FflasFfpack

Generated on Thu Jan 16 2025 00:00:00 for FflasFfpack by Doxygen 1.13.2

Thu Jan 16 2025 00:00:00

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 specifities 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

2.5.0

1.6 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`

1.7 Tutorial

no doc.

1.8 Configuring and Installing FFLAS-FFPACK

FFLAS-FFPACK is a header-only package.

Hower 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.

1.9 Architecture of the library.

no doc.

Chapter 2

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 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::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::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::Element_ptr 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::freduce (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 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::fscalin (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::fscalin (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)

α 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)

α 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 ldx, 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 incx 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 ldX 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, const int nbw, const bool par, RandIter &G)

test for ldX equal

test for transpo

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

does not belong here

Chapter 3

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::productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, size_t t, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t *Ku, const size_t *Tu, const size_t *MU, typename Field::ConstElement_ptr Xl, size_t ldl, size_t NbBlocksL, const size_t *Kl, const size_t *Tl, const size_t *ML, typename Field::Element_ptr B, size_t ldb, const typename Field::Element beta, typename Field::Element_ptr D, size_t ldd)

Pernet C. and Storjohann A. `Time` *and space efficient generators for quasiseparable matrices* , JSC (85), 2018, doi:10.1016/j.jsc.2017.07.010

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 ftrsmLeftUpperNoTransNonUnit< Element >

- Dumas, Giorgi, Pernet 06, arXiv:cs/0601133.

Chapter 4

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 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 triagular matrix

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 triagular 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::fscalin (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::fscalin (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::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::MatF2MatFl_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)

swap to save space ??

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

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_redcolechelon (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_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 lda

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 lda

Chapter 5

Topic Index

5.1 Topics

Here is a list of all topics with brief descriptions:

Chapter 6

Namespace Index

6.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 7

Hierarchical Index

7.1 Class Hierarchy

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

Chapter 8

Data Structure Index

8.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 9

File Index

9.1 File List

Here is a list of all files with brief descriptions:

Chapter 10

Topic Documentation

10.1 CHECKER

Class CHECKER provides functions to verify computations in FFLAS and FFPACK.

Class CHECKER provides functions to verify computations in FFLAS and FFPACK.

10.2 FFLAS-FFPACK

the FFLAS FFPACK library

Topics

- FFLAS
  The C-style wrapper of BLAS for finite field linear algebra.
- Interfaces
  Intefaces for FFLAS-FFPACK.

10.2.1 Detailed Description

the FFLAS FFPACK library

C++ header library for fast exact dense linear algebra

See also

FFLAS
FFPACK

10.2.2 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.

10.2.3 Interfaces

Intefaces for FFLAS-FFPACK.

Intefaces for FFLAS-FFPACK.

C interface in folder

See also

libs

10.3 Matrix Multiplication Algorithms

Matrix Multiplication (level 3) algorithms.

Files

- file schedule_bini.inl

  Bini implementation.

10.3.1 Detailed Description

Matrix Multiplication (level 3) algorithms.

Todo biblio

10.4 SIMD wrapper

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

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

Todo biblio

10.5 FFPACK

Class FFPACK provides functions using fflas much as Lapack uses BLAS.

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

10.5.1 Detailed Description

Class FFPACK provides functions using fflas much as Lapack uses BLAS.

10.6 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

10.6.1 Detailed Description

fields in the FFLAS-FFPACK library

Unparametric/Random elements

Todo biblio

10.7 RNS

just include them all

just include them all

Chapter 11

Namespace Documentation

11.1 FFLAS Namespace Reference

Namespaces

- namespace _ftranspose_impl
- 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 BlockTransposeSIMD
- struct Checker_Empty
- class CheckerImplem_fgemm
- class CheckerImplem_ftrsm
- 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< RecInt::rint< K > >
- struct ElementTraits< RecInt::rmint< K, MG > >
- struct ElementTraits< RecInt::ruint< K > >
- struct ElementTraits< uint16_t >
- struct ElementTraits< uint32_t >
- struct ElementTraits< uint64_t >
- struct ElementTraits< uint8_t >
- struct EllMat
- 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< RecInt::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 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< int64_t, uint64_t > >
- struct ModeTraits< Givaro::Modular< int8_t, Compute > >
- struct ModeTraits< Givaro::Modular< RecInt::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::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_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
- 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 , FflasLeftTri = 123 , FflasRightTri = 124 }

  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 faddin (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fadding for symmetric matrices
- 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: 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: **F**ield **GE**neral **M**atrix **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: 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>
  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)
- 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, 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)
- 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::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)
- 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)

  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 freduce (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::Element_ptr A, const size_t lda)

  freduce for square symmetric matrices
- 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)
- 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: **F**reivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.*
- template<class Field>
  void fscalin (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x$.
- 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 fscalin (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::↩ Element a, Givaro::DoubleDomain::Element_ptr y, const size_t incy)
- template<> void fscalin (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 fscalin (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)

  fscalin $A \leftarrow a \cdot A$.
- 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 fscalin (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 fscalin (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 fscalin (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 fscalin (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::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq)
- 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, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- 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, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >, ParSeqHelper::Sequential > &H)
- 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, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)
- 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, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)
- template<class Field, typename Mode>
  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, MMHelper< Field, MMHelperAlgo::DivideAndConquer, Mode > &H)

- 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, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)
- Givaro::FloatDomain::Element_ptr fsyrk (const Givaro::FloatDomain &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const Givaro::FloatDomain::Element alpha, Givaro::FloatDomain::ConstElement_ptr A, const size_t lda, const Givaro::FloatDomain::Element beta, Givaro::FloatDomain::Element_ptr C, const size_t ldc, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- Givaro::DoubleDomain::Element_ptr fsyrk (const Givaro::DoubleDomain &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const Givaro::DoubleDomain::Element alpha, Givaro::DoubleDomain::ConstElement_ptr A, const size_t lda, const Givaro::DoubleDomain::Element beta, Givaro::DoubleDomain::Element_ptr C, const size_t ldc, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- 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 > &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, class FieldTrait>
  void computeS1S2 (const Field &F, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element x, const typename Field::Element y, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr S, const size_t lds, typename Field::Element_ptr T, const size_t ldt, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- 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, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)
- template<class Field, class Mode>
  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, MMHelper< Field, MMHelperAlgo::Winograd, Mode > &H)
- template<class Field, class FieldTrait>
  Field::Element_ptr fsyrk_strassen (const Field &F, const FFLAS_UPLO uplo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element y1, const typename Field::Element

y2, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)

- 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**iangular **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**iangular **M**atrix **M**ultiply with 3 operands Computes* $C \leftarrow \alpha \mathrm{op}(A)B + betaC$ *or* $C \leftarrow \alpha B\mathrm{op}(A) + betaC$*.*
- 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 $X \leftarrow \mathrm{op}(A^{-1})X$
- 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 $x \leftarrow ymodF$*.*
- 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 $x \leftarrow ymodF$*.*

- 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>
  void fzero (const Field &F, const FFLAS_UPLO shape, const FFLAS_DIAG diag, const size_t n, typename Field::Element_ptr A, const size_t lda)

  fzero : $A \leftarrow 0$ *for a triangular matrix.*
- 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 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 $X \leftarrow \mathrm{op}(A)X$
- 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: **TR**iangular **S**ystem solve with **M**atrix.*
- template<class Field, typename FieldTrait>
  Field::Element_ptr fsyrk_strassen (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element y1, const typename Field::Element y2, 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, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &H)
- 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, typenameField::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 > 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)
- 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<typename Field, typename Simd = Simd<typename Field::Element>, size_t bs = FFLAS_TRANSPOSE_BLOCKSIZE, typename std::enable_if< Simd::template is_same_element< Field >::value >::type * = nullptr, typename std::enable_if< bs > = 1 && bs % Simd::vect_size == 0, ::type * = nullptr>
  Field::Element_ptr ftranspose (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

- template<class Field, class enable = void>
  Field::Residu_t maxCardinality ()
- template<> uint64_t maxCardinality< Givaro::Modular< int64_t > > ()
- template<> uint32_t maxCardinality< Givaro::Modular< int32_t > > ()
- template<class Field>
  Field::Residu_t minCardinality ()
- 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)

  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 fscalin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT *X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x$.
- 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)

  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)

  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 $A \leftarrow AmodF$.
- 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 $A \leftarrow BmodF$.
- 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 $A \leftarrow BmodF$.
- 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 fscalin (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)

  fscalin $A \leftarrow a \cdot A$.
- 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)

  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 : 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 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: TRiangular System solve with Vector Computes $X \leftarrow \mathrm{op}(A^{-1})X$

- 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.

- 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)
- char ∗ getArgumentValue (int argc, char ∗∗argv, int i)

  Get the value of an argument and avoid core dump when no value was given after an argument.
- 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 &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.
- 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 ()

11.1.1 Typedef Documentation

11.1.1.1 Checker_fgemm

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

11.1.1.2 Checker_ftrsm

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

11.1.1.3 ForceCheck_fgemm

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

11.1.1.4 ForceCheck_ftrsm

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

11.1.1.5 ZOSparseMatrix

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

11.1.1.6 NotZOSparseMatrix

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

11.1.1.7 SimdSparseMatrix

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

11.1.1.8 NoSimdSparseMatrix

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

11.1.1.9 MKLSparseMatrixFormat

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

11.1.1.10 NotMKLSparseMatrixFormat

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

11.1.1.11 has_plus

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

11.1.1.12 has_minus

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

11.1.1.13 has_equal

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

11.1.1.14 has_plus_eq

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

11.1.1.15 has_minus_eq

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

11.1.1.16 has_mul

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

11.1.1.17 has_mul_eq

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

11.1.1.18 Timer

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

11.1.1.19 BaseTimer

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

11.1.1.20 UserTimer

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

11.1.1.21 SysTimer

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

11.1.2 Enumeration Type Documentation

11.1.2.1 FFLAS_ORDER

```
enum FFLAS_ORDER
```

Storage by row or col ?

Enumerator

FflasRowMajor	row major
FflasColMajor	col major

11.1.2.2 FFLAS_TRANSPOSE

```
enum FFLAS_TRANSPOSE
```

Is matrix transposed ?

Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.

11.1.2.3 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 $T_{i,j} = 0$)
FflasLeftTri	Triangular matrix is Left triangular (if $j > n - i - 1$ then $T_{i,j} = 0$)
FflasRightTri	Triangular matrix is Right triangular (if $j < n - i - 1$ then $T_{i,j} = 0$)

11.1.2.4 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 ($T_{i,i} = 1$)

11.1.2.5 FFLAS_SIDE

`enum FFLAS_SIDE`

On what side ?

Enumerator

FflasLeft	Operator applied on the left.
FflasRight	Operator applied on the rigth.

11.1.2.6 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

11.1.2.7 number_kind

`enum number_kind`

Enumerator

zero	
one	
mone	
other	

11.1.2.8 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	

11.1.2.9 FFLAS_FORMAT

```
enum FFLAS_FORMAT
```

Enumerator

FflasAuto	
FflasDense	
FflasSMS	
FflasBinary	
FflasMath	
FflasMaple	
FflasSageMath	

11.1.3 Function Documentation

11.1.3.1 InfNorm()

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

11.1.3.2 min3()

```
template<class T>
const T & min3 (
            const T & m,
            const T & n,
            const T & k)
```

11.1.3.3 max3()

```
template<class T>
const T & max3 (
            const T & m,
            const T & n,
            const T & k)
```

11.1.3.4 min4()

```
template<class T>
const T & min4 (
            const T & m,
            const T & n,
            const T & k,
            const T & l)
```

11.1.3.5 max4()

```
template<class T>
const T & max4 (
            const T & m,
            const T & n,
            const T & k,
            const T & l)
```

11.1.3.6 fadd() [1/8]

```
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)
```

11.1.3.7 faddin() [1/5]

```
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)
```

11.1.3.8 fsub() [1/4]

```
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)
```

11.1.3.9 fsubin() [1/3]

```
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)
```

11.1.3.10 fadd() [2/8]

```
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)
```

Todo optimise here

11.1.3.11 pfadd()

```
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)
```

11.1.3.12 pfsub()

```
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)
```

11.1.3.13 pfaddin()

```
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)
```

11.1.3.14 pfsubin()

```
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)
```

11.1.3.15 fadd() [3/8]

```
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.

Computes C = A + B.

Parameters

F	field
M	rows
N	cols
A	dense matrix of size MxN
lda	leading dimension of A
B	dense matrix of size MxN
ldb	leading dimension of B
C	dense matrix of size MxN
ldc	leading dimension of C

11.1.3.16 fsub() [2/4]

```
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.

Computes C = A - B.

Parameters

F	field
M	rows
N	cols
A	dense matrix of size MxN

Parameters

lda	leading dimension of A
B	dense matrix of size MxN
ldb	leading dimension of B
C	dense matrix of size MxN
ldc	leading dimension of C

11.1.3.17 faddin() [2/5]

```
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

11.1.3.18 faddin() [3/5]

```
template<class Field>
void faddin (
            const Field & F,
            const FFLAS_UPLO uplo,
            const size_t N,
            typename Field::ConstElement_ptr B,
            const size_t ldb,
            typename Field::Element_ptr C,
            const size_t ldc)
```

fadding for symmetric matrices

11.1.3.19 fsubin() [2/3]

```
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

11.1.3.20 fadd() [4/8]

```
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.

Computes `C` = `A` + alpha `B`.

Parameters

F	field
M	rows
N	cols
A	dense matrix of size `MxN`
lda	leading dimension of `A`
alpha	some scalar
B	dense matrix of size `MxN`
ldb	leading dimension of `B`
C	dense matrix of size `MxN`
ldc	leading dimension of `C`

11.1.3.21 fassign() [1/10]

```
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)  [inline]
```

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`

11.1.3.22 fassign() [2/10]

```
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)  [inline]
```

11.1.3.23 fassign() [3/10]

```
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)  [inline]
```

11.1.3.24 fassign() [4/10]

```
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)  [inline]
```

11.1.3.25 fassign() [5/10]

```
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)  [inline]
```

11.1.3.26 fassign() [6/10]

```
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)  [inline]
```

11.1.3.27 fassign() [7/10]

```
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)  [inline]
```

11.1.3.28 fassign() [8/10]

```
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 : $A \leftarrow B$.

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

11.1.3.29 faxpy() [1/6]

```
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)  [inline]
```

faxpy : $y \leftarrow \alpha \cdot x + y$.

Parameters

	F	field

Parameters

	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

11.1.3.30 faxpy() [2/6]

```
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)  [inline]
```

11.1.3.31 faxpy() [3/6]

```
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)  [inline]
```

11.1.3.32 faxpy() [4/6]

```
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)  [inline]
```

faxpy : $y \leftarrow \alpha \cdot x + y$.

Parameters

	F	field

Parameters

	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

11.1.3.33 fdot() [1/11]

```
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)  [inline]
```

11.1.3.34 fdot() [2/11]

```
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)  [inline]
```

11.1.3.35 fdot() [3/11]

```
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)  [inline]
```

11.1.3.36 fdot() [4/11]

```
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)  [inline]
```

11.1.3.37 fdot() [5/11]

```
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)  [inline]
```

11.1.3.38 fdot() [6/11]

```
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)  [inline]
```

11.1.3.39 fdot() [7/11]

```
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)  [inline]
```

11.1.3.40 fdot() [8/11]

```
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) [inline]
```

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

11.1.3.41 fgemm() [1/23]

```
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::MachineFl
>, ParSeqHelper::Sequential > & H) [inline]
```

11.1.3.42 fgemm() [2/23]

```
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)  [inline]
```

11.1.3.43 fgemm() [3/23]

```
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)  [inline]
```

11.1.3.44 fgemm() [4/23]

```
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)  [inline]
```

fgemm: **F**ield **GE**neral **M**atrix **M**ultiply.

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

Parameters

F	field.
ta	if `ta==FflasTrans` then $\mathrm{op}(A) = A^t$, else $\mathrm{op}(A) = A$,
tb	same for matrix `B`
m	see `A`
n	see `B`
k	see `A`
alpha	scalar
beta	scalar
A	$\mathrm{op}(A)$ is $m \times k$
B	$\mathrm{op}(B)$ 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`
w	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of `w`.

Warning

α *must* be invertible

11.1.3.45 fgemm() [5/23]

```
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)  [inline]
```

11.1.3.46 fgemm() [6/23]

```
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)  [inline]
```

11.1.3.47 fsquare() [1/6]

```
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)  [inline]
```

fsquare: Squares a matrix.

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

Parameters

ta	if `ta==FflasTrans`, $\mathrm{op}(A) = A^T$.
F	field
n	size of `A`
alpha	scalar
beta	scalar
A	dense matrix of size `nxn`
lda	leading dimension of `A`
C	dense matrix of size `nxn`
ldc	leading dimension of `C`

Bug why double ?

11.1.3.48 fsquare() [2/6]

```
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)  [inline]
```

11.1.3.49 fsquare() [3/6]

```
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)  [inline]
```

11.1.3.50 fsquare() [4/6]

```
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)  [inline]
```

11.1.3.51 fsquare() [5/6]

```
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)  [inline]
```

11.1.3.52 fgemm() [7/23]

```
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)  [inline]
```

11.1.3.53 fgemm() [8/23]

```
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)  [inline]
```

11.1.3.54 fgemm() [9/23]

```
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)  [inline]
```

11.1.3.55 fgemm() [10/23]

```
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)  [inline]
```

11.1.3.56 fgemm() [11/23]

```
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)  [inline]
```

11.1.3.57 fgemm() [12/23]

```
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) [inline]
```

11.1.3.58 fgemm() [13/23]

```
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) [inline]
```

11.1.3.59 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]
```

11.1.3.60 fgemm() [15/23]

```
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) [inline]
```

11.1.3.61 fgemm() [16/23]

```
template<size_t K1, size_t K2, class ParSeq>
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]
```

11.1.3.62 fgemm() [17/23]

```
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)  [inline]
```

11.1.3.63 fgemm() [18/23]

```
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)  [inline]
```

11.1.3.64 fgemv() [1/19]

```
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::MachineFlo
> > & H)  [inline]
```

11.1.3.65 fgemv() [2/19]

```
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)  [inline]
```

11.1.3.66 fgemv() [3/19]

```
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)  [inline]
```

11.1.3.67 fgemv() [4/19]

```
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)  [inline]
```

11.1.3.68 fgemv() [5/19]

```
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)  [inline]
```

finite prime Field GEneral Matrix Vector multiplication.

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

Parameters

	F	field
	TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
	M	rows
	N	cols
	alpha	scalar
	A	dense matrix of size `MxN`
	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`

11.1.3.69 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]
```

11.1.3.70 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]
```

11.1.3.71 fgemv() [8/19]

```
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)
[inline]
```

11.1.3.72 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]
```

11.1.3.73 fgemv() [10/19]

```
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)
```

11.1.3.74 fgemv() [11/19]

```
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)
```

11.1.3.75 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::Defau
> & H)  [inline]
```

11.1.3.76 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]
```

11.1.3.77 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]
```

11.1.3.78 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]
```

11.1.3.79 fgemv() [16/19]

```
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)  [inline]
```

11.1.3.80 fger() [1/12]

```
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)  [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 MxN 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

11.1.3.81 fger() [2/12]

```
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::MachineFlo
> > & H)  [inline]
```

11.1.3.82 fger() [3/12]

```
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)  [inline]
```

11.1.3.83 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]
```

11.1.3.84 fger() [5/12]

```
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)
[inline]
```

11.1.3.85 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]
```

11.1.3.86 fger() [7/12]

```
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)  [inline]
```

11.1.3.87 fger() [8/12]

```
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)  [inline]
```

11.1.3.88 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]
```

11.1.3.89 fger() [10/12]

```
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)  [inline]
```

11.1.3.90 fger() [11/12]

```
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) [inline]
```

11.1.3.91 freduce() [1/11]

```
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 $x \leftarrow ymodF$.

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

11.1.3.92 freduce() [2/11]

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

freduce $x \leftarrow xmodF$.

Parameters

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

Bug use cblas_(d)scal when possible

11.1.3.93 freduce_constoverride() [1/2]

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

11.1.3.94 finit() [1/8]

```
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)
```

11.1.3.95 finit() [2/8]

```
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$.

Parameters

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

11.1.3.96 freduce() [3/11]

```
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 $A \leftarrow AmodF$.

Parameters

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

11.1.3.97 freduce() [4/11]

```
template<class Field>
void freduce (
            const Field & F,
            const FFLAS_UPLO UpLo,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda)
```

freduce for square symmetric matrices

11.1.3.98 pfreduce()

```
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)
```

11.1.3.99 freduce() [5/11]

```
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 $A \leftarrow BmodF$.

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`

11.1.3.100 freduce_constoverride() [2/2]

```
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)
```

11.1.3.101 finit() [3/8]

```
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 $A \leftarrow BmodF$.

Parameters

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

11.1.3.102 finit() [4/8]

```
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)
```

11.1.3.103 freduce() [6/11]

```
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)  [inline]
```

11.1.3.104 freduce() [7/11]

```
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)  [inline]
```

11.1.3.105 freivalds()

```
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)  [inline]
```

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

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

Parameters

F	field.
ta	if `ta==FflasTrans` then $\mathrm{op}(A) = A^t$, else $\mathrm{op}(A) = A$,
tb	same for matrix `B`
m	see `A`
n	see `B`
k	see `A`
alpha	scalar
A	$\mathrm{op}(A)$ is $m \times k$
B	$\mathrm{op}(B)$ 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`

11.1.3.106 fscalin() [1/10]

```
template<class Field>
void fscalin (
            const Field & F,
            const size_t n,
            const typename Field::Element alpha,
            typename Field::Element_ptr X,
            const size_t incX) [inline]
```

fscalin $x \leftarrow \alpha \cdot x$.

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.

11.1.3.107 fscal() [1/10]

```
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) [inline]
```

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

Parameters

	F	field
	n	size of the vectors
	alpha	scalar
in	*X*	vector in `F`
	incX	stride of `X`
out	*Y*	vector in `F`
	incY	stride of `Y`

Bug use cblas_(d)scal when possible

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

11.1.3.108 fscal() [2/10]

```
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)  [inline]
```

11.1.3.109 fscal() [3/10]

```
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)  [inline]
```

11.1.3.110 fscalin() [2/10]

```
template<>
void fscalin (
            const Givaro::DoubleDomain & ,
            const size_t N,
            const Givaro::DoubleDomain::Element a,
            Givaro::DoubleDomain::Element_ptr y,
            const size_t incy)  [inline]
```

11.1.3.111 fscalin() [3/10]

```
template<>
void fscalin (
            const Givaro::FloatDomain & ,
            const size_t N,
            const Givaro::FloatDomain::Element a,
            Givaro::FloatDomain::Element_ptr y,
            const size_t incy)  [inline]
```

11.1.3.112 fscalin() [4/10]

```
template<class Field>
void fscalin (
            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]
```

fscalin $A \leftarrow a \cdot A$.

Parameters

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

11.1.3.113 fscal() [4/10]

```
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) [inline]
```

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

11.1.3.114 fscalin() [5/10]

```
template<>
void fscalin (
            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]
```

11.1.3.115 fscal() [5/10]

```
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)  [inline]
```

11.1.3.116 fscalin() [6/10]

```
template<>
void fscalin (
            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]
```

11.1.3.117 fscal() [6/10]

```
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)  [inline]
```

11.1.3.118 fscalin() [7/10]

```
template<>
void fscalin (
            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]
```

11.1.3.119 fscal() [7/10]

```
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)  [inline]
```

11.1.3.120 fscalin() [8/10]

```
template<>
void fscalin (
        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]
```

11.1.3.121 fscal() [8/10]

```
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)  [inline]
```

11.1.3.122 fsyr2k()

```
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)  [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

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

Warning

α *must* be invertible

11.1.3.123 fsyrk() [1/16]

```
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)  [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

F	field.
UpLo	whether to compute the upper or the lower triangular part of the symmetric matrix C
trans	if ta==FflasNoTrans then comput $C = \alpha A \times A^T + \beta C$, 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
beta	scalar
C	C is $n \times n$
ldc	leading dimension of C

Warning

α *must* be invertible

11.1.3.124 fsyrk() [2/16]

```
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,
            const ParSeqHelper::Sequential seq) [inline]
```

11.1.3.125 fsyrk() [3/16]

```
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,
            MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > & H) [inline]
```

11.1.3.126 fsyrk() [4/16]

```
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,
            MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFlo
>, ParSeqHelper::Sequential > & H) [inline]
```

11.1.3.127 fsyrk() [5/16]

```
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,
            MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > & H) [inline]
```

11.1.3.128 fsyrk() [6/16]

```
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,
            MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > & H) [inline]
```

11.1.3.129 fsyrk() [7/16]

```
template<class Field, typename Mode>
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,
            MMHelper< Field, MMHelperAlgo::DivideAndConquer, Mode > & H) [inline]
```

11.1.3.130 fsyrk() [8/16]

```
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,
            MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > & H)
[inline]
```

11.1.3.131 fsyrk() [9/16]

```
Givaro::FloatDomain::Element_ptr fsyrk (
            const Givaro::FloatDomain & F,
            const FFLAS_UPLO UpLo,
            const FFLAS_TRANSPOSE trans,
            const size_t N,
            const size_t K,
            const Givaro::FloatDomain::Element alpha,
            Givaro::FloatDomain::ConstElement_ptr A,
            const size_t lda,
            const Givaro::FloatDomain::Element beta,
            Givaro::FloatDomain::Element_ptr C,
            const size_t ldc,
            MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H) [inline]
```

11.1.3.132 fsyrk() [10/16]

```
Givaro::DoubleDomain::Element_ptr fsyrk (
            const Givaro::DoubleDomain & F,
            const FFLAS_UPLO UpLo,
            const FFLAS_TRANSPOSE trans,
            const size_t N,
            const size_t K,
            const Givaro::DoubleDomain::Element alpha,
            Givaro::DoubleDomain::ConstElement_ptr A,
            const size_t lda,
            const Givaro::DoubleDomain::Element beta,
            Givaro::DoubleDomain::Element_ptr C,
            const size_t ldc,
            MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag
> & H) [inline]
```

11.1.3.133 fsyrk() [11/16]

```
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) [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 `ta==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

α *must* be invertible

11.1.3.134 fsyrk() [12/16]

```
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) [inline]
```

11.1.3.135 fsyrk() [13/16]

```
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) [inline]
```

11.1.3.136 fsyrk() [14/16]

```
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) [inline]
```

fsyrk: Symmetric Rank K update with diagonal scaling

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

F	field.
UpLo	whether to compute the upper or the lower triangular part of the symmetric matrix C
trans	if ta==FflasNoTrans then compute $C = \alpha ADeltaD \times A^T + \beta C$, else $C = \alpha A^T DeltaD \times A + \beta C$
n	see B
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
twoBlocks	a vector boolean indicating the beginning of each 2x2 blocs in Delta
lda	leading dimension of A
beta	scalar
C	C is $n \times n$
ldc	leading dimension of C

Warning

α *must* be invertible

11.1.3.137 computeS1S2()

```
template<class Field, class FieldTrait>
void computeS1S2 (
            const Field & F,
            const FFLAS_TRANSPOSE trans,
            const size_t N,
            const size_t K,
            const typename Field::Element x,
            const typename Field::Element y,
            typename Field::Element_ptr A,
            const size_t lda,
            typename Field::Element_ptr S,
            const size_t lds,
            typename Field::Element_ptr T,
            const size_t ldt,
            MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH)  [inline]
```

11.1.3.138 fsyrk() [15/16]

```
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,
            MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential
> & H) [inline]
```

11.1.3.139 fsyrk() [16/16]

```
template<class Field, class Mode>
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,
            MMHelper< Field, MMHelperAlgo::Winograd, Mode > & H) [inline]
```

11.1.3.140 fsyrk_strassen() [1/2]

```
template<class Field, class FieldTrait>
Field::Element_ptr fsyrk_strassen (
            const Field & F,
            const FFLAS_UPLO uplo,
            const FFLAS_TRANSPOSE trans,
            const size_t N,
            const size_t K,
            const typename Field::Element y1,
            const typename Field::Element y2,
            const typename Field::Element alpha,
            typename Field::Element_ptr A,
            const size_t lda,
            const typename Field::Element beta,
            typename Field::Element_ptr C,
            const size_t ldc,
            MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & WH) [inline]
```

11.1.3.141 ftrmm() [1/3]

```
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)  [inline]
```

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

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

Parameters

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

11.1.3.142 ftrmm() [2/3]

```
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)  [inline]
```

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

Parameters

F	field
Side	if `Side==FflasLeft` then $B \leftarrow \alpha \mathrm{op}(A) B$ is computed.

Parameters

Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is implicitly unit.
M	rows of `B`
N	cols of `B`
alpha	scalar
A	triangular matrix. If `Side==FflasLeft` then `A` is $N \times N$, otherwise `A` is $M \times M$
lda	leading dim of `A`
B	matrix of size `MxN`
ldb	leading dim of `B`
beta	scalar
C	matrix of size `MxN`
ldc	leading dim of `C`

11.1.3.143 ftrsm() [1/9]

```
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)   [inline]
```

11.1.3.144 ftrsm() [2/9]

```
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)   [inline]
```

11.1.3.145 ftrsm() [3/9]

```
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)  [inline]
```

11.1.3.146 ftrsm() [4/9]

```
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)  [inline]
```

11.1.3.147 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]
```

11.1.3.148 cblas_imptrsm()

```
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) [inline]
```

11.1.3.149 ftrsv() [1/2]

```
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) [inline]
```

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

Parameters

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

11.1.3.150 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]
```

11.1.3.151 finit() [5/8]

```
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 $x \leftarrow ymodF$.

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

11.1.3.152 fconvert() [1/3]

```
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 $x \leftarrow ymodF$.

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

11.1.3.153 fnegin() [1/4]

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

fnegin $x \leftarrow -x$.

Parameters

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

Bug use cblas_(d)scal when possible

11.1.3.154 fneg() [1/4]

```
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 $x \leftarrow -y$.

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

11.1.3.155 fzero() [1/5]

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

fzero : $A \leftarrow 0$.

Parameters

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

11.1.3.156 frand() [1/2]

```
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 : $A \leftarrow random$.

Parameters

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

11.1.3.157 fiszero() [1/4]

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

fiszero : test $X = 0$.

Parameters

F	field
n	vector dimension
X	vector in F
incX	increment of X

11.1.3.158 fequal() [1/4]

```
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 $X = Y$.

Parameters

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

11.1.3.159 faxpby() [1/2]

```
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 : $y \leftarrow \alpha \cdot x + \beta \cdot y$.

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

11.1.3.160 fdot() [9/11]

```
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)  [inline]
```

11.1.3.161 fswap() [1/2]

```
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: $X \leftrightarrow Y$.

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

11.1.3.162 fzero() [2/5]

```
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 : $A \leftarrow 0$.

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

11.1.3.163 fzero() [3/5]

```
template<class Field>
void fzero (
            const Field & F,
            const FFLAS_UPLO shape,
            const FFLAS_DIAG diag,
            const size_t n,
            typename Field::Element_ptr A,
            const size_t lda)
```

fzero : $A \leftarrow 0$ for a triangular matrix.

Parameters

F	field
shape	shape of the triangular matrix
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

11.1.3.164 frand() [2/2]

```
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 : $A \leftarrow random.$

Parameters

F	field
G	randomiterator
m	number of rows to randomize
n	number of cols to randomize
A	matrix in `F`
lda	stride of `A`

11.1.3.165 fequal() [2/4]

```
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 $A = B$.

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

11.1.3.166 fiszero() [2/4]

```
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 $A = 0$.

Parameters

F	field
m	row dimension
n	column dimension
A	m x n matrix in `F`
lda	leading dimension of A

11.1.3.167 fidentity() [1/4]

```
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

11.1.3.168 fidentity() [2/4]

```
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

11.1.3.169 finit() [6/8]

```
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$.

Parameters

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

11.1.3.170 fconvert() [2/3]

```
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 $A \leftarrow BmodF$.

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

11.1.3.171 fnegin() [2/4]

```
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 $A \leftarrow -A$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in `F`
lda	stride of `A`

Todo check if n == lda

11.1.3.172 fneg() [2/4]

```
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 $A \leftarrow -B$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in `F`
lda	stride of `A`

Todo check if n == lda

11.1.3.173 faxpby() [2/2]

```
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 : $y \leftarrow \alpha \cdot x + \beta \cdot y$.

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

11.1.3.174 fmove() [1/2]

```
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 $B \leftarrow 0$.

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

11.1.3.175 bitsize()

```
template<class Field>
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 MxN
lda	leading dimension of A

11.1.3.176 bitsize< Givaro::ZRing< Givaro::Integer > >()

```
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)  [inline]
```

11.1.3.177 ftrmv()

```
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 $X \leftarrow \mathrm{op}(A)X$

Parameters

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

11.1.3.178 ftrsm() [6/9]

```
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: **TR**iangular **S**ystem solve with **M**atrix.

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

Parameters

F	field
Side	if `Side==FflasLeft` then $B \leftarrow \alpha\mathrm{op}(A^{-1})B$ is computed.

Parameters

Uplo	if `Uplo==FflasUpper` then `A` is upper triangular
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is unit.
M	rows of `B`
N	cols of `B`
alpha	scalar
A	triangular invertible matrix. If `Side==FflasLeft` then `A` is $N \times N$, otherwise `A` is $M \times M$
lda	leading dim of `A`
B	matrix of size `MxN`
ldb	leading dim of `B`

Bug α must be non zero.

11.1.3.179 fsyrk_strassen() [2/2]

```
template<class Field, typename FieldTrait>
Field::Element_ptr fsyrk_strassen (
            const Field & F,
            const FFLAS_UPLO UpLo,
            const FFLAS_TRANSPOSE trans,
            const size_t N,
            const size_t K,
            const typename Field::Element y1,
            const typename Field::Element y2,
            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,
            MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > & H)  [inline]
```

11.1.3.180 pfgemm() [1/7]

```
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)
```

11.1.3.181 pfgemm_1D_rec()

```
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)
```

11.1.3.182 pfgemm_2D_rec()

```
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)
```

11.1.3.183 pfgemm_3D_rec()

```
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)
```

11.1.3.184 pfgemm_3D_rec2()

```
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)
```

11.1.3.185 fgemm() [19/23]

```
template<class Field, class ModeTrait, class Strat, class Param>
std::enable_if<!std::is_same< ModeTrait, ModeCategories::ConvertTo< ElementCategories::RNSElementTag
> >::value, typenameField::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]
```

11.1.3.186 ftrsm() [7/9]

```
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) [inline]
```

11.1.3.187 ftrsm() [8/9]

```
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)
[inline]
```

11.1.3.188 fspmv() [1/2]

```
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) [inline]
```

11.1.3.189 fspmm()

```
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) [inline]
```

11.1.3.190 sparse_init() [1/16]

```
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) [inline]
```

11.1.3.191 sparse_init() [2/16]

```
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) [inline]
```

11.1.3.192 sparse_delete() [1/12]

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

11.1.3.193 sparse_delete() [2/12]

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

11.1.3.194 sparse_init() [3/16]

```
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) [inline]
```

11.1.3.195 sparse_init() [4/16]

```
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)  [inline]
```

11.1.3.196 sparse_delete() [3/12]

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

11.1.3.197 sparse_delete() [4/12]

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

11.1.3.198 sparse_print() [1/3]

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

11.1.3.199 sparse_init() [5/16]

```
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)  [inline]
```

11.1.3.200 sparse_init() [6/16]

```
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) [inline]
```

11.1.3.201 sparse_init() [7/16]

```
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) [inline]
```

11.1.3.202 sparse_init() [8/16]

```
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) [inline]
```

11.1.3.203 sparse_delete() [5/12]

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

11.1.3.204 sparse_init() [9/16]

```
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)  [inline]
```

11.1.3.205 sparse_init() [10/16]

```
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)  [inline]
```

11.1.3.206 sparse_init() [11/16]

```
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)  [inline]
```

11.1.3.207 sparse_delete() [6/12]

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

11.1.3.208 sparse_delete() [7/12]

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

11.1.3.209 sparse_init() [12/16]

```
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)  [inline]
```

11.1.3.210 sparse_init() [13/16]

```
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)  [inline]
```

11.1.3.211 sparse_delete() [8/12]

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

11.1.3.212 sparse_delete() [9/12]

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

11.1.3.213 sparse_print() [2/3]

```
template<class Field>
void sparse_print (
            const Sparse< Field, SparseMatrix_t::ELL_simd > & A)  [inline]
```

11.1.3.214 sparse_delete() [10/12]

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

11.1.3.215 sparse_init() [14/16]

```
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) [inline]
```

11.1.3.216 operator<<()

```
template<typename _Field>
std::ostream & operator<< (
            std::ostream & os,
            const Sparse< _Field, SparseMatrix_t::HYB_ZO > & A)
```

11.1.3.217 readSmsFormat()

```
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)
```

11.1.3.218 readSprFormat()

```
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)
```

11.1.3.219 getDataType() [1/4]

```
template<class T>
std::enable_if< std::is_integral< T >::value, int > getDataType ()
```

11.1.3.220 getDataType() [2/4]

```
template<class T>
std::enable_if< std::is_floating_point< T >::value, int > getDataType ()
```

11.1.3.221 getDataType() [3/4]

```
template<class T>
std::enable_if< std::is_same< T, mpz_t >::value, int > getDataType ()
```

11.1.3.222 getDataType() [4/4]

```
template<class T>
int getDataType ()
```

11.1.3.223 readMachineType()

```
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)
```

11.1.3.224 readDnsFormat()

```
template<class Field>
void readDnsFormat (
            const std::string & path,
            const Field & F,
            index_t & rowdim,
            index_t & coldim,
            typename Field::Element_ptr & val)
```

11.1.3.225 writeDnsFormat()

```
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)
```

11.1.3.226 fspmv() [2/2]

```
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 )  [inline]
```

11.1.3.227 sparse_delete() [11/12]

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

11.1.3.228 sparse_delete() [12/12]

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

11.1.3.229 sparse_print() [3/3]

```
template<class Field>
void sparse_print (
            const Sparse< Field, SparseMatrix_t::SELL > & A)  [inline]
```

11.1.3.230 sparse_init() [15/16]

```
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)  [inline]
```

11.1.3.231 sparse_init() [16/16]

```
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)  [inline]
```

11.1.3.232 computeDeviation()

```
template<class It>
double computeDeviation (
            It begin,
            It end)
```

11.1.3.233 getStat()

```
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)
```

11.1.3.234 ftranspose()

```
template<typename Field, typename Simd = Simd<typename Field::Element>, size_t bs = FFLAS_↩
TRANSPOSE_BLOCKSIZE, typename std::enable_if< Simd::template is_same_element< Field >::value
>::type * = nullptr, typename std::enable_if< bs > = 1 && bs % Simd::vect_size == 0, ::type *
= nullptr>
Field::Element_ptr ftranspose (
            const Field & F,
            const size_t m,
            const size_t n,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::Element_ptr B,
            const size_t ldb) [inline]
```

11.1.3.235 maxCardinality()

```
template<class Field, class enable = void>
Field::Residu_t maxCardinality () [inline]
```

11.1.3.236 maxCardinality< Givaro::Modular< int64_t > >()

```
template<>
uint64_t maxCardinality< Givaro::Modular< int64_t > > () [inline]
```

11.1.3.237 maxCardinality< Givaro::Modular< int32_t > >()

```
template<>
uint32_t maxCardinality< Givaro::Modular< int32_t > > () [inline]
```

11.1.3.238 minCardinality()

```
template<class Field>
Field::Residu_t minCardinality ()  [inline]
```

11.1.3.239 fflas_delete() [1/4]

```
template<>
void fflas_delete (
            FFPACK::rns_double_elt_ptr A)  [inline]
```

11.1.3.240 fflas_delete() [2/4]

```
template<>
void fflas_delete (
            FFPACK::rns_double_elt_cstptr A)  [inline]
```

11.1.3.241 fflas_new() [1/7]

```
template<>
FFPACK::rns_double_elt_ptr fflas_new (
            const FFPACK::RNSIntegerMod< FFPACK::rns_double > & F,
            const size_t m,
            const Alignment align)  [inline]
```

11.1.3.242 fflas_new() [2/7]

```
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)  [inline]
```

11.1.3.243 finit_rns() [1/2]

```
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)
```

11.1.3.244 finit_trans_rns()

```
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)
```

11.1.3.245 fconvert_rns() [1/2]

```
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)
```

11.1.3.246 fconvert_trans_rns()

```
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)
```

11.1.3.247 fflas_new() [3/7]

```
template<>
FFPACK::rns_double_elt_ptr fflas_new (
            const FFPACK::RNSInteger< FFPACK::rns_double > & F,
            const size_t m,
            const Alignment align)  [inline]
```

11.1.3.248 fflas_new() [4/7]

```
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)  [inline]
```

11.1.3.249 finit_rns() [2/2]

```
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)
```

11.1.3.250 fconvert_rns() [2/2]

```
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)
```

11.1.3.251 freduce() [8/11]

```
template INST_OR_DECL void freduce (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t n,
            FFLAS_ELT * X,
            const size_t incX)
```

freduce $x \leftarrow xmodF$.

Parameters

F	field
n	size of the vectors
X	vector in $\mathbb{F}$
incX	stride of X

Bug use cblas_(d)scal when possible

11.1.3.252 freduce() [9/11]

```
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 $x \leftarrow ymodF$.

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

11.1.3.253 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 $x \leftarrow ymodF$.

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

11.1.3.254 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 $x \leftarrow ymodF$.

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

11.1.3.255 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

11.1.3.256 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

11.1.3.257 fzero() [4/5]

```
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

11.1.3.258 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

11.1.3.259 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

11.1.3.260 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

11.1.3.261 fscalin() [9/10]

```
template INST_OR_DECL void fscalin (
            const FFLAS_FIELD< FFLAS_ELT > & F,
            const size_t n,
            const FFLAS_ELT alpha,
            FFLAS_ELT * X,
            const size_t incX)
```

fscalin $x \leftarrow \alpha \cdot x$.

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.

11.1.3.262 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)
```

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

Parameters

	F	field
	n	size of the vectors
	alpha	scalar
in	*X*	vector in F
	incX	stride of X
out	*Y*	vector in F
	incY	stride of Y

Bug use cblas_(d)scal when possible

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

11.1.3.263 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)
```

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

11.1.3.264 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

11.1.3.265 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

11.1.3.266 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)
```

11.1.3.267 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)
```

11.1.3.268 faddin() [4/5]

```
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)
```

11.1.3.269 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)
```

11.1.3.270 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

11.1.3.271 fzero() [5/5]

```
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

11.1.3.272 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

11.1.3.273 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

11.1.3.274 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

11.1.3.275 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

11.1.3.276 freduce() [10/11]

```
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 $A \leftarrow A mod F$.

Parameters

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

11.1.3.277 freduce() [11/11]

```
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 $A \leftarrow B mod F$.

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

11.1.3.278 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)
```

finit $A \leftarrow BmodF$.

Parameters

F	field
m	number of rows
n	number of cols
A	matrix in F
lda	stride of A
B	matrix in F
ldb	stride of B

11.1.3.279 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)
```

fnegin $A \leftarrow -A$.

Parameters

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

11.1.3.280 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)
```

fneg $A \leftarrow -B$.

Parameters

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

11.1.3.281 fscalin() [10/10]

```
template INST_OR_DECL void fscalin (
            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)
```

fscalin $A \leftarrow a \cdot A$.

Parameters

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

11.1.3.282 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)
```

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

11.1.3.283 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)
```

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

11.1.3.284 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)
```

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

11.1.3.285 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 MxN
lda	leading dimension of A
B	dense matrix of size MxN
ldb	leading dimension of B
C	dense matrix of size MxN
ldc	leading dimension of C

11.1.3.286 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

F	field
M	rows
N	cols
A	dense matrix of size MxN
lda	leading dimension of A
B	dense matrix of size MxN
ldb	leading dimension of B
C	dense matrix of size MxN
ldc	leading dimension of C

11.1.3.287 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

11.1.3.288 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

F	field
M	rows
N	cols
A	dense matrix of size MxN
lda	leading dimension of A
alpha	some scalar
B	dense matrix of size MxN
ldb	leading dimension of B
C	dense matrix of size MxN
ldc	leading dimension of C

11.1.3.289 faddin() [5/5]

```
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

11.1.3.290 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 $Y \leftarrow \alpha \mathrm{op}(A) X + \beta Y$.

Parameters

	F	field
	TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
	M	rows
	N	cols
	alpha	scalar
	A	dense matrix of size `MxN`
	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`

11.1.3.291 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

	F	field
	M	rows
	N	cols
	alpha	scalar
in,out	*A*	dense matrix of size MxN 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

11.1.3.292 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: TRiangular System solve with Vector Computes $X \leftarrow \mathrm{op}(A^{-1})X$

Parameters

F	field
X	vector of size N on a field F
incX	stride of X
A	a matrix of leading dimension lda and size N
lda	leading dimension of A

Parameters

N	number of rows or columns of `A` according to `TransA`
TransA	if `TransA==FflasTrans` then $\mathrm{op}(A) = A^t$.
Diag	if `Diag==FflasUnit` then `A` is unit.
Uplo	if `Uplo==FflasUpper` then `A` is upper triangular

11.1.3.293 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**iangular **S**ystem solve with **M**atrix.

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

Parameters

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

Bug α must be non zero.

11.1.3.294 ftrmm() [3/3]

```
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.

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

Parameters

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

11.1.3.295 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 **GE**neral **M**atrix **M**ultiply.

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

Parameters

F	field.
ta	if ta==FflasTrans then $\mathrm{op}(A) = A^t$, else $\mathrm{op}(A) = A$,
tb	same for matrix B
m	see A
n	see B
k	see A
alpha	scalar
beta	scalar
A	$\mathrm{op}(A)$ is $m \times k$
B	$\mathrm{op}(B)$ 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
w	recursive levels of Winograd's algorithm are used. No argument (or -1) does auto computation of w.

Warning

α *must* be invertible

11.1.3.296 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)
```

11.1.3.297 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)
```

11.1.3.298 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)
```

11.1.3.299 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 $C \leftarrow \alpha \mathrm{op}(A)\mathrm{op}(A) + \beta C$ over a FFLAS_FIELD <FFLAS_ELT> F Avoid the conversion of B

Parameters

ta	if ta==FflasTrans, $\mathrm{op}(A) = A^T$.
F	field
n	size of A
alpha	scalar

Parameters

beta	scalar
A	dense matrix of size nxn
lda	leading dimension of A
C	dense matrix of size nxn
ldc	leading dimension of C

11.1.3.300 BlockCuts() [1/2]

```
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)  [inline]
```

11.1.3.301 BlockCuts< CuttingStrategy::Single, StrategyParameter::Threads >()

```
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)  [inline]
```

11.1.3.302 BlockCuts< CuttingStrategy::Row, StrategyParameter::Fixed >()

```
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)  [inline]
```

11.1.3.303 BlockCuts< CuttingStrategy::Row, StrategyParameter::Grain >()

```
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)  [inline]
```

11.1.3.304 BlockCuts< CuttingStrategy::Block, StrategyParameter::Grain >()

```
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)  [inline]
```

11.1.3.305 BlockCuts< CuttingStrategy::Column, StrategyParameter::Fixed >()

```
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)  [inline]
```

11.1.3.306 BlockCuts< CuttingStrategy::Column, StrategyParameter::Grain >()

```
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)  [inline]
```

11.1.3.307 BlockCuts< CuttingStrategy::Block, StrategyParameter::Fixed >()

```
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)  [inline]
```

11.1.3.308 BlockCuts< CuttingStrategy::Row, StrategyParameter::Threads >()

```
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)  [inline]
```

11.1.3.309 BlockCuts< CuttingStrategy::Column, StrategyParameter::Threads >()

```
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)  [inline]
```

11.1.3.310 BlockCuts< CuttingStrategy::Block, StrategyParameter::Threads >()

```
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)  [inline]
```

11.1.3.311 BlockCuts() [2/2]

```
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)  [inline]
```

11.1.3.312 pfzero()

```
template<class Field>
void pfzero (
            const Field & F,
            size_t m,
            size_t n,
            typename Field::Element_ptr C,
            size_t BS = 0)
```

11.1.3.313 pfrand()

```
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)
```

11.1.3.314 fdot() [11/11]

```
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)  [inline]
```

11.1.3.315 pfgemm() [2/7]

```
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)
```

11.1.3.316 pfgemm() [3/7]

```
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)
```

11.1.3.317 pfgemm() [4/7]

```
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)
```

11.1.3.318 pfgemm() [5/7]

```
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)
```

11.1.3.319 pfgemm() [6/7]

```
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)
```

11.1.3.320 pfgemm() [7/7]

```
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)
```

11.1.3.321 fgemv() [18/19]

```
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)
```

11.1.3.322 fgemv() [19/19]

```
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)
```

11.1.3.323 parseArguments()

```
void parseArguments (
            int argc,
            char ** argv,
            Argument * args,
            bool printDefaults = true)
```

11.1.3.324 getArgumentValue()

```
char * getArgumentValue (
            int argc,
            char ** argv,
            int i)
```

Get the value of an argument and avoid core dump when no value was given after an argument.

Parameters

argv	argument value list
i	argument index

Returns

char∗ argument value

11.1.3.325 writeCommandString()

```
std::ostream & writeCommandString (
            std::ostream & os,
            Argument * args,
            const char * programName = nullptr)
```

writes the values of all arguments, preceded by the programName

11.1.3.326 WriteMatrix() [1/2]

```
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) [inline]
```

WriteMatrix: write a matrix to an output stream.

Parameters

c	output stream
F	base field
m	row dimension
n	column dimension
A	matrix
format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
column_major	whether the matrix is stored in column or row major (row by default)

11.1.3.327 preamble()

```
void preamble (
            std::ifstream & ifs,
            FFLAS_FORMAT & format) [inline]
```

11.1.3.328 ReadMatrix() [1/2]

```
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.

Parameters

	ifs	input stream
	F	base field
out	*m*	row dimension
out	*n*	column dimension
out	*A*	output matrix
	format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

11.1.3.329 ReadMatrix() [2/2]

```
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)  [inline]
```

ReadMatrix: read a matrix from a file.

Parameters

	matrix_file	filename
	F	base field
out	*m*	row dimension
out	*n*	column dimension
out	*A*	output matrix
	format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)

11.1.3.330 WriteMatrix() [2/2]

```
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.

Parameters

matrix_file	file name
F	base field
m	row dimension
n	column dimension
A	matrix
format	input format (FflasAuto, FflasDense, FflasSMS, FflasBinary)
column_major	whether the matrix is stored in column or row major (row by default)

11.1.3.331 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

c	output stream
P	permutation
N	size of the permutation

11.1.3.332 alignable()

```
template<class Element>
bool alignable ()  [inline]
```

11.1.3.333 alignable< Givaro::Integer * >()

```
template<>
bool alignable< Givaro::Integer * > ()  [inline]
```

11.1.3.334 fflas_new() [5/7]

```
template<class Field>
Field::Element_ptr fflas_new (
            const Field & F,
            const size_t m,
            const Alignment align = Alignment::DEFAULT)  [inline]
```

11.1.3.335 fflas_new() [6/7]

```
template<class Field>
Field::Element_ptr fflas_new (
            const Field & F,
            const size_t m,
            const size_t n,
            const Alignment align = Alignment::DEFAULT)  [inline]
```

11.1.3.336 fflas_new() [7/7]

```
template<class Element>
Element * fflas_new (
            const size_t m,
            const Alignment align = Alignment::DEFAULT)  [inline]
```

11.1.3.337 fflas_delete() [3/4]

```
template<class Element_ptr>
void fflas_delete (
            Element_ptr A)  [inline]
```

11.1.3.338 fflas_delete() [4/4]

```
template<class Ptr, class ... Args>
void fflas_delete (
            Ptr p,
            Args ... args) [inline]
```

11.1.3.339 prefetch()

```
void prefetch (
            const int64_t * ) [inline]
```

11.1.3.340 getTLBSize()

```
void getTLBSize (
            int & tlb) [inline]
```

11.1.3.341 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

11.1.3.342 queryL1CacheSize()

```
int queryL1CacheSize () [inline]
```

Returns

the size in Bytes of the L1 data cache

11.1.3.343 queryTopLevelCacheSize()

```
int queryTopLevelCacheSize () [inline]
```

Returns

the size in Bytes of the L2 or L3 cache if this later is present

11.1.3.344 getSeed()

```
uint64_t getSeed ()
```

11.2 FFLAS::_ftranspose_impl Namespace Reference

Functions

- template<size_t bs, typename Field, typename BTSimd>
  void not_inplace (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)
- template<size_t bs, typename Field, typename BTSimd>
  void square_inplace (const Field &F, const BTSimd &BTS, const size_t m, typename Field::Element_ptr A, const size_t lda)
- template<size_t bs, typename Field, typename BTSimd>
  void nonsquare_inplace_v1 (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::Element_ptr A)
- template<size_t bs, typename Field, typename BTSimd>
  void nonsquare_inplace_v2 (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::Element_ptr A)

11.2.1 Function Documentation

11.2.1.1 not_inplace()

```
template<size_t bs, typename Field, typename BTSimd>
void not_inplace (
            const Field & F,
            const BTSimd & BTS,
            const size_t m,
            const size_t n,
            typename Field::ConstElement_ptr A,
            const size_t lda,
            typename Field::Element_ptr B,
            const size_t ldb)
```

11.2.1.2 square_inplace()

```
template<size_t bs, typename Field, typename BTSimd>
void square_inplace (
            const Field & F,
            const BTSimd & BTS,
            const size_t m,
            typename Field::Element_ptr A,
            const size_t lda)
```

11.2.1.3 nonsquare_inplace_v1()

```
template<size_t bs, typename Field, typename BTSimd>
void nonsquare_inplace_v1 (
            const Field & F,
            const BTSimd & BTS,
            const size_t m,
            const size_t n,
            typename Field::Element_ptr A)
```

11.2.1.4 nonsquare_inplace_v2()

```
template<size_t bs, typename Field, typename BTSimd>
void nonsquare_inplace_v2 (
            const Field & F,
            const BTSimd & BTS,
            const size_t m,
            const size_t n,
            typename Field::Element_ptr A)
```

11.3 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, FieldTrait > &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 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)

- 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, 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, 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, MMHelperAlgo::Winograd, FieldTrait > &WH)

11.3.1 Function Documentation

11.3.1.1 Bini()

```
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)  [inline]
```

11.3.1.2 WinoPar()

```
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)  [inline]
```

11.3.1.3 Winograd()

```
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, FieldTrait > & WH)  [inline]
```

11.3.1.4 WinogradAcc_3_23()

```
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)  [inline]
```

11.3.1.5 WinogradAcc_3_21()

```
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)  [inline]
```

11.3.1.6 WinogradAcc_2_24()

```
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)  [inline]
```

11.3.1.7 WinogradAcc_2_27()

```
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)  [inline]
```

11.3.1.8 WinogradAcc_LR()

```
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, FieldTrait > & WH)  [inline]
```

11.3.1.9 WinogradAcc_R_S()

```
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, MMHelperAlgo::Winograd, FieldTrait > & WH) [inline]
```

11.3.1.10 WinogradAcc_L_S()

```
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) [inline]
```

11.3.1.11 Winograd_LR_S()

```
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) [inline]
```

11.3.1.12 Winograd_L_S()

```
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) [inline]
```

11.3.1.13 Winograd_R_S()

```
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) [inline]
```

11.4 FFLAS::csr_hyb_details Namespace Reference

Data Structures

- struct Coo
- struct Info

11.5 FFLAS::CuttingStrategy Namespace Reference

Data Structures

- struct Block
- struct Column
- struct Recursive
- struct Row
- struct Single

Typedefs

- typedef Row RNSModulus

11.5.1 Typedef Documentation

11.5.1.1 RNSModulus

```
typedef Row RNSModulus
```

11.6 FFLAS::details Namespace Reference

Functions

- template<class Field, bool ADD>
  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)
- template<class Field, bool ADD>
  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)
- template<class Field, bool ADD>
  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)
- template<class Field, bool ADD>
  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)
- template<class Field, bool ADD>
  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)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type faxpy (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 faxpy (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>
  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, typename 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)

11.6.1 Function Documentation

11.6.1.1 fadd() [1/5]

```
template<class Field, bool ADD>
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 )
```

11.6.1.2 fadd() [2/5]

```
template<class Field, bool ADD>
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 )
```

11.6.1.3 fadd() [3/5]

```
template<class Field, bool ADD>
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 )
```

11.6.1.4 fadd() [4/5]

```
template<class Field, bool ADD>
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]
```

11.6.1.5 fadd() [5/5]

```
template<class Field, bool ADD>
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]
```

11.6.1.6 faxpy() [1/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type faxpy
(
            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]
```

11.6.1.7 faxpy() [2/2]

```
template<class Field, class FC>
void faxpy (
            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]
```

11.6.1.8 freduce() [1/4]

```
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 )  [inline]
```

11.6.1.9 freduce() [2/4]

```
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 )  [inline]
```

11.6.1.10 freduce() [3/4]

```
template<class Field, class FC>
void freduce (
            const Field & F,
            const size_t m,
            typename Field::Element_ptr A,
            const size_t incX,
            FC )  [inline]
```

11.6.1.11 freduce() [4/4]

```
template<class Field, class FC>
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]
```

11.6.1.12 fscalin() [1/2]

```
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 )  [inline]
```

11.6.1.13 fscal() [1/2]

```
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 )  [inline]
```

11.6.1.14 fscalin() [2/2]

```
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 )  [inline]
```

11.6.1.15 fscal() [2/2]

```
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 )  [inline]
```

11.6.1.16 igebb44()

```
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)  [inline]
```

11.6.1.17 igebb24()

```
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)  [inline]
```

11.6.1.18 igebb14()

```
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)  [inline]
```

11.6.1.19 igebb41()

```
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)  [inline]
```

bug ,B_0 dans VEC_MADD_32 ?

11.6.1.20 igebb21()

```
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)  [inline]
```

11.6.1.21 igebb11()

```
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)  [inline]
```

11.6.1.22 igebp()

```
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)
```

11.6.1.23 pack_lhs()

```
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)
```

Bug this is fassign

Bug this is fassign

11.6.1.24 pack_rhs()

```
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)
```

Bug this is fassign

Bug this is fassign

11.6.1.25 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)
```

11.6.1.26 BlockingFactor()

```
void BlockingFactor (
            size_t & m,
            size_t & n,
            size_t & k)  [inline]
```

11.7 FFLAS::details_spmv Namespace Reference

Data Structures

- struct Coo

11.8 FFLAS::ElementCategories Namespace Reference

Data Structures

- struct ArbitraryPrecIntTag

    Arbitrary precision integers: GMP.
- struct FixedPrecIntTag

    Fixed precision integers above machine precision: Givaro::recInt.
- 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.

11.9 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.

11.9.1 Detailed Description

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

11.10 FFLAS::MMHelperAlgo Namespace Reference

Data Structures

- struct Auto
- struct Bini
- struct Classic
- struct DivideAndConquer
- struct Winograd
- struct WinogradPar

11.11 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.

11.11.1 Detailed Description

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

its field

11.12 FFLAS::ParSeqHelper Namespace Reference

ParSeqHelper for both fgemm and ftrsm.

Data Structures

- struct Compose
- struct Parallel
- struct Sequential

11.12.1 Detailed Description

ParSeqHelper for both fgemm and ftrsm.

ParSeqHelper for both fgemm and ftrsm

11.13 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 double Computes nmax s.t.
- 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 NewField, class Field, class FieldMode>
  Field::Element_ptr fsyrk_convert (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, MMHelper< Field, MMHelperAlgo::Classic, FieldMode > &H)
- template<class Field, class AlgoT, class ParSeqTrait>
  void ScalAndReduce (const Field &F, const FFLAS_UPLO UpLo, 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, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, const Element &x, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreAxpyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element

&Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)

- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreAxpyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class DFE>
  size_t min_types (const DFE &k)
- template<> size_t min_types (const RecInt::rint< 6 > &k)
- template<> size_t min_types (const RecInt::rint< 7 > &k)
- template<> size_t min_types (const RecInt::rint< 8 > &k)
- template<> size_t min_types (const RecInt::rint< 9 > &k)
- template<> size_t min_types (const RecInt::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 (RecInt::rint< K > x)
- template<> bool unfit (RecInt::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, typename Field::ConstElement_ptr const E, const size_t lde, const size_t m, const size_t n)
- template<class Field>
  void MatF2MatFl_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)

11.13.1 Function Documentation

11.13.1.1 computeFactorClassic() [1/3]

```
template<class Field>
double computeFactorClassic (
            const Field & F)  [inline]
```

11.13.1.2 computeFactorClassic() [2/3]

```
template<>
double computeFactorClassic (
            const Givaro::ModularBalanced< double > & F)  [inline]
```

11.13.1.3 computeFactorClassic() [3/3]

```
template<>
double computeFactorClassic (
            const Givaro::ModularBalanced< float > & F) [inline]
```

11.13.1.4 DotProdBoundClassic()

```
template<class Field>
size_t DotProdBoundClassic (
            const Field & F,
            const typename Field::Element & beta) [inline]
```

11.13.1.5 TRSMBound() [1/3]

```
template<class Field>
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

11.13.1.6 TRSMBound() [2/3]

```
template<class Element>
size_t TRSMBound (
            const Givaro::Modular< Element > & F) [inline]
```

Specialization for positive modular representation over double Computes nmax s.t.

(p-1)/2∗(p^{nmax-1} + (p-2)^{nmax-1}) < 2^53 See [Dumas Giorgi Pernet 06, arXiv:cs/0601133] Specialization for positive modular representation over float. Computes nmax s.t. (p-1)/2∗(p^{nmax-1} + (p-2)^{nmax-1}) < 2^24 @pbi See [Dumas Giorgi Pernet 06, arXiv:cs/0601133]

11.13.1.7 TRSMBound() [3/3]

```
template<class Element>
size_t TRSMBound (
            const Givaro::ModularBalanced< Element > & F) [inline]
```

Specialization for balanced modular representation over double.

Computes nmax s.t. (p-1)/2∗(((p+1)/2)^{nmax-1}) < 2^53

Bibliography
- Dumas Giorgi Pernet 06, arXiv:cs/0601133

11.13.1.8 fgemm_convert()

```
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)  [inline]
```

11.13.1.9 NeedPreAddReduction() [1/2]

```
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)  [inline]
```

11.13.1.10 NeedPreAddReduction() [2/2]

```
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)  [inline]
```

11.13.1.11 NeedPreSubReduction() [1/2]

```
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)  [inline]
```

11.13.1.12 NeedPreSubReduction() [2/2]

```
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)  [inline]
```

11.13.1.13 NeedDoublePreAddReduction() [1/2]

```
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)  [inline]
```

11.13.1.14 NeedDoublePreAddReduction() [2/2]

```
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)  [inline]
```

11.13.1.15 ScalAndReduce() [1/3]

```
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)  [inline]
```

11.13.1.16 ScalAndReduce() [2/3]

```
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)  [inline]
```

11.13.1.17 fsquareCommon()

```
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)  [inline]
```

11.13.1.18 WinogradThreshold() [1/4]

```
template<class Field>
int WinogradThreshold (
            const Field & F)  [inline]
```

Computes the number of recursive levels to perform.

Parameters

m	the common dimension in the product AxB

11.13.1.19 WinogradThreshold() [2/4]

```
template<>
int WinogradThreshold (
            const Givaro::Modular< float > & F)  [inline]
```

11.13.1.20 WinogradThreshold() [3/4]

```
template<>
int WinogradThreshold (
            const Givaro::ModularBalanced< double > & F)  [inline]
```

11.13.1.21 WinogradThreshold() [4/4]

```
template<>
int WinogradThreshold (
            const Givaro::ModularBalanced< float > & F)  [inline]
```

11.13.1.22 WinogradSteps()

```
template<class Field>
int WinogradSteps (
            const Field & F,
            const size_t & m)  [inline]
```

Computes the number of recursive levels to perform.

Parameters

m	the common dimension in the product AxB

11.13.1.23 DynamicPeeling()

```
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 >::Delayed↩
Field::Element Cmin,
            const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed↩
Field::Element Cmax)  [inline]
```

11.13.1.24 DynamicPeeling2()

```
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 >::Delayed←
Field::Element Cmin,
            const typename MMHelper< Field, MMHelperAlgo::Winograd, FieldMode >::Delayed←
Field::Element Cmax) [inline]
```

11.13.1.25 WinogradCalc()

```
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) [inline]
```

11.13.1.26 fgemv_convert()

```
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)  [inline]
```

11.13.1.27 fger_convert()

```
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)  [inline]
```

11.13.1.28 fsyrk_convert()

```
template<class NewField, class Field, class FieldMode>
Field::Element_ptr fsyrk_convert (
            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,
            MMHelper< Field, MMHelperAlgo::Classic, FieldMode > & H)  [inline]
```

11.13.1.29 ScalAndReduce() [3/3]

```
template<class Field, class AlgoT, class ParSeqTrait>
void ScalAndReduce (
            const Field & F,
            const FFLAS_UPLO UpLo,
            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]
```

11.13.1.30 NeedPreScalReduction() [1/2]

```
template<class Field, class Element, class AlgoT, class ParSeqTrait>
bool NeedPreScalReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            const Element & x,
            MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH)  [inline]
```

11.13.1.31 NeedPreScalReduction() [2/2]

```
template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
bool NeedPreScalReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            const Element & x,
            MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH)  [inline]
```

11.13.1.32 NeedPreAxpyReduction() [1/2]

```
template<class Field, class Element, class AlgoT, class ParSeqTrait>
bool NeedPreAxpyReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            const Element & x,
            MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > & WH)  [inline]
```

11.13.1.33 NeedPreAxpyReduction() [2/2]

```
template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
bool NeedPreAxpyReduction (
            Element & Outmin,
            Element & Outmax,
            Element & Op1min,
            Element & Op1max,
            Element & Op2min,
            Element & Op2max,
            const Element & x,
            MMHelper< Field, AlgoT, ModeT, ParSeqTrait > & WH)  [inline]
```

11.13.1.34 min_types() [1/7]

```
template<class DFE>
size_t min_types (
            const DFE & k)  [inline]
```

11.13.1.35 min_types() [2/7]

```
template<>
size_t min_types (
            const RecInt::rint< 6 > & k) [inline]
```

11.13.1.36 min_types() [3/7]

```
template<>
size_t min_types (
            const RecInt::rint< 7 > & k) [inline]
```

11.13.1.37 min_types() [4/7]

```
template<>
size_t min_types (
            const RecInt::rint< 8 > & k) [inline]
```

11.13.1.38 min_types() [5/7]

```
template<>
size_t min_types (
            const RecInt::rint< 9 > & k) [inline]
```

11.13.1.39 min_types() [6/7]

```
template<>
size_t min_types (
            const RecInt::rint< 10 > & k) [inline]
```

11.13.1.40 min_types() [7/7]

```
template<>
size_t min_types (
            const Givaro::Integer & k) [inline]
```

11.13.1.41 unfit() [1/4]

```
template<class T>
bool unfit (
            T x) [inline]
```

11.13.1.42 unfit() [2/4]

```
template<>
bool unfit (
            int64_t x) [inline]
```

11.13.1.43 unfit() [3/4]

```
template<size_t K>
bool unfit (
            RecInt::rint< K > x)  [inline]
```

11.13.1.44 unfit() [4/4]

```
template<>
bool unfit (
            RecInt::rint< 6 > x)  [inline]
```

11.13.1.45 igemm_colmajor() [1/2]

```
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)
```

11.13.1.46 igemm_colmajor() [2/2]

```
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)
```

11.13.1.47 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.

11.13.1.48 MatF2MatD_Triangular()

```
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)
```

11.13.1.49 MatF2MatFl_Triangular()

```
template<class Field>
void MatF2MatFl_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)

11.14 FFLAS::sell_details Namespace Reference

Data Structures

- struct Coo
- struct Info

11.15 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::MachineFloa >::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::MachineFloatT >::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::MachineFloa >::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::MachineFloa
>::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)

&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)

11.15.1 Function Documentation

11.15.1.1 init_y() [1/2]

```
template<class Field>
void init_y (
            const Field & F,
            const size_t m,
            const typename Field::Element b,
            typename Field::Element_ptr y)  [inline]
```

11.15.1.2 init_y() [2/2]

```
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)  [inline]
```

11.15.1.3 fspmv_dispatch() [1/2]

```
template<class Field, class SM, class FC, class MZO>
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Ma
>::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]
```

11.15.1.4 fspmv_dispatch() [2/2]

```
template<class Field, class SM, class FC, class MZO>
std::enable_if< std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Mac
>::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]
```

11.15.1.5 fspmv() [1/12]

```
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 )  [inline]
```

11.15.1.6 fspmv() [2/12]

```
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 )  [inline]
```

11.15.1.7 fspmv() [3/12]

```
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 )  [inline]
```

11.15.1.8 fspmv() [4/12]

```
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 )  [inline]
```

11.15.1.9 fspmv() [5/12]

```
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 )  [inline]
```

11.15.1.10 fspmv() [6/12]

```
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 ) [inline]
```

11.15.1.11 fspmv() [7/12]

```
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 ) [inline]
```

11.15.1.12 fspmv() [8/12]

```
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 ) [inline]
```

11.15.1.13 fspmv() [9/12]

```
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 ) [inline]
```

11.15.1.14 fspmm_dispatch() [1/2]

```
template<class Field, class SM, class FCat, class MZO>
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Ma
>::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]
```

11.15.1.15 fspmm_dispatch() [2/2]

```
template<class Field, class SM, class FCat, class MZO>
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]
```

11.15.1.16 fspmm() [1/9]

```
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 )  [inline]
```

11.15.1.17 fspmm() [2/9]

```
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 )  [inline]
```

11.15.1.18 fspmm() [3/9]

```
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 )  [inline]
```

11.15.1.19 fspmm() [4/9]

```
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 )  [inline]
```

11.15.1.20 fspmm() [5/9]

```
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 )  [inline]
```

11.15.1.21 fspmm() [6/9]

```
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 )  [inline]
```

11.15.1.22 fspmm() [7/9]

```
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 )  [inline]
```

11.15.1.23 fspmm() [8/9]

```
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 )  [inline]
```

11.15.1.24 fspmm() [9/9]

```
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 )  [inline]
```

11.15.1.25 pfspmm_dispatch() [1/2]

```
template<class Field, class SM, class FCat, class MZO>
std::enable_if<!(std::is_same< typenameElementTraits< typenameField::Element >::value, ElementCategories::Ma
>::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]
```

11.15.1.26 pfspmm_dispatch() [2/2]

```
template<class Field, class SM, class FCat, class MZO>
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]
```

11.15.1.27 pfspmm() [1/9]

```
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 )  [inline]
```

11.15.1.28 pfspmm() [2/9]

```
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 )  [inline]
```

11.15.1.29 pfspmm() [3/9]

```
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 )  [inline]
```

11.15.1.30 pfspmm() [4/9]

```
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 )  [inline]
```

11.15.1.31 pfspmm() [5/9]

```
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 )  [inline]
```

11.15.1.32 pfspmm() [6/9]

```
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 )  [inline]
```

11.15.1.33 pfspmm() [7/9]

```
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 )  [inline]
```

11.15.1.34 pfspmm() [8/9]

```
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 )  [inline]
```

11.15.1.35 pfspmm() [9/9]

```
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 )  [inline]
```

11.15.1.36 pfspmv() [1/6]

```
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 ) [inline]
```

11.15.1.37 pfspmv() [2/6]

```
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 ) [inline]
```

11.15.1.38 pfspmv() [3/6]

```
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 ) [inline]
```

11.15.1.39 pfspmv() [4/6]

```
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 ) [inline]
```

11.15.1.40 pfspmv() [5/6]

```
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 ) [inline]
```

11.15.1.41 pfspmv() [6/6]

```
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 ) [inline]
```

11.15.1.42 fspmv() [10/12]

```
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 ) [inline]
```

11.15.1.43 fspmv() [11/12]

```
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 ) [inline]
```

11.15.1.44 fspmv() [12/12]

```
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 ) [inline]
```

11.16 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 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)
- 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, 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)
- 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 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)
- 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)
- 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)
- 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)
- 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)
- 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::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, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)
- 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)
- 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)
- 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)
- 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)
- 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)

11.16.1 Function Documentation

11.16.1.1 fspmm() [1/15]

```
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 )  [inline]
```

11.16.1.2 fspmm() [2/15]

```
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 )  [inline]
```

11.16.1.3 fspmm() [3/15]

```
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)  [inline]
```

11.16.1.4 fspmm_simd_aligned() [1/2]

```
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)  [inline]
```

11.16.1.5 fspmm_simd_unaligned() [1/2]

```
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)  [inline]
```

11.16.1.6 fspmm_one() [1/4]

```
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 )  [inline]
```

11.16.1.7 fspmm_mone() [1/4]

```
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 )  [inline]
```

11.16.1.8 fspmm_one_simd_aligned() [1/3]

```
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 )  [inline]
```

11.16.1.9 fspmm_one_simd_unaligned() [1/3]

```
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 )  [inline]
```

11.16.1.10 fspmm_mone_simd_aligned() [1/3]

```
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 )  [inline]
```

11.16.1.11 fspmm_mone_simd_unaligned() [1/3]

```
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 ) [inline]
```

11.16.1.12 fspmv() [1/21]

```
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 ) [inline]
```

11.16.1.13 fspmv() [2/21]

```
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 ) [inline]
```

11.16.1.14 fspmv() [3/21]

```
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) [inline]
```

11.16.1.15 fspmv_one() [1/10]

```
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 ) [inline]
```

11.16.1.16 fspmv_mone() [1/10]

```
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 ) [inline]
```

11.16.1.17 fspmv_one() [2/10]

```
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 ) [inline]
```

11.16.1.18 fspmv_mone() [2/10]

```
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 ) [inline]
```

11.16.1.19 pfspmm() [1/18]

```
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 ) [inline]
```

11.16.1.20 pfspmm() [2/18]

```
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 ) [inline]
```

11.16.1.21 pfspmm() [3/18]

```
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)  [inline]
```

11.16.1.22 pfspmm_one() [1/2]

```
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 )  [inline]
```

11.16.1.23 pfspmm_mone() [1/2]

```
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 )  [inline]
```

11.16.1.24 pfspmm_one() [2/2]

```
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 )  [inline]
```

11.16.1.25 pfspmm_mone() [2/2]

```
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 ) [inline]
```

11.16.1.26 pfspmv() [1/18]

```
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 ) [inline]
```

11.16.1.27 pfspmv_task()

```
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 ) [inline]
```

11.16.1.28 pfspmv() [2/18]

```
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 ) [inline]
```

11.16.1.29 pfspmv() [3/18]

```
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) [inline]
```

11.16.1.30 pfspmv_one() [1/8]

```
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 )  [inline]
```

11.16.1.31 pfspmv_mone() [1/8]

```
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 )  [inline]
```

11.16.1.32 pfspmv_one() [2/8]

```
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 )  [inline]
```

11.16.1.33 pfspmv_mone() [2/8]

```
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 )  [inline]
```

11.16.1.34 fspmm() [4/15]

```
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 )  [inline]
```

11.16.1.35 fspmm() [5/15]

```
template<class Field>
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]
```

11.16.1.36 fspmm_simd_aligned() [2/2]

```
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 )  [inline]
```

11.16.1.37 fspmm_simd_unaligned() [2/2]

```
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 )  [inline]
```

11.16.1.38 fspmm() [6/15]

```
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)  [inline]
```

11.16.1.39 fspmm_one() [2/4]

```
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 ) [inline]
```

11.16.1.40 fspmm_mone() [2/4]

```
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 ) [inline]
```

11.16.1.41 fspmm_one_simd_aligned() [2/3]

```
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 ) [inline]
```

11.16.1.42 fspmm_one_simd_unaligned() [2/3]

```
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 ) [inline]
```

11.16.1.43 fspmm_mone_simd_aligned() [2/3]

```
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 )  [inline]
```

11.16.1.44 fspmm_mone_simd_unaligned() [2/3]

```
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 )  [inline]
```

11.16.1.45 fspmv() [4/21]

```
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 )  [inline]
```

11.16.1.46 fspmv() [5/21]

```
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 )  [inline]
```

11.16.1.47 fspmv() [6/21]

```
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)  [inline]
```

11.16.1.48 fspmv_one() [3/10]

```
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 ) [inline]
```

11.16.1.49 fspmv_mone() [3/10]

```
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 ) [inline]
```

11.16.1.50 fspmv_one() [4/10]

```
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 ) [inline]
```

11.16.1.51 fspmv_mone() [4/10]

```
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 ) [inline]
```

11.16.1.52 pfspmm() [4/18]

```
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 ) [inline]
```

11.16.1.53 pfspmm() [5/18]

```
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 )  [inline]
```

11.16.1.54 pfspmm() [6/18]

```
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 )  [inline]
```

11.16.1.55 pfspmm() [7/18]

```
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 )  [inline]
```

11.16.1.56 pfspmm() [8/18]

```
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)  [inline]
```

11.16.1.57 pfspmm() [9/18]

```
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)  [inline]
```

11.16.1.58 pfspmv() [4/18]

```
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 )  [inline]
```

11.16.1.59 pfspmv() [5/18]

```
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 )  [inline]
```

11.16.1.60 pfspmv() [6/18]

```
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)  [inline]
```

11.16.1.61 fspmm() [7/15]

```
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 )  [inline]
```

11.16.1.62 fspmm() [8/15]

```
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 )  [inline]
```

11.16.1.63 fspmm() [9/15]

```
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)  [inline]
```

11.16.1.64 fspmv() [7/21]

```
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 )  [inline]
```

11.16.1.65 fspmv() [8/21]

```
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 )  [inline]
```

11.16.1.66 fspmv() [9/21]

```
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)  [inline]
```

11.16.1.67 pfspmm() [10/18]

```
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 )  [inline]
```

11.16.1.68 pfspmm() [11/18]

```
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 )  [inline]
```

11.16.1.69 pfspmm() [12/18]

```
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 )  [inline]
```

11.16.1.70 pfspmm() [13/18]

```
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 )  [inline]
```

11.16.1.71 pfspmm() [14/18]

```
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)  [inline]
```

11.16.1.72 pfspmm() [15/18]

```
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)  [inline]
```

11.16.1.73 pfspmm_zo() [1/2]

```
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)  [inline]
```

11.16.1.74 pfspmm_zo() [2/2]

```
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)  [inline]
```

11.16.1.75 pfspmv() [7/18]

```
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 )  [inline]
```

11.16.1.76 pfspmv() [8/18]

```
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 )  [inline]
```

11.16.1.77 pfspmv() [9/18]

```
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)  [inline]
```

11.16.1.78 pfspmv_one() [3/8]

```
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 )  [inline]
```

11.16.1.79 pfspmv_mone() [3/8]

```
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 )  [inline]
```

11.16.1.80 pfspmv_one() [4/8]

```
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 )  [inline]
```

11.16.1.81 pfspmv_mone() [4/8]

```
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 )  [inline]
```

11.16.1.82 fspmm() [10/15]

```
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 )  [inline]
```

11.16.1.83 fspmm() [11/15]

```
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 )  [inline]
```

11.16.1.84 fspmm() [12/15]

```
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)  [inline]
```

11.16.1.85 fspmm_mone() [3/4]

```
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 )  [inline]
```

11.16.1.86 fspmm_one() [3/4]

```
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 )  [inline]
```

11.16.1.87 fspmm_mone() [4/4]

```
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 )  [inline]
```

11.16.1.88 fspmm_one() [4/4]

```
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 )  [inline]
```

11.16.1.89 fspmm_one_simd_aligned() [3/3]

```
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 )  [inline]
```

11.16.1.90 fspmm_one_simd_unaligned() [3/3]

```
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 )  [inline]
```

11.16.1.91 fspmm_mone_simd_aligned() [3/3]

```
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 )  [inline]
```

11.16.1.92 fspmm_mone_simd_unaligned() [3/3]

```
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 ) [inline]
```

11.16.1.93 fspmv() [10/21]

```
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 ) [inline]
```

11.16.1.94 fspmv() [11/21]

```
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 ) [inline]
```

11.16.1.95 fspmv() [12/21]

```
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) [inline]
```

11.16.1.96 fspmv_one() [5/10]

```
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 ) [inline]
```

11.16.1.97 fspmv_mone() [5/10]

```
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 ) [inline]
```

11.16.1.98 fspmv_one() [6/10]

```
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 ) [inline]
```

11.16.1.99 fspmv_mone() [6/10]

```
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 ) [inline]
```

11.16.1.100 pfspmv() [10/18]

```
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 ) [inline]
```

11.16.1.101 pfspmv() [11/18]

```
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 ) [inline]
```

11.16.1.102 pfspmv() [12/18]

```
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)  [inline]
```

11.16.1.103 pfspmv_one() [5/8]

```
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 )  [inline]
```

11.16.1.104 pfspmv_mone() [5/8]

```
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 )  [inline]
```

11.16.1.105 pfspmv_one() [6/8]

```
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 )  [inline]
```

11.16.1.106 pfspmv_mone() [6/8]

```
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 )  [inline]
```

11.16.1.107 fspmv() [13/21]

```
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 )  [inline]
```

11.16.1.108 fspmv_simd() [1/4]

```
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 )  [inline]
```

11.16.1.109 fspmv() [14/21]

```
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 )  [inline]
```

11.16.1.110 fspmv_simd() [2/4]

```
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)  [inline]
```

11.16.1.111 fspmv() [15/21]

```
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)  [inline]
```

11.16.1.112 fspmv_one() [7/10]

```
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 ) [inline]
```

11.16.1.113 fspmv_mone() [7/10]

```
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 ) [inline]
```

11.16.1.114 fspmv_one() [8/10]

```
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 ) [inline]
```

11.16.1.115 fspmv_mone() [8/10]

```
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 ) [inline]
```

11.16.1.116 fspmv_one_simd() [1/2]

```
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 ) [inline]
```

11.16.1.117 fspmv_mone_simd() [1/2]

```
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 )  [inline]
```

11.16.1.118 pfspmm() [16/18]

```
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::GenericTag )  [inline]
```

11.16.1.119 pfspmm() [17/18]

```
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 )  [inline]
```

11.16.1.120 pfspmm() [18/18]

```
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,
            uint64_t kmax)  [inline]
```

11.16.1.121 pfspmv() [13/18]

```
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 )  [inline]
```

11.16.1.122 pfspmv() [14/18]

```
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 )  [inline]
```

11.16.1.123 pfspmv() [15/18]

```
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)  [inline]
```

11.16.1.124 fspmm() [13/15]

```
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 )  [inline]
```

11.16.1.125 fspmm() [14/15]

```
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 )  [inline]
```

11.16.1.126 fspmm() [15/15]

```
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)  [inline]
```

11.16.1.127 fspmv() [16/21]

```
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 )  [inline]
```

11.16.1.128 fspmv() [17/21]

```
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 )  [inline]
```

11.16.1.129 fspmv() [18/21]

```
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)  [inline]
```

11.16.1.130 pfspmv() [16/18]

```
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 )  [inline]
```

11.16.1.131 pfspmv() [17/18]

```
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 ) [inline]
```

11.16.1.132 pfspmv() [18/18]

```
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) [inline]
```

11.16.1.133 pfspmv_one() [7/8]

```
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 ) [inline]
```

11.16.1.134 pfspmv_mone() [7/8]

```
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 ) [inline]
```

11.16.1.135 pfspmv_one() [8/8]

```
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 ) [inline]
```

11.16.1.136 pfspmv_mone() [8/8]

```
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 ) [inline]
```

11.16.1.137 fspmv() [19/21]

```
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 ) [inline]
```

11.16.1.138 fspmv_simd() [3/4]

```
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 ) [inline]
```

11.16.1.139 fspmv() [20/21]

```
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 ) [inline]
```

11.16.1.140 fspmv_simd() [4/4]

```
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) [inline]
```

11.16.1.141 fspmv() [21/21]

```
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)  [inline]
```

11.16.1.142 fspmv_one() [9/10]

```
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 )  [inline]
```

11.16.1.143 fspmv_mone() [9/10]

```
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 )  [inline]
```

11.16.1.144 fspmv_one_simd() [2/2]

```
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 )  [inline]
```

11.16.1.145 fspmv_mone_simd() [2/2]

```
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 )  [inline]
```

11.16.1.146 fspmv_one() [10/10]

```
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 )  [inline]
```

11.16.1.147 fspmv_mone() [10/10]

```
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 )  [inline]
```

11.17 FFLAS::StrategyParameter Namespace Reference

Data Structures

- struct Fixed
- struct Grain
- struct Threads
- struct ThreeD
- struct ThreeDAdaptive
- struct ThreeDInPlace
- struct TwoD
- struct TwoDAdaptive

11.18 FFLAS::StructureHelper Namespace Reference

StructureHelper for ftrsm.

Data Structures

- struct Hybrid
- struct Iterative
- struct Recursive

11.18.1 Detailed Description

StructureHelper for ftrsm.

11.19 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 Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, const size_t incX, const size_t incY)
- 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< 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 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 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, const size_t &incY)

11.19.1 Function Documentation

11.19.1.1 VEC_ADD()

```
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)  [inline]
```

11.19.1.2 addp()

```
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_)  [inline]
```

11.19.1.3 VEC_SUB()

```
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) [inline]
```

11.19.1.4 subp()

```
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_) [inline]
```

11.19.1.5 add()

```
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) [inline]
```

11.19.1.6 sub()

```
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) [inline]
```

11.19.1.7 axpyp() [1/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp
(
            const Field & F,
            const typename Field::Element a,
            typename Field::ConstElement_ptr X,
            typename Field::Element_ptr Y,
            const size_t n) [inline]
```

11.19.1.8 axpyp() [2/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp
(
            const Field & F,
            const typename Field::Element a,
            typename Field::ConstElement_ptr X,
            typename Field::Element_ptr Y,
            const size_t n,
            const size_t incX,
            const size_t incY)  [inline]
```

11.19.1.9 reduce() [1/9]

```
template<class T>
std::enable_if<!std::is_integral< T >::value, T >::type reduce (
            T A,
            T B)  [inline]
```

11.19.1.10 reduce() [2/9]

```
template<class T>
std::enable_if< std::is_integral< T >::value, T >::type reduce (
            T A,
            T B)  [inline]
```

11.19.1.11 reduce() [3/9]

```
template<>
Givaro::Integer reduce (
            Givaro::Integer A,
            Givaro::Integer B)  [inline]
```

11.19.1.12 reduce() [4/9]

```
float reduce (
            float A,
            float B,
            float invB,
            float min,
            float max)  [inline]
```

11.19.1.13 reduce() [5/9]

```
double reduce (
            double A,
            double B,
            double invB,
            double min,
            double max)  [inline]
```

11.19.1.14 reduce() [6/9]

```
int64_t reduce (
            int64_t A,
            int64_t p,
            double invp,
            double min,
            double max,
            int64_t pow50rem) [inline]
```

11.19.1.15 reduce() [7/9]

```
template<class Field>
Field::Element reduce (
            typename Field::Element A,
            HelperMod< Field, ElementCategories::MachineIntTag > & H) [inline]
```

11.19.1.16 reduce() [8/9]

```
template<class Field>
Field::Element reduce (
            typename Field::Element A,
            HelperMod< Field, ElementCategories::MachineFloatTag > & H) [inline]
```

11.19.1.17 reduce() [9/9]

```
template<class Field>
Field::Element reduce (
            typename Field::Element A,
            HelperMod< Field, ElementCategories::ArbitraryPrecIntTag > & H) [inline]
```

11.19.1.18 modp() [1/2]

```
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) [inline]
```

11.19.1.19 modp() [2/2]

```
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) [inline]
```

11.19.1.20 scalp() [1/3]

```
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)  [inline]
```

11.19.1.21 scalp() [2/3]

```
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)  [inline]
```

11.19.1.22 scalp() [3/3]

```
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,
            const size_t & incY)  [inline]
```

11.20 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 axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, HelperMod< Field > &H)
- template<class Field>
  std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, const size_t incX, const size_t incY, 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 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)
- 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, const size_t &incY, HelperMod< Field > &H)

11.20.1 Function Documentation

11.20.1.1 axpyp() [1/2]

```
template<class Field>
std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod<
typenameField::Element >::value, void >::type axpyp (
            const Field & F,
            const typename Field::Element a,
            typename Field::ConstElement_ptr X,
            typename Field::Element_ptr Y,
            const size_t n,
            HelperMod< Field > & H) [inline]
```

11.20.1.2 axpyp() [2/2]

```
template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp
(
            const Field & F,
            const typename Field::Element a,
            typename Field::ConstElement_ptr X,
            typename Field::Element_ptr Y,
            const size_t n,
            const size_t incX,
            const size_t incY,
            HelperMod< Field > & H) [inline]
```

11.20.1.3 modp() [1/2]

```
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)  [inline]
```

11.20.1.4 modp() [2/2]

```
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)  [inline]
```

11.20.1.5 scalp() [1/3]

```
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)  [inline]
```

11.20.1.6 scalp() [2/3]

```
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)  [inline]
```

11.20.1.7 scalp() [3/3]

```
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,
            const size_t & incY,
            HelperMod< Field > & H)  [inline]
```

11.21 FFPACK Namespace Reference

Finite **F**ield **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_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >
- class CheckerImplem_Det
- class CheckerImplem_invert
- class CheckerImplem_PLUQ
- class Failure

    A precondtion 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 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>
  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 A X = B or X A = 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 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 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_THRESHOL
- 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_
- 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 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 lda, size_t ∗P, size_t ∗Qt, const bool transform=false, const FFPACK_LU_TAG Lu←Tag=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 $(v, Av, ..., A^N v)$.
- 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 L X = B or X L = 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*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*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\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\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\U of rank R obtained by RowEchelonForm or Column↩ EchelonForm.
- 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\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\U of rank R obtained by RowEchelon↩ Form 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 Lu←Tag=FfpackSlabRecursive)

  Extracts a matrix in echelon form from a compact storage A=L\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\U of rank R obtained by ReducedRowEchelonForm or ReducedColumnEchelon←Form 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\U of rank R obtained by RowEchelon←Form 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>
  size_t LTBruhatGen (const Field &Fi, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t ∗P, size_t ∗Q)

  LTBruhatGen Suppose A is Left Triangular Matrix This procedure computes the Bruhat Representation of A and return the rank of A.
- template<class Field>
  void getLTBruhatGen (const Field &Fi, const size_t N, const size_t r, const size_t ∗P, const size_t ∗Q, typename Field::Element_ptr R, const size_t ldr)

  GetLTBruhatGen This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.
- template<class Field>
  void getLTBruhatGen (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, 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)

  GetLTBruhatGen This procedure computes the matrix L or U f the Bruhat Representation Suppose that A is the bruhat representation of a matrix.
- size_t LTQSorder (const size_t N, const size_t r, const size_t ∗P, const size_t ∗Q)

  LTQSorder This procedure computes the order of quasiseparability of a matrix.
- template<class Field>
  size_t CompressToBlockBiDiagonal (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size←_t s, size_t r, const size_t ∗P, const size_t ∗Q, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t ∗K, size_t ∗M, size_t ∗T)

  CompressToBlockBiDiagonal This procedure compress a compact representation of a row echelon form or column echelon form.
- template<class Field>
  void ExpandBlockBiDiagonalToBruhat (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size←_t s, size_t r, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t NbBlocks, size_t ∗K, size_t ∗M, size_t ∗T)

  ExpandBlockBiDiagonal This procedure expand a compact representation of a row echelon form or column echelon form.
- void Bruhat2EchelonPermutation (size_t N, size_t R, const size_t ∗P, const size_t ∗Q, size_t ∗M)

  Bruhat2EchelonPermutation (N,R,P,Q) Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Rpresentation.
- size_t ∗ TInverter (size_t ∗T, size_t r)

- template<class Field>
  void ComputeRPermutation (const Field &Fi, size_t N, size_t r, const size_t ∗P, const size_t ∗Q, size_t ∗R, size_t ∗MU, size_t ∗ML)
- template<class Field>
  void productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, const size_t ∗P, const size_t ∗Q, const typename Field::Element_ptr Xu, size_t ldu, size_t NbBlocksU, size_t ∗Ku, size_t ∗Tu, size_t ∗MU, const typename Field::Element_ptr Xl, size_t ldl, size_t NbBlocksL, size_t ∗Kl, size_t ∗Tl, size_t ∗ML, typename Field::Element_ptr B, size_t t, size_t ldb, typename Field::Element_ptr C, size_t ldc)

  productBruhatxTS Comput the product between the CRE compact representation of a matrix A and B a tall matrix
- 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.
- 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 L X = B or X L = 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)
- size_t ∗ TInverter (const size_t ∗T, size_t r)
- template<class Field>
  void ComputeRPermutation (const Field &Fi, size_t N, size_t r, const size_t ∗P, const size_t ∗Q, size_t ∗R, const size_t ∗MU, const size_t ∗ML)
- template<class Field>
  Field::Element_ptr expandLCRE (const Field &Fi, size_t N, size_t s, size_t r, size_t ∗R, size_t i, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t ∗Ku, const size_t ∗Tuinv, typename Field::ConstElement_ptr Xl, size_t ldl, size_t NbBlocksL, const size_t ∗Kl, const size_t ∗Tlinv, typename Field::Element_ptr CRE, size_t ldcre)

  Expands an anti-diagonal block of a left triangular matrix from its compact Bruhat representation.
- template<class Field>
  void productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, size_t t, const size_t ∗P, const size_t ∗Q, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t ∗Ku, const size_t ∗Tu, const size_t ∗MU, typename Field::ConstElement_ptr Xl, size_t ldl, size_t NbBlocksL, const size_t ∗Kl, const size_t ∗Tl, const size_t ∗ML, typename Field::Element_ptr B, size_t ldb, const typename Field::Element beta, typename Field::Element_ptr D, size_t ldd)

  Compute the product of a left-triangular quasi-separable matrix A, represented by a compact Bruhat generator, with a dense rectangular matrix B: $C \leftarrow A \times B + betaC$.
- 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 &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>
  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 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.

- void composePermutationsLLM (size_t ∗MathP, const size_t ∗P1, const size_t ∗P2, const size_t R, const size_t N)

  Computes P1 x 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>
  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, 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 > &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 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 $\{0, \ldots, N-1\}$ 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 x 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 x N : a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.
- void RandomLTQSRankProfileMatrix (size_t n, size_t r, size_t t, size_t *rows, size_t *cols)
- 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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 RandIter>
  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)

  Random Matrix with prescribed det.
- template<class Field, class RandIter>
  Field::Element_ptr RandomLTQSMatrixWithRankandQSorder (Field &F, size_t n, size_t r, size_t t, typename Field::Element_ptr A, size_t lda, RandIter &G)
- 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)

11.21.1 Detailed Description

Finite **F**ield **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.

11.21.2 Typedef Documentation

11.21.2.1 Checker_PLUQ

```
template<class Field>
using Checker_PLUQ = FFLAS::Checker_Empty<Field>
```

11.21.2.2 Checker_Det

```
template<class Field>
using Checker_Det = FFLAS::Checker_Empty<Field>
```

11.21.2.3 Checker_invert

```
template<class Field>
using Checker_invert = FFLAS::Checker_Empty<Field>
```

11.21.2.4 Checker_charpoly

```
template<class Field, class Polynomial>
using Checker_charpoly = FFLAS::Checker_Empty<Field>
```

11.21.2.5 ForceCheck_PLUQ

```
template<class Field>
using ForceCheck_PLUQ = CheckerImplem_PLUQ<Field>
```

11.21.2.6 ForceCheck_Det

```
template<class Field>
using ForceCheck_Det = CheckerImplem_Det<Field>
```

11.21.2.7 ForceCheck_invert

```
template<class Field>
using ForceCheck_invert = CheckerImplem_invert<Field>
```

11.21.2.8 ForceCheck_charpoly

```
template<class Field, class Polynomial>
using ForceCheck_charpoly = CheckerImplem_charpoly<Field,Polynomial>
```

11.21.3 Function Documentation

11.21.3.1 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.

11.21.3.2 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.

11.21.3.3 applyP() [1/4]

```
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)  [inline]
```

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.

Parameters

in,out	*P1*	a LAPACK permutation of size N
	P2	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

F	base field
Side	decides if rows (FflasLeft) or columns (FflasRight) are permuted
Trans	decides if the matrix is seen as columns (FflasTrans) or rows (FflasNoTrans)
M	size of the elements to permute
ibeg	first index to consider in P
iend	last index to consider in P
A	input matrix
lda	leading dimension of A
P	permutation in LAPACK format
psh	(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]

11.21.3.4 applyP() [2/4]

```
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)  [inline]
```

11.21.3.5 applyP() [3/4]

```
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)  [inline]
```

11.21.3.6 MonotonicApplyP()

```
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)  [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 reprensentation (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

F	base field
Side	selects if it is a row (FflasLeft) or column (FflasRight) permutation
Trans	inverse permutation (FflasTrans/NoTrans)
M	
ibeg	
iend	
A	input matrix
lda	leading dimension of A
P	LAPACK permuation
R	first values of P

The non pivot elements, are located in montonically increasing order.

11.21.3.7 fgetrs() [1/4]

```
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 $XA = B$.

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

F	base field
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
M	row dimension of B
N	col dimension of B
R	rank of A
A	input matrix
lda	leading dimension of A
P	row permutation of the PLUQ decomposition of A
Q	column permutation of the PLUQ decomposition of A
B	Right/Left hand side matrix. Initially stores B, finally stores the solution X.
ldb	leading dimension of B
info	Success of the computation: 0 if successfull, $>$0 if system is inconsistent

11.21.3.8 fgetrs() [2/4]

```
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 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

F	base field
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking.
M	row dimension of A
N	col dimension of A
NRHS	number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B
R	rank of A
A	input matrix
lda	leading dimension of A
P	row permutation of the PLUQ decomposition of A
Q	column permutation of the PLUQ decomposition of A
X	solution matrix
ldx	leading dimension of X
B	Right/Left hand side matrix.
ldb	leading dimension of B
info	Succes of the computation: 0 if successfull, >0 if system is inconsistent

11.21.3.9 fgesv() [1/4]

```
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.

Parameters

F	The computation domain
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
M	row dimension of B
N	col dimension of B
A	input matrix
lda	leading dimension of A
B	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
ldb	leading dimension of B
info	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

11.21.3.10 fgesv() [2/4]

```
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.

Parameters

F	The computation domain
Side	Determine wheter the resolution is left (FflasLeft) or right (FflasRight) looking
M	row dimension of A
N	col dimension of A

Parameters

NRHS	number of columns (if Side = FFLAS::FflasLeft) or row (if Side = FFLAS::FflasRight) of the matrices X and B
A	input matrix
lda	leading dimension of A
B	Right/Left hand side matrix. Initially contains B, finally contains the solution X.
ldb	leading dimension of B
X	
ldx	
info	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

11.21.3.11 ftrtri() [1/2]

```
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.

Parameters

F	base field
Uplo	whether the matrix is upper or lower triangular
Diag	whether the matrix is unit diagonal (FflasUnit/NoUnit)
N	input matrix order
A	the input matrix
lda	leading dimension of A

11.21.3.12 trinv_left() [1/2]

```
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)
```

11.21.3.13 ftrtrm() [1/2]

```
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.

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

F	base field
Side	set to FflasLeft to compute the product UL, FflasRight to compute LU
diag	whether the matrix U is unit diagonal (FflasUnit/NoUnit)
N	input matrix order
A	the input matrix
lda	leading dimension of A

11.21.3.14 ftrstr()

```
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)  [inline]
```

Solve a triangular system with a triangular right hand side of the same shape.

Parameters

F	base field
Side	set to FflasLeft to compute U1^-1∗U2 or L1^-1∗L2, FflasRight to compute U1∗U2^-1 or L1∗L2^-1
Uplo	whether the matrix A is upper or lower triangular
diag1	whether the matrix U1 or L2 is unit diagonal (FflasUnit/NoUnit)
diag2	whether the matrix U2 or L2 is unit diagonal (FflasUnit/NoUnit)
N	order of the input matrices
A	the input matrix to be inverted (U1 or L1)
lda	leading dimension of A
B	the input right hand side (U2 or L2)
ldb	leading dimension of B

11.21.3.15 ftrssyr2k()

```
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)  [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

	F	base field
	Side	set to FflasLeft to compute U1^-1∗U2 or L1^-1∗L2, FflasRight to compute U1∗U2^-1 or L1∗L2^-1
	Uplo	whether the matrix A is upper or lower triangular
	diagA	whether the matrix A is unit diagonal (FflasUnit/NoUnit)
	N	order of the input matrices
	A	the input matrix
	lda	leading dimension of A
in,out	*B*	the input right hand side where the output is written
	ldb	leading dimension of B

11.21.3.16 fsytrf() [1/3]

```
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.

Parameters

	F	The computation domain
	UpLo	Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor
	N	order of the matrix A
in,out	*A*	input matrix
	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: $A = L \times D \times L^T$ 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.

11.21.3.17 fsytrf() [2/3]

```
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)  [inline]
```

11.21.3.18 fsytrf() [3/3]

```
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)  [inline]
```

11.21.3.19 fsytrf_nonunit() [1/3]

```
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.

Parameters

	F	The computation domain
	UpLo	Determine wheter to store the upper (FflasUpper) or lower (FflasLower) triangular factor

	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: $A = L \times Dinv \times L^T$ 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.

11.21.3.20 PLUQ() [1/6]

```
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)  [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

11.21.3.21 pPLUQ()

```
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)  [inline]
```

11.21.3.22 PLUQ() [2/6]

```
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)  [inline]
```

11.21.3.23 PLUQ() [3/6]

```
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)
```

11.21.3.24 LUdivine() [1/4]

```
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.

Using a block algorithm and return its rank. The permutations P and Q are represented using LAPACK's convention.

Parameters

F	base field
Diag	whether the transformation matrix (U of the CUP, L of the PLE) should have a unit diagonal (FflasUnit) or not (FflasNoUnit)
trans	whether to compute the CUP decomposition (FflasNoTrans) or the PLE decomposition (FflasTrans)
M	matrix row dimension
N	matrix column dimension
A	input matrix
lda	leading dimension of `A`
P	the factor of CUP or PLE
Q	a permutation indicating the pivot position in the echelon form C or E in its first r positions
LuTag	flag for setting the earling termination if the matrix is singular
cutoff	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

11.21.3.25 ColumnEchelonForm() [1/3]

```
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 = FfpackSlabRecursive)  [inline]
```

Compute the Column Echelon form of the input matrix in-place.

If LuTag == FfpackTileRecursive, then after the computation A = [M \ V] such that AU = C is a column echelon decomposition of A, with U = P^T [V] and C = M + Q [Ir] [0 In-r] [0] If LuTag == FfpackTileRecursive then A = [N \ V] such that the same holds with M = Q N

Qt = Q^T If transform=false, the matrix V is not computed. See also test-colechelon for an example of use

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 V is computed
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.21.3.26 pColumnEchelonForm()

```
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)  [inline]
```

11.21.3.27 ColumnEchelonForm() [2/3]

```
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)  [inline]
```

11.21.3.28 RowEchelonForm() [1/3]

```
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 = FfpackSlabRecursive)  [inline]
```

Compute the Row Echelon form of the input matrix in-place.

If LuTag == FfpackTileRecursive, then after the computation A = [L \ M] such that X A = R is a row echelon decomposition of A, with X = [L 0] P and R = M + [Ir 0] Q^T [In-r] If LuTag == FfpackTileRecursive then A = [L \ N] such that the same holds with M = N Q^T Qt = Q^T If transform=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
	transform	decides whether L is computed
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

11.21.3.29 pRowEchelonForm()

```
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)  [inline]
```

11.21.3.30 RowEchelonForm() [2/3]

```
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)  [inline]
```

11.21.3.31 ReducedColumnEchelonForm() [1/3]

```
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)  [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 [Ir] [0 In-r] [M 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 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

11.21.3.32 pReducedColumnEchelonForm()

```
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)  [inline]
```

11.21.3.33 ReducedColumnEchelonForm() [2/3]

```
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)  [inline]
```

11.21.3.34 ReducedRowEchelonForm() [1/3]

```
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)  [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 = [Ir 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

11.21.3.35 pReducedRowEchelonForm()

```
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)  [inline]
```

11.21.3.36 ReducedRowEchelonForm() [2/3]

```
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)  [inline]
```

11.21.3.37 Invert() [1/4]

```
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.

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 $A \leftarrow A^{-1}$

11.21.3.38 Invert() [2/4]

```
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.

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 $\texttt{nullity} = 0$). It is untouched otherwise.
	ldx	leading dimension of X
	nullity	dimension of the kernel of A

Returns

pointer to $X = A^{-1}$

11.21.3.39 Invert2() [1/2]

```
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.

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 $\texttt{nullity} = 0$). 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)

11.21.3.40 CharPoly() [1/8]

```
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)  [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 as a list of factors
	N	order of the matrix A
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 LUKrylov and ArithProg)

11.21.3.41 CharPoly() [2/8]

```
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)  [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 as a single polynomial
	N	order of the matrix A
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 LUKrylov and ArithProg)

11.21.3.42 CharPoly() [3/8]

```
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 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 `as` a single polynomial
	N	order of the matrix `A`
in	*A*	the input matrix ($N \times N$) (could be overwritten in some algorithmic variants)
	lda	leading dimension of `A`
	CharpTag	the algorithmic variant

11.21.3.43 MinPoly() [1/4]

```
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)  [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, .., A^kv)

Parameters

	F	the base field
out	*minP*	the minimal polynomial of `A`
	N	order of the matrix `A`
in	*A*	the input matrix ($N \times N$)
	lda	leading dimension of `A`

11.21.3.44 MinPoly() [2/4]

```
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) [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, .., A^kv)

Parameters

	F	the base field
out	*minP*	the minimal polynomial of A
	N	order of the matrix A
in	*A*	the input matrix ($N \times N$)
	lda	leading dimension of A
	G	a random iterator

11.21.3.45 MatVecMinPoly() [1/2]

```
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) [inline]
```

Compute the minimal polynomial of the matrix A and a vector v, namely the first linear dependency relation in the Krylov basis $(v, Av, ..., A^N v)$.

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

11.21.3.46 Rank() [1/3]

```
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.

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

11.21.3.47 pRank()

```
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)
```

11.21.3.48 Rank() [2/3]

```
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)
```

11.21.3.49 IsSingular() [1/2]

```
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.

The method is a block elimination with early termination

using LQUP factorization with early termination. If `M != 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

11.21.3.50 Det() [1/6]

```
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) [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

	F	base field
out	*det*	the determinant of A
	N	the order of the square matrix A.
in,out	*A*	input matrix
	lda	leading dimension of A
	psH	(optional) a ParSeqHelper to choose between sequential and parallel execution
	P,Q	(optional) row and column permutations to be used by the PLUQ factorization. randomized checkers (see cherckes/checker_det.inl) need them for certification

11.21.3.51 pDet()

```
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) [inline]
```

11.21.3.52 Det() [2/6]

```
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)
```

11.21.3.53 Solve() [1/3]

```
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) [inline]
```

Solves a linear system AX = b using PLUQ factorization.

@oaram F base field @oaram M matrix order

Parameters

in	*A*	input matrix
	lda	leading dimension of A
out	*x*	output solution vector
	incx	increment of x
	b	input right hand side of the system
	incb	increment of b

11.21.3.54 Solve() [2/3]

```
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)
```

11.21.3.55 pSolve()

```
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)  [inline]
```

11.21.3.56 RandomNullSpaceVector() [1/3]

```
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 L X = B or X L = B in place.

L is M∗M if Side == FFLAS::FflasLeft and N∗N if Side == FFLAS::FflasRight, B is M∗N. Only the R non trivial column of L are stored in the M∗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

	F	The computation domain
	Side	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	M	number of rows
	N	number of columns
in,out	*A*	input matrix of dimension M x N, A is modified to its LU version
	lda	leading dimension of A
out	*X*	output vector
	incX	increment of X

11.21.3.57 NullSpaceBasis() [1/2]

```
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.

return the dimension of the nullspace.

Parameters

	F	The computation domain
	Side	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	M	number of rows
	N	number of columns
in,out	*A*	input matrix of dimension M x N, A is modified
	lda	leading dimension of A
out	*NS*	output matrix of dimension N x NSdim (allocated here)
out	*ldn*	leading dimension of NS
out	*NSdim*	the dimension of the Nullspace (N-rank(A))

11.21.3.58 RowRankProfile() [1/3]

```
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) [inline]
```

Computes the row rank profile 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	*rkprofile*	return the rank profile as an array of row indexes, of dimension r=rank(A)
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

A is modified rkprofile is allocated during the computation.

Returns

R

11.21.3.59 pRowRankProfile()

```
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)  [inline]
```

11.21.3.60 RowRankProfile() [2/3]

```
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)  [inline]
```

11.21.3.61 ColumnRankProfile() [1/3]

```
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)  [inline]
```

Computes the column rank profile of A.

Parameters

	F	base field
	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=rank(A)
	LuTag	chooses the elimination algorithm. SlabRecursive for LUdivine, TileRecursive for PLUQ

A is modified rkprofile is allocated during the computation.

Returns

R

11.21.3.62 pColumnRankProfile()

```
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)  [inline]
```

11.21.3.63 ColumnRankProfile() [2/3]

```
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)  [inline]
```

11.21.3.64 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

11.21.3.65 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 x 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.

11.21.3.66 RowRankProfileSubmatrixIndices() [1/2]

```
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.

Computes the indices of the submatrix r*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
	rowindices	array of the row indices of X in A
	colindices	array of the col indices of X in A
	lda	leading dimension of A
out	*R*	list of indices

rowindices and colindices are allocated during the computation. A is modified

Returns

R

11.21.3.67 ColRankProfileSubmatrixIndices() [1/2]

```
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∗r X of A whose columns correspond to the column rank profile of A.

Parameters

	F	base field
	M	number of rows
	N	number of columns
in	*A*	input matrix of dimension
	rowindices	array of the row indices of X in A
	colindices	array of the col indices of X in A
	lda	leading dimension of A
out	*R*	list of indices

rowindices and colindices are allocated during the computation.

Warning

A is modified

Returns

R

11.21.3.68 RowRankProfileSubmatrix() [1/2]

```
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*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

11.21.3.69 ColRankProfileSubmatrix() [1/2]

```
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.

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

11.21.3.70 getTriangular() [1/2]

```
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)  [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

	F	base field
	UpLo	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is returned
	diag	selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)
	M	row dimension of T
	N	column dimension of T
	R	rank of the triangular matrix (how many rows/columns need to be copied)
in	*A*	input matrix
	lda	leading dimension of A
out	*T*	output matrix
	ldt	leading dimension of T
	OnlyNonZeroVectors	decides whether the last zero rows/columns should be ignored

Todo just one triangular fzero+fassign ?

11.21.3.71 getTriangular() [2/2]

```
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)  [inline]
```

Cleans up a compact storage A=L\U to reveal a triangular matrix of rank R.

Parameters

	F	base field
	UpLo	selects if the upper (FflasUpper) or lower (FflasLower) triangular matrix is revealed
	diag	selects if the triangular matrix unit-diagonal (FflasUnit/NoUnit)
	M	row dimension of A
	N	column dimension of A
	R	rank of the triangular matrix
in,out	*A*	input/output matrix
	lda	leading dimension of A

Todo just one triangular fzero+fassign ?

11.21.3.72 getEchelonForm() [1/2]

```
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) [inline]
```

Extracts a matrix in echelon form from a compact storage A=L\U 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

	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 T
	N	column dimension of T
	R	rank of the triangular matrix (how many rows/columns need to be copied)
	P	positions of the R pivots
in	*A*	input matrix
	lda	leading dimension of A
out	*T*	output matrix
	ldt	leading dimension of T
	OnlyNonZeroVectors	decides whether the last zero rows/columns should be ignored
	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.21.3.73 getEchelonForm() [2/2]

```
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)  [inline]
```

Cleans up a compact storage A=L\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)

11.21.3.74 getEchelonTransform()

```
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)  [inline]
```

Extracts a transformation matrix to echelon form from a compact storage A=L\U of rank R obtained by Row↩ EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower: T is N x N (already allocated) such that A T = C is a transformation of A in Column echelon form Else T is M x M (already allocated) such that T A = E is a transformation of A in Row Echelon form

Parameters

	F	base field
	UpLo	Lower (FflasLower) means Transformation to Column Echelon Form, Upper (FflasUpper), to Row Echelon Form
	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
	P	permutation matrix
in	*A*	input matrix
	lda	leading dimension of A
out	*T*	output matrix
	ldt	leading dimension of T
	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.21.3.75 getReducedEchelonForm() [1/2]

```
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)  [inline]
```

Extracts a matrix in echelon form from a compact storage A=L\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

	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 T
	N	column dimension of T
	R	rank of the triangular matrix (how many rows/columns need to be copied)
	P	positions of the R pivots
in	*A*	input matrix
	lda	leading dimension of A
	ldt	leading dimension of T

Parameters

	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)
	OnlyNonZeroVectors	decides whether the last zero rows/columns should be ignored

11.21.3.76 getReducedEchelonForm() [2/2]

```
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)  [inline]
```

Cleans up a compact storage A=L\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

	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)

11.21.3.77 getReducedEchelonTransform()

```
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) [inline]
```

Extracts a transformation matrix to echelon form from a compact storage A=L\U of rank R obtained by Row↩EchelonForm or ColumnEchelonForm.

If Uplo == FflasLower: T is N x N (already allocated) such that A T = C is a transformation of A in Column echelon form Else T is M x M (already allocated) such that T A = E is a transformation of A in Row Echelon form

Parameters

	F	base field
	UpLo	selects Col (FflasLower) or Row (FflasUpper) Echelon Form
	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
	P	permutation matrix
in	*A*	input matrix
	lda	leading dimension of A
out	*T*	output matrix
	ldt	leading dimension of T
	LuTag	which factorized form (CUP/PLE if FfpackSlabRecursive, PLUQ if FfpackTileRecursive)

11.21.3.78 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.

11.21.3.79 LTBruhatGen()

```
template<class Field>
size_t LTBruhatGen (
            const Field & Fi,
            const FFLAS::FFLAS_DIAG diag,
            const size_t N,
            typename Field::Element_ptr A,
            const size_t lda,
            size_t * P,
            size_t * Q) [inline]
```

LTBruhatGen Suppose A is Left Triangular Matrix This procedure computes the Bruhat Representation of A and return the rank of A.

Parameters

Fi	base Field
diag	
N	size of A
A	the matrix we search the Bruhat representation
lda	the leading dimension of A
P	a permutation matrix
Q	a permutation matrix

11.21.3.80 getLTBruhatGen() [1/2]

```
template<class Field>
void getLTBruhatGen (
            const Field & Fi,
            const size_t N,
            const size_t r,
            const size_t * P,
            const size_t * Q,
            typename Field::Element_ptr R,
            const size_t ldr)  [inline]
```

GetLTBruhatGen This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.

Parameters

Fi	base Field
N	size of the matrix
r	the rank of the matrix
P	a permutation matrix
Q	a permutation matrix
R	the matrix that will contain the rank revealing matrix
ldr	the leading fimension of R

11.21.3.81 getLTBruhatGen() [2/2]

```
template<class Field>
void getLTBruhatGen (
            const Field & Fi,
            const FFLAS::FFLAS_UPLO Uplo,
            const FFLAS::FFLAS_DIAG diag,
            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)  [inline]
```

GetLTBruhatGen This procedure computes the matrix L or U f the Bruhat Representation Suppose that A is the bruhat representation of a matrix.

Parameters

Fi	base Field
Uplo	choose if the procedure return L or U
diag	
N	size of A
r	rank of A
P	permutaion matrix
Q	permutation matrix
A	a bruhat representation
lda	leading dimension of A
T	matrix that will contains L or U
ldt	leading dimension of T

11.21.3.82 LTQSorder()

```
size_t LTQSorder (
            const size_t N,
            const size_t r,
            const size_t * P,
            const size_t * Q)  [inline]
```

LTQSorder This procedure computes the order of quasiseparability of a matrix.

Parameters

N	size of the matrix
r	rank of the matrix
P	permutation matrix
Q	permutation matrix

11.21.3.83 CompressToBlockBiDiagonal()

```
template<class Field>
size_t CompressToBlockBiDiagonal (
            const Field & Fi,
            const FFLAS::FFLAS_UPLO Uplo,
            size_t N,
            size_t s,
            size_t r,
            const size_t * P,
            const size_t * Q,
            typename Field::Element_ptr A,
            size_t lda,
            typename Field::Element_ptr X,
            size_t ldx,
            size_t * K,
            size_t * M,
            size_t * T)  [inline]
```

CompressToBlockBiDiagonal This procedure compress a compact representation of a row echelon form or column echelon form.

Parameters

Fi	base Field
Uplo	chosse if the procedure is based on row or column
N	size of the matrix
s	order of qausiseparability
r	rank
P	permutation matrix
Q	permutation matrix
A	the matrix to compact
lda	leading dimension of A
X	matrix that will stock the representation
ldx	leading dimension of X
K	stock the position of the blocks in A
M	permutation matrix
T	stock the operation done in the procedure

11.21.3.84 ExpandBlockBiDiagonalToBruhat()

```
template<class Field>
void ExpandBlockBiDiagonalToBruhat (
            const Field & Fi,
            const FFLAS::FFLAS_UPLO Uplo,
            size_t N,
            size_t s,
            size_t r,
            typename Field::Element_ptr A,
            size_t lda,
            typename Field::Element_ptr X,
            size_t ldx,
            size_t NbBlocks,
            size_t * K,
            size_t * M,
            size_t * T) [inline]
```

ExpandBlockBiDiagonal This procedure expand a compact representation of a row echelon form or column echelon form.

Parameters

Fi	base Field
Uplo	chosse if the procedure is based on row or column
N	size of the matrix
s	order of qausiseparability
r	rank
A	the matrix that will sotck the expanded representation
lda	leading dimension of A
X	matrix to expand
ldx	leading dimension of X
K	stock the position of the blocks in A
M	permutation matrix
T	stock the operation done in the procedure

11.21.3.85 Bruhat2EchelonPermutation()

```
void Bruhat2EchelonPermutation (
            size_t N,
            size_t R,
            const size_t * P,
            const size_t * Q,
            size_t * M)  [inline]
```

Bruhat2EchelonPermutation (N,R,P,Q) Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Rpresentation.

Parameters

in	*N*	size of the matrix
in	*R*	rank
in	*P*	permutation Matrix
in	*Q*	permutation Matrix
out	*M*	output permutation matrix

11.21.3.86 TInverter() [1/2]

```
size_t * TInverter (
            size_t * T,
            size_t r)
```

11.21.3.87 ComputeRPermutation() [1/2]

```
template<class Field>
void ComputeRPermutation (
            const Field & Fi,
            size_t N,
            size_t r,
            const size_t * P,
            const size_t * Q,
            size_t * R,
            size_t * MU,
            size_t * ML)
```

11.21.3.88 productBruhatxTS() [1/2]

```
template<class Field>
void productBruhatxTS (
            const Field & Fi,
            size_t N,
            size_t s,
            size_t r,
            const size_t * P,
            const size_t * Q,
            const typename Field::Element_ptr Xu,
            size_t ldu,
```

```
              size_t NbBlocksU,
              size_t * Ku,
              size_t * Tu,
              size_t * MU,
              const typename Field::Element_ptr Xl,
              size_t ldl,
              size_t NbBlocksL,
              size_t * Kl,
              size_t * Tl,
              size_t * ML,
              typename Field::Element_ptr B,
              size_t t,
              size_t ldb,
              typename Field::Element_ptr C,
              size_t ldc)
```

productBruhatxTS Comput the product between the CRE compact representation of a matrix A and B a tall matrix

11.21.3.89 LQUPtoInverseOfFullRankMinor() [1/2]

```
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.

Suppose A has been factorized as L.Q.U.P, with rank r. Then Qt.A.Pt has an invertible leading principal r x 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

F	base field
rank	rank of the matrix.
A_factors	matrix containing the L and U entries of the factorization
lda	leading dimension of A
QtPointer	theLQUP->getQ()->getPointer() (note: getQ returns Qt!)
X	desired location for output
ldx	leading dimension of X

11.21.3.90 RandomNullSpaceVector() [2/3]

```
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 L X = B or X L = B in place.

L is M∗M if Side == FFLAS::FflasLeft and N∗N if Side == FFLAS::FflasRight, B is M∗N. Only the R non trivial column of L are stored in the M∗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

	F	The computation domain
	Side	decides whether it computes the left (FflasLeft) or right (FflasRight) nullspace
	M	number of rows
	N	number of columns
in,out	*A*	input matrix of dimension M x N, A is modified to its LU version
	lda	leading dimension of A
out	*X*	output vector
	incX	increment of X

11.21.3.91 solveLB() [1/2]

```
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)
```

11.21.3.92 solveLB2() [1/2]

```
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)
```

11.21.3.93 TInverter() [2/2]

```
size_t * TInverter (
            const size_t * T,
            size_t r)  [inline]
```

11.21.3.94 ComputeRPermutation() [2/2]

```
template<class Field>
void ComputeRPermutation (
            const Field & Fi,
            size_t N,
            size_t r,
            const size_t * P,
            const size_t * Q,
            size_t * R,
            const size_t * MU,
            const size_t * ML)  [inline]
```

11.21.3.95 expandLCRE()

```
template<class Field>
Field::Element_ptr expandLCRE (
            const Field & Fi,
            size_t N,
            size_t s,
            size_t r,
            size_t * R,
            size_t i,
            typename Field::ConstElement_ptr Xu,
            size_t ldu,
            size_t NbBlocksU,
            const size_t * Ku,
            const size_t * Tuinv,
            typename Field::ConstElement_ptr Xl,
            size_t ldl,
            size_t NbBlocksL,
            const size_t * Kl,
            const size_t * Tlinv,
            typename Field::Element_ptr CRE,
            size_t ldcre)  [inline]
```

Expands an anti-diagonal block of a left triangular matrix from its compact Bruhat representation.

11.21.3.96 productBruhatxTS() [2/2]

```
template<class Field>
void productBruhatxTS (
            const Field & Fi,
            size_t N,
            size_t s,
            size_t r,
            size_t t,
            const size_t * P,
            const size_t * Q,
            typename Field::ConstElement_ptr Xu,
            size_t ldu,
            size_t NbBlocksU,
            const size_t * Ku,
            const size_t * Tu,
            const size_t * MU,
            typename Field::ConstElement_ptr Xl,
            size_t ldl,
            size_t NbBlocksL,
            const size_t * Kl,
            const size_t * Tl,
            const size_t * ML,
            typename Field::Element_ptr B,
            size_t ldb,
            const typename Field::Element beta,
            typename Field::Element_ptr D,
            size_t ldd)  [inline]
```

Compute the product of a left-triangular quasi-separable matrix A, represented by a compact Bruhat generator, with a dense rectangular matrix B: $C \leftarrow A \times B + betaC$.

Parameters

	F	the base field
	N	the order of A
	s	the order of quasiseparability of A
	r	the number of pivots in the left-triangular par of the rank profile matrix of A
	t	the number of columns of B
	P	the row indices of the pivots of A
	Q	the column indices of the pivots of A
	Xu	the compact storage of U: Du blocks in the first s rows, Su blocks in the last s rows
	ldxu	the leading dimension of Xu
	NbBlocksU	the number of diagonal blocks in the compact storage of U
	Ku	the list of starting column positions for each block of the storage of U
	Tu	the folding matrix for the compact storage of U: $Du + TuSu$ is in row echelon form
	Mu	a permutation matrix such that $Mu(Du + TuSu)$ is the U factor of the Bruhat generator
	Xl	the compact storage of L: Dl blocks in the first s columns, Sl blocks in the last s columns
	ldxl	the leading dimension of Xl
	NbBlocksL	the number of diagonal blocks in the compact storage of L
	Kl	the list of starting row positions for each block of the storage of L
	Tl	the folding matrix for the compact storage of L: $Dl + SlTl$ is in column echelon form

Parameters

	Ml	a permutation matrix such that $(Dl + TlSl)Ml$ is the L factor of the Bruhat generator
	B	an $N \times t$ dense matrix
	ldb	leading dimension of B
	beta	scaling constant
in,out	*C*	output matrix
	ldc	leading dimension of C

Bibliography Pernet C. and Storjohann A. *Time and space efficient generators for quasiseparable matrices* , JSC (85), 2018, doi:10.1016/j.jsc.2017.07.010

11.21.3.97 Danilevski()

```
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)
```

11.21.3.98 buildMatrix()

```
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)
```

Bug is this :

11.21.3.99 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]
```

11.21.3.100 CharPoly() [5/8]

```
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)  [inline]
```

11.21.3.101 Det() [3/6]

```
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)  [inline]
```

11.21.3.102 Det() [4/6]

```
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)  [inline]
```

11.21.3.103 fsytrf_BC_Crout()

```
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)  [inline]
```

11.21.3.104 fsytrf_BC_RL()

```
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)  [inline]
```

11.21.3.105 fsytrf_UP_RPM_BC_RL()

```
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)  [inline]
```

11.21.3.106 fsytrf_LOW_RPM_BC_Crout()

```
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)  [inline]
```

11.21.3.107 fsytrf_UP_RPM_BC_Crout()

```
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)  [inline]
```

11.21.3.108 fsytrf_UP_RPM()

```
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)  [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

11.21.3.109 fsytrf_nonunit() [2/3]

```
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)  [inline]
```

11.21.3.110 fsytrf_nonunit() [3/3]

```
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)  [inline]
```

11.21.3.111 fsytrf_RPM()

```
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)  [inline]
```

11.21.3.112 getTridiagonal()

```
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)  [inline]
```

11.21.3.113 LUdivine_gauss() [1/2]

```
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)  [inline]
```

11.21.3.114 LUdivine_small() [1/2]

```
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)  [inline]
```

11.21.3.115 LUdivine() [2/4]

```
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)  [inline]
```

Todo std::swap ?

11.21.3.116 LUdivine() [3/4]

```
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)  [inline]
```

11.21.3.117 MonotonicCompress()

```
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)  [inline]
```

11.21.3.118 MonotonicCompressMorePivots()

```
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)  [inline]
```

11.21.3.119 MonotonicCompressCycles()

```
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)  [inline]
```

11.21.3.120 MonotonicExpand()

```
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)
```

11.21.3.121 applyP_block()

```
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)  [inline]
```

11.21.3.122 doApplyS()

```
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)  [inline]
```

11.21.3.123 MatrixApplyS() [1/3]

```
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) [inline]
```

11.21.3.124 MatrixApplyS() [2/3]

```
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) [inline]
```

11.21.3.125 MatrixApplyS() [3/3]

```
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) [inline]
```

11.21.3.126 PermApplyS()

```
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) [inline]
```

11.21.3.127 doApplyT()

```
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)  [inline]
```

11.21.3.128 MatrixApplyT() [1/3]

```
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)  [inline]
```

11.21.3.129 MatrixApplyT() [2/3]

```
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)  [inline]
```

11.21.3.130 MatrixApplyT() [3/3]

```
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)  [inline]
```

11.21.3.131 PermApplyT()

```
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)  [inline]
```

11.21.3.132 composePermutationsLLL()

```
void composePermutationsLLL (
            size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N)  [inline]
```

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.

Parameters

in,out	*P1*	a LAPACK permutation of size N
	P2	a LAPACK permutation of size N-R

11.21.3.133 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 x Diag (I_R, P2) where P1 is a LAPACK and P2 a LAPACK permutation and store the result in MathP as a MathPermutation format.

Parameters

out		

a MathPermutation of size N

Parameters

P1	a LAPACK permutation of size N
P2	a LAPACK permutation of size N-R

11.21.3.134 composePermutationsMLM()

```
void composePermutationsMLM (
            size_t * MathP1,
            const size_t * P2,
            const size_t R,
            const size_t N)  [inline]
```

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.

Parameters

in,out	*MathP1*	a MathPermutation of size N
	P2	a LAPACK permutation of size N-R

11.21.3.135 cyclic_shift_mathPerm()

```
void cyclic_shift_mathPerm (
            size_t * P,
            const size_t s)  [inline]
```

11.21.3.136 cyclic_shift_row_col() [1/2]

```
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)  [inline]
```

11.21.3.137 cyclic_shift_row() [1/3]

```
template<class Field>
void cyclic_shift_row (
            const Field & F,
            typename Field::Element_ptr A,
            size_t m,
            size_t n,
            size_t lda)  [inline]
```

11.21.3.138 cyclic_shift_row() [2/3]

```
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)  [inline]
```

11.21.3.139 cyclic_shift_col() [1/3]

```
template<class Field>
void cyclic_shift_col (
            const Field & F,
            typename Field::Element_ptr A,
            size_t m,
            size_t n,
            size_t lda)  [inline]
```

11.21.3.140 cyclic_shift_col() [2/3]

```
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)  [inline]
```

11.21.3.141 PLUQ_basecaseV3()

```
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)  [inline]
```

11.21.3.142 PLUQ_basecaseV2()

```
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)  [inline]
```

11.21.3.143 PLUQ_basecaseCrout()

```
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)  [inline]
```

11.21.3.144 _PLUQ()

```
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)  [inline]
```

11.21.3.145 PLUQ() [4/6]

```
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)  [inline]
```

11.21.3.146 threads_fgemm()

```
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)
```

11.21.3.147 threads_ftrsm()

```
template<class Field>
void threads_ftrsm (
            const size_t m,
            const size_t n,
            int nbthreads,
            size_t * t1,
            size_t * t2)
```

11.21.3.148 PLUQ() [5/6]

```
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:
> & PSHelper)  [inline]
```

11.21.3.149 fflas_const_cast() [1/3]

```
template<>
rns_double_elt_ptr fflas_const_cast (
            rns_double_elt_cstptr x)  [inline]
```

11.21.3.150 fflas_const_cast() [2/3]

```
template<>
rns_double_elt_cstptr fflas_const_cast (
            rns_double_elt_ptr x)  [inline]
```

11.21.3.151 cyclic_shift_row_col() [2/2]

```
template<typename Base_t>
void cyclic_shift_row_col (
            Base_t * A,
            size_t m,
            size_t n,
            size_t lda)
```

11.21.3.152 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)
```

11.21.3.153 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)
```

11.21.3.154 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)
```

11.21.3.155 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)
```

11.21.3.156 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)
```

11.21.3.157 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)
```

11.21.3.158 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)
```

11.21.3.159 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)
```

11.21.3.160 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)
```

11.21.3.161 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)
```

11.21.3.162 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)
```

11.21.3.163 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)
```

11.21.3.164 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)
```

11.21.3.165 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)
```

11.21.3.166 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)
```

11.21.3.167 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)
```

11.21.3.168 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)
```

11.21.3.169 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)
```

11.21.3.170 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)
```

11.21.3.171 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)
```

11.21.3.172 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)
```

11.21.3.173 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)
```

11.21.3.174 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)
```

11.21.3.175 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)
```

11.21.3.176 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)
```

11.21.3.177 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)
```

11.21.3.178 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)
```

11.21.3.179 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)
```

11.21.3.180 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)
```

11.21.3.181 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)
```

11.21.3.182 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)
```

11.21.3.183 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)
```

11.21.3.184 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)
```

11.21.3.185 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)
```

11.21.3.186 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)
```

11.21.3.187 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)
```

11.21.3.188 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)
```

11.21.3.189 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)
```

11.21.3.190 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)
```

11.21.3.191 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)
```

11.21.3.192 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)
```

11.21.3.193 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)
```

11.21.3.194 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)
```

11.21.3.195 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)
```

11.21.3.196 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)
```

11.21.3.197 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)
```

11.21.3.198 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)
```

11.21.3.199 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)
```

11.21.3.200 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)
```

11.21.3.201 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)
```

11.21.3.202 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)
```

11.21.3.203 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)
```

11.21.3.204 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)
```

11.21.3.205 fflas_const_cast() [3/3]

```
template<class T, class CT = const T>
T fflas_const_cast (
            CT x)
```

11.21.3.206 failure()

```
Failure & failure ()  [inline]
```

11.21.3.207 isOdd() [1/3]

```
template<class T>
bool isOdd (
            const T & a)  [inline]
```

11.21.3.208 isOdd() [2/3]

```
bool isOdd (
            const float & a)  [inline]
```

11.21.3.209 isOