn\_plus\_eq, [765](#)
  - operator(), [766](#)
- tgemm
  - test-lu.C, [1105](#)
  - test-permutations.C, [1110](#)
- THREADS
  - benchmark-fgemm-rns.C, [788](#)
- Threads, [766](#)
- threads\_fgemm
  - FFPACK, [365](#)
- threads\_ftsm
  - FFPACK, [365](#)
- THREED
  - benchmark-fgemm-rns.C, [789](#)
- ThreeD, [766](#)
- THREEDA
  - benchmark-fgemm-rns.C, [789](#)
- ThreeDAdaptive, [766](#)
- ThreeDInPlace, [766](#)
- THREEDIIP
  - benchmark-fgemm-rns.C, [789](#)
- TIME
  - test-fger.C, [1081](#)
- TIME\_CHECKER\_CHARPOLY
  - test-charpoly-check.C, [1066](#)
- TIME\_CHECKER\_Det
  - test-det-check.C, [1069](#)
- Timer
  - FFLAS, [80](#)
- timer.h, [1057](#)
- timtot
  - test-lu.C, [1105](#)
  - test-permutations.C, [1111](#)
- tmain
  - benchmark-fgemm-mp.C, [787](#)
  - benchmark-fgemv-mp.C, [791](#)
- tperm
  - test-lu.C, [1105](#)
  - test-permutations.C, [1110](#)
- trest
  - test-lu.C, [1105](#)
  - test-permutations.C, [1110](#)
- trinv\_left
  - FFPACK, [316](#), [368](#)
- trinv\_left\_modular\_double
  - ffpack.C, [995](#)
  - ffpack\_c.h, [1019](#)
- TRSMBound
  - FFLAS::Protected, [220](#)
- TRSMHelper
  - TRSMHelper< ReclterTrait, ParSeqTrait >, [767](#)
- TRSMHelper< ReclterTrait, ParSeqTrait >, [766](#)
  - parseq, [767](#)
  - pMMH, [767](#)
  - TRSMHelper, [767](#)
- TTimer
  - arithprog.C, [770](#)
  - autotune/charpoly.C, [771](#)
  - autotune/pluq.C, [775](#)
  - benchmark-dgemm.C, [780](#)
  - benchmark-dgetrf.C, [781](#)
  - benchmark-dgetri.C, [782](#)
  - benchmark-dsytrf.C, [783](#)
  - benchmark-dtrsm.C, [783](#)
  - benchmark-dtrtri.C, [784](#)
  - fsyrk.C, [772](#)
  - fsytrf.C, [773](#)
  - fttri.C, [774](#)
  - winograd.C, [776](#)
- ttrsm
  - test-lu.C, [1105](#)
  - test-permutations.C, [1110](#)
- TWOD
  - benchmark-fgemm-rns.C, [788](#)
- TwoD, [768](#)
- TWODA
  - benchmark-fgemm-rns.C, [788](#)
- TwoDAdaptive, [768](#)
- type
  - Argument, [407](#)

- associatedDelayedField< const FFPACK::RNSIntegerModSimd512i\_base, 734  
RNS > >, 408
- associatedDelayedField< const Givaro::Modular<  
T, X > >, 408
- associatedDelayedField< const Givaro::ModularBalanced< test-simd.C, 1115  
T > >, 409
- associatedDelayedField< const Givaro::ZRing< T  
> >, 409
- associatedDelayedField< Field >, 407
- CompactElement< double >, 419
- CompactElement< Element >, 418
- CompactElement< float >, 419
- CompactElement< int16\_t >, 419
- CompactElement< int32\_t >, 419
- CompactElement< int64\_t >, 420
- is\_simd< T >, 479
- TYPE\_BOOL  
args-parser.h, 1048
- TYPE\_DOUBLE  
args-parser.h, 1048
- TYPE\_INT  
args-parser.h, 1048
- TYPE\_INTEGER  
args-parser.h, 1048
- type\_integer  
args-parser.h, 1048
- TYPE\_INTLIST  
args-parser.h, 1048
- TYPE\_LONGLONG  
args-parser.h, 1048
- TYPE\_NONE  
args-parser.h, 1048
- TYPE\_STR  
args-parser.h, 1048
- type\_string  
NoSimd< T >, 521
- Simd128\_impl< true, false, true, 4 >, 567
- Simd128\_impl< true, false, true, 8 >, 567
- Simd128\_impl< true, true, false, 2 >, 576
- Simd128\_impl< true, true, false, 4 >, 586
- Simd128\_impl< true, true, false, 8 >, 596
- Simd128\_impl< true, true, true, 2 >, 605
- Simd128\_impl< true, true, true, 4 >, 613
- Simd128\_impl< true, true, true, 8 >, 622
- Simd128fp\_base, 624
- Simd128i\_base, 625
- Simd256\_impl< true, true, false, 2 >, 642
- Simd256\_impl< true, true, false, 4 >, 659
- Simd256\_impl< true, true, false, 8 >, 670
- Simd256\_impl< true, true, true, 2 >, 679
- Simd256\_impl< true, true, true, 4 >, 694, 695
- Simd256\_impl< true, true, true, 8 >, 704
- Simd256i\_base, 705
- Simd512\_impl< true, false, true, 4 >, 706
- Simd512\_impl< true, false, true, 8 >, 712
- Simd512\_impl< true, true, false, 8 >, 723
- Simd512\_impl< true, true, true, 8 >, 732
- Simd512fp\_base, 734
- TYPE\_UINT64  
args-parser.h, 1048
- TypeName
- uint16\_t  
instrset.h, 1062
- uint32\_t  
instrset.h, 1062
- uint64\_t  
instrset.h, 1062
- uint8\_t  
instrset.h, 1062
- unfit  
FFLAS::Protected, 227
- unpackhi  
Simd128\_impl< true, true, false, 2 >, 574
- Simd128\_impl< true, true, false, 4 >, 584
- Simd128\_impl< true, true, false, 8 >, 593
- Simd128\_impl< true, true, true, 2 >, 601
- Simd128\_impl< true, true, true, 4 >, 610
- Simd128\_impl< true, true, true, 8 >, 618
- Simd256\_impl< true, true, false, 2 >, 640
- Simd256\_impl< true, true, false, 4 >, 655
- Simd256\_impl< true, true, false, 8 >, 667
- Simd256\_impl< true, true, true, 2 >, 675
- Simd256\_impl< true, true, true, 4 >, 685
- Simd256\_impl< true, true, true, 8 >, 700
- Simd512\_impl< true, true, false, 8 >, 720
- Simd512\_impl< true, true, true, 8 >, 728
- unpackhi\_twice  
Simd256\_impl< true, false, true, 8 >, 629
- Simd256\_impl< true, true, false, 2 >, 640
- Simd256\_impl< true, true, false, 4 >, 655
- Simd256\_impl< true, true, false, 8 >, 667
- Simd256\_impl< true, true, true, 2 >, 674
- Simd256\_impl< true, true, true, 4 >, 685
- Simd256\_impl< true, true, true, 8 >, 699
- Simd512\_impl< true, false, true, 8 >, 709
- Simd512\_impl< true, true, false, 8 >, 720
- Simd512\_impl< true, true, true, 8 >, 728
- unpacklo  
Simd128\_impl< true, true, false, 2 >, 574
- Simd128\_impl< true, true, false, 4 >, 584
- Simd128\_impl< true, true, false, 8 >, 593
- Simd128\_impl< true, true, true, 2 >, 601
- Simd128\_impl< true, true, true, 4 >, 609
- Simd128\_impl< true, true, true, 8 >, 618
- Simd256\_impl< true, true, false, 2 >, 640
- Simd256\_impl< true, true, false, 4 >, 655
- Simd256\_impl< true, true, false, 8 >, 667
- Simd256\_impl< true, true, true, 2 >, 675
- Simd256\_impl< true, true, true, 4 >, 685
- Simd256\_impl< true, true, true, 8 >, 700
- Simd512\_impl< true, true, false, 8 >, 720
- Simd512\_impl< true, true, true, 8 >, 728
- unpacklo\_twice  
Simd256\_impl< true, false, true, 8 >, 629

- Simd256\_impl< true, true, false, 2 >, [640](#)
- Simd256\_impl< true, true, false, 4 >, [655](#)
- Simd256\_impl< true, true, false, 8 >, [667](#)
- Simd256\_impl< true, true, true, 2 >, [674](#)
- Simd256\_impl< true, true, true, 4 >, [685](#)
- Simd256\_impl< true, true, true, 8 >, [699](#)
- Simd512\_impl< true, false, true, 8 >, [709](#)
- Simd512\_impl< true, true, false, 8 >, [720](#)
- Simd512\_impl< true, true, true, 8 >, [728](#)
- unpacklohi
  - Simd256\_impl< true, true, false, 2 >, [640](#)
  - Simd256\_impl< true, true, false, 4 >, [655](#)
  - Simd256\_impl< true, true, false, 8 >, [668](#)
  - Simd256\_impl< true, true, true, 2 >, [675](#)
  - Simd256\_impl< true, true, true, 4 >, [685](#)
  - Simd256\_impl< true, true, true, 8 >, [700](#)
  - Simd512\_impl< true, true, false, 8 >, [720](#)
  - Simd512\_impl< true, true, true, 8 >, [728](#)
- UnparametricTag, [768](#)
- updateD
  - FFPACK::Protected, [400](#)
- USE\_OPENMP
  - config.h, [807](#)
- UserTimer
  - FFLAS, [81](#)
- utils.h, [915](#)
- v
  - Simd128\_impl< true, true, false, 2 >::Converter, [422](#)
  - Simd128\_impl< true, true, false, 4 >::Converter, [423](#)
  - Simd128\_impl< true, true, false, 8 >::Converter, [423](#)
  - Simd128\_impl< true, true, true, 2 >::Converter, [424](#)
  - Simd128\_impl< true, true, true, 4 >::Converter, [424](#)
  - Simd128\_impl< true, true, true, 8 >::Converter, [424](#)
  - Simd256\_impl< true, true, false, 2 >::Converter, [425](#)
  - Simd256\_impl< true, true, false, 4 >::Converter, [425](#)
  - Simd256\_impl< true, true, false, 8 >::Converter, [425](#)
  - Simd256\_impl< true, true, true, 2 >::Converter, [426](#)
  - Simd256\_impl< true, true, true, 4 >::Converter, [426](#)
  - Simd256\_impl< true, true, true, 8 >::Converter, [427](#)
  - Simd512\_impl< true, true, false, 8 >::Converter, [427](#)
  - Simd512\_impl< true, true, true, 8 >::Converter, [427](#)
- val
  - Coo< Field >, [431](#)
  - Coo< ValT, IdxT >, [429](#), [432](#)
- valid
  - NoSimd< T >, [521](#)
  - Simd128\_impl< true, true, false, 2 >, [572](#)
  - Simd128\_impl< true, true, false, 4 >, [582](#)
  - Simd128\_impl< true, true, false, 8 >, [592](#)
  - Simd128\_impl< true, true, true, 2 >, [599](#)
  - Simd128\_impl< true, true, true, 4 >, [608](#)
  - Simd128\_impl< true, true, true, 8 >, [617](#)
  - Simd256\_impl< true, false, true, 8 >, [627](#)
  - Simd256\_impl< true, true, false, 2 >, [638](#)
  - Simd256\_impl< true, true, false, 4 >, [652](#)
  - Simd256\_impl< true, true, false, 8 >, [666](#)
  - Simd256\_impl< true, true, true, 2 >, [673](#)
  - Simd256\_impl< true, true, true, 4 >, [683](#), [689](#)
  - Simd256\_impl< true, true, true, 8 >, [698](#)
  - Simd512\_impl< true, false, true, 8 >, [708](#)
  - Simd512\_impl< true, true, false, 8 >, [718](#)
  - Simd512\_impl< true, true, true, 8 >, [726](#)
- VALUE
  - parallel.h, [1041](#)
- value
  - AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >, [405](#)
  - AlgoChooser< ModeT, ParSeq >, [405](#)
  - AreEqual< X, X >, [406](#)
  - AreEqual< X, Y >, [406](#)
  - compatible\_data\_type< Field >, [420](#)
  - compatible\_data\_type< Givaro::ZRing< double > >, [420](#)
  - compatible\_data\_type< Givaro::ZRing< float > >, [421](#)
  - ElementTraits< double >, [436](#)
  - ElementTraits< Element >, [435](#)
  - ElementTraits< FFPACK::rns\_double\_elt >, [436](#)
  - ElementTraits< float >, [436](#)
  - ElementTraits< Givaro::Integer >, [437](#)
  - ElementTraits< int16\_t >, [437](#)
  - ElementTraits< int32\_t >, [437](#)
  - ElementTraits< int64\_t >, [438](#)
  - ElementTraits< int8\_t >, [438](#)
  - ElementTraits< Reclnt::rint< K > >, [438](#)
  - ElementTraits< Reclnt::rmint< K, MG > >, [439](#)
  - ElementTraits< Reclnt::ruint< K > >, [439](#)
  - ElementTraits< uint16\_t >, [439](#)
  - ElementTraits< uint32\_t >, [440](#)
  - ElementTraits< uint64\_t >, [440](#)
  - ElementTraits< uint8\_t >, [440](#)
  - has\_minus\_eq\_impl< C >, [469](#)
  - has\_minus\_impl< C >, [470](#)
  - has\_mul\_eq\_impl< C >, [470](#)
  - has\_mul\_impl< C >, [470](#)
  - has\_operation< T >, [471](#)
  - has\_plus\_eq\_impl< C >, [471](#)
  - has\_plus\_impl< C >, [471](#)
  - is\_simd< T >, [479](#)
  - ModeTraits< Field >, [513](#)



- ModeTraits< Givaro::Modular< Element, Compute > >, [513](#)
- ModeTraits< Givaro::Modular< Givaro::Integer, Compute > >, [514](#)
- ModeTraits< Givaro::Modular< int16\_t, Compute > >, [514](#)
- ModeTraits< Givaro::Modular< int32\_t, Compute > >, [514](#)
- ModeTraits< Givaro::Modular< int8\_t, Compute > >, [515](#)
- ModeTraits< Givaro::Modular< Reclnt::ruint< K >, Compute > >, [515](#)
- ModeTraits< Givaro::Modular< uint16\_t, Compute > >, [515](#)
- ModeTraits< Givaro::Modular< uint32\_t, Compute > >, [516](#)
- ModeTraits< Givaro::Modular< uint8\_t, Compute > >, [516](#)
- ModeTraits< Givaro::ModularBalanced< Element > >, [516](#)
- ModeTraits< Givaro::ModularBalanced< Givaro::Integer > >, [517](#)
- ModeTraits< Givaro::ModularBalanced< int16\_t > >, [517](#)
- ModeTraits< Givaro::ModularBalanced< int32\_t > >, [518](#)
- ModeTraits< Givaro::ModularBalanced< int8\_t > >, [518](#)
- ModeTraits< Givaro::Montgomery< T > >, [518](#)
- ModeTraits< Givaro::ZRing< double > >, [519](#)
- ModeTraits< Givaro::ZRing< float > >, [519](#)
- ModeTraits< Givaro::ZRing< Givaro::Integer > >, [519](#)
- need\_field\_characteristic< Field >, [520](#)
- need\_field\_characteristic< Givaro::Modular< Field > >, [520](#)
- need\_field\_characteristic< Givaro::ModularBalanced< Field > >, [521](#)
- SimdChooser< T, false, b >, [735](#)
- SimdChooser< T, true, false >, [736](#)
- SimdChooser< T, true, true >, [736](#)
- vand
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [557](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [561](#)
  - Simd128\_impl< true, true, false, 2 >, [577](#)
  - Simd128\_impl< true, true, false, 4 >, [586](#)
  - Simd128\_impl< true, true, false, 8 >, [596](#)
  - Simd128\_impl< true, true, true, 2 >, [605](#)
  - Simd128\_impl< true, true, true, 4 >, [614](#)
  - Simd128\_impl< true, true, true, 8 >, [623](#)
  - Simd128i\_base, [625](#)
  - Simd256\_impl< true, false, true, 8 >, [631](#)
  - Simd256\_impl< true, true, false, 4 >, [660](#)
  - Simd256\_impl< true, true, true, 4 >, [695](#)
  - Simd512\_impl< true, true, false, 8 >, [723](#)
- Simd512\_impl< true, true, true, 8 >, [733](#)
- Simd512i\_base, [735](#)
- vandnot
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [558](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [562](#)
  - Simd128\_impl< true, true, false, 2 >, [577](#)
  - Simd128\_impl< true, true, false, 4 >, [587](#)
  - Simd128\_impl< true, true, false, 8 >, [597](#)
  - Simd128\_impl< true, true, true, 2 >, [605](#)
  - Simd128\_impl< true, true, true, 4 >, [614](#)
  - Simd128\_impl< true, true, true, 8 >, [623](#)
  - Simd128i\_base, [625](#)
  - Simd256\_impl< true, false, true, 8 >, [632](#)
  - Simd256\_impl< true, true, false, 4 >, [660](#)
  - Simd256\_impl< true, true, true, 4 >, [695](#)
  - Simd512\_impl< true, true, false, 8 >, [723](#)
  - Simd512\_impl< true, true, true, 8 >, [733](#)
  - Simd512i\_base, [735](#)
- VEC\_ADD
  - FFLAS::vectorised, [288](#)
- VEC\_SUB
  - FFLAS::vectorised, [288](#)
- vect\_size
  - FieldSimd< \_Field >, [450](#)
  - NoSimd< T >, [522](#)
  - Simd128\_impl< true, true, false, 2 >, [577](#)
  - Simd128\_impl< true, true, false, 4 >, [587](#)
  - Simd128\_impl< true, true, false, 8 >, [597](#)
  - Simd128\_impl< true, true, true, 2 >, [606](#)
  - Simd128\_impl< true, true, true, 4 >, [614](#)
  - Simd128\_impl< true, true, true, 8 >, [623](#)
  - Simd256\_impl< true, false, true, 8 >, [633](#)
  - Simd256\_impl< true, true, false, 2 >, [643](#)
  - Simd256\_impl< true, true, false, 4 >, [660](#)
  - Simd256\_impl< true, true, false, 8 >, [670](#)
  - Simd256\_impl< true, true, true, 2 >, [679](#)
  - Simd256\_impl< true, true, true, 4 >, [695](#)
  - Simd256\_impl< true, true, true, 8 >, [704](#)
  - Simd512\_impl< true, false, true, 8 >, [712](#)
  - Simd512\_impl< true, true, false, 8 >, [723](#)
  - Simd512\_impl< true, true, true, 8 >, [733](#)
- vect\_t
  - FieldSimd< \_Field >, [445](#)
  - NoSimd< T >, [521](#)
  - Simd128\_impl< true, true, false, 2 >, [569](#)
  - Simd128\_impl< true, true, false, 4 >, [579](#)
  - Simd128\_impl< true, true, false, 8 >, [589](#)
  - Simd128\_impl< true, true, true, 2 >, [599](#)
  - Simd128\_impl< true, true, true, 4 >, [608](#)
  - Simd128\_impl< true, true, true, 8 >, [616](#)
  - simd128\_int64.inl, [877](#)
  - Simd128i\_base, [624](#)
  - Simd256\_impl< true, false, true, 8 >, [627](#)
  - Simd256\_impl< true, true, false, 2 >, [635](#)
  - Simd256\_impl< true, true, false, 4 >, [646](#)

- Simd256\_impl< true, true, false, 8 >, [663](#)
- Simd256\_impl< true, true, true, 2 >, [672](#)
- Simd256\_impl< true, true, true, 4 >, [682](#)
- Simd256\_impl< true, true, true, 8 >, [697](#)
- simd256\_int64.inl, [880](#)
- Simd256i\_base, [705](#)
- Simd512\_impl< true, false, true, 8 >, [707](#)
- Simd512\_impl< true, true, false, 8 >, [715](#)
- Simd512\_impl< true, true, true, 8 >, [726](#)
- simd512\_int64.inl, [882](#)
- Simd512i\_base, [734](#)
- verification\_PLUQ
  - benchmark-pluq.C, [802](#)
- verifPLUQ
  - test-lu.C, [1103](#)
- VERSION
  - config.h, [807](#)
- vor
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [557](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [561](#)
  - Simd128\_impl< true, true, false, 2 >, [577](#)
  - Simd128\_impl< true, true, false, 4 >, [586](#)
  - Simd128\_impl< true, true, false, 8 >, [596](#)
  - Simd128\_impl< true, true, true, 2 >, [605](#)
  - Simd128\_impl< true, true, true, 4 >, [614](#)
  - Simd128\_impl< true, true, true, 8 >, [623](#)
  - Simd128i\_base, [625](#)
  - Simd256\_impl< true, false, true, 8 >, [632](#)
  - Simd256\_impl< true, true, false, 4 >, [660](#)
  - Simd256\_impl< true, true, true, 4 >, [695](#)
  - Simd512\_impl< true, true, false, 8 >, [723](#)
  - Simd512\_impl< true, true, true, 8 >, [732](#)
  - Simd512i\_base, [734](#)
- vxor
  - ScalFunctions< Element, typename enable\_if< is\_floating\_point< Element >::value >::type >, [557](#)
  - ScalFunctions< Element, typename enable\_if< is\_integral< Element >::value >::type >, [562](#)
  - Simd128\_impl< true, true, false, 2 >, [577](#)
  - Simd128\_impl< true, true, false, 4 >, [587](#)
  - Simd128\_impl< true, true, false, 8 >, [597](#)
  - Simd128\_impl< true, true, true, 2 >, [605](#)
  - Simd128\_impl< true, true, true, 4 >, [614](#)
  - Simd128\_impl< true, true, true, 8 >, [623](#)
  - Simd128i\_base, [625](#)
  - Simd256\_impl< true, false, true, 8 >, [632](#)
  - Simd256\_impl< true, true, false, 4 >, [660](#)
  - Simd256\_impl< true, true, true, 4 >, [695](#)
  - Simd512\_impl< true, true, false, 8 >, [723](#)
  - Simd512\_impl< true, true, true, 8 >, [733](#)
  - Simd512i\_base, [735](#)
- WAIT
  - parallel.h, [1040](#)
- Winograd, [768](#)
  - FFLAS::BLAS3, [201](#)
  - winograd.C, [776](#)
    - balanced, [777](#)
    - DOUBLE\_TO\_FLOAT\_CROSSOVER, [776](#)
    - GFOPS, [776](#)
    - main, [777](#)
    - TTimer, [776](#)
  - Winograd\_L\_S
    - FFLAS::BLAS3, [204](#)
  - Winograd\_LR\_S
    - FFLAS::BLAS3, [204](#)
  - Winograd\_R\_S
    - FFLAS::BLAS3, [205](#)
  - WinogradAcc\_2\_24
    - FFLAS::BLAS3, [202](#)
  - WinogradAcc\_2\_27
    - FFLAS::BLAS3, [202](#)
  - WinogradAcc\_3\_21
    - FFLAS::BLAS3, [202](#)
  - WinogradAcc\_3\_23
    - FFLAS::BLAS3, [201](#)
  - WinogradAcc\_L\_S
    - FFLAS::BLAS3, [204](#)
  - WinogradAcc\_LR
    - FFLAS::BLAS3, [203](#)
  - WinogradAcc\_R\_S
    - FFLAS::BLAS3, [203](#)
  - WinogradCalc
    - FFLAS::Protected, [225](#)
  - WinogradPar, [768](#)
  - WinogradSteps
    - FFLAS::Protected, [224](#)
  - WinogradThreshold
    - FFLAS::Protected, [223](#), [224](#)
  - WinoPar
    - FFLAS::BLAS3, [200](#)
  - WINOTHRESHOLD
    - fflas.h, [829](#)
- WRITE
  - parallel.h, [1041](#)
- write
  - RNSInteger< RNS >, [548](#)
  - RNSIntegerMod< RNS >, [553](#)
- write\_field
  - Matio.h, [1056](#)
- write\_matrix
  - benchmark-fgemv-mp.C, [791](#)
  - RNSIntegerMod< RNS >, [553](#)
- write\_matrix\_long
  - RNSIntegerMod< RNS >, [554](#)
- writeCommandString
  - FFLAS, [194](#)
- writeDnsFormat
  - FFLAS, [158](#)
- WriteMatrix
  - FFLAS, [194](#), [196](#)
- WritePermutation
  - FFLAS, [196](#)



## zero

FFLAS, [83](#)  
FieldSimd< \_Field >, [447](#), [448](#)  
RNSInteger< RNS >, [548](#)  
RNSIntegerMod< RNS >, [555](#)  
ScalFunctions< Element, typename enable\_if<  
    is\_floating\_point< Element >::value >::type  
    >, [557](#)  
ScalFunctions< Element, typename enable\_if<  
    is\_integral< Element >::value >::type >, [561](#)  
Simd128\_impl< true, true, false, 2 >, [576](#)  
Simd128\_impl< true, true, false, 4 >, [586](#)  
Simd128\_impl< true, true, false, 8 >, [596](#)  
Simd128\_impl< true, true, true, 2 >, [605](#)  
Simd128\_impl< true, true, true, 4 >, [613](#)  
Simd128\_impl< true, true, true, 8 >, [622](#)  
Simd128i\_base, [625](#)  
Simd256\_impl< true, false, true, 8 >, [628](#)  
Simd256\_impl< true, true, false, 2 >, [643](#)  
Simd256\_impl< true, true, false, 4 >, [660](#)  
Simd256\_impl< true, true, false, 8 >, [670](#)  
Simd256\_impl< true, true, true, 2 >, [679](#)  
Simd256\_impl< true, true, true, 4 >, [694](#), [695](#)  
Simd256\_impl< true, true, true, 8 >, [704](#)  
Simd256i\_base, [705](#)  
Simd512\_impl< true, false, true, 8 >, [708](#)  
Simd512\_impl< true, true, false, 8 >, [723](#)  
Simd512\_impl< true, true, true, 8 >, [732](#)  
Simd512i\_base, [734](#)

## ZOSparseMatrix

FFLAS, [79](#)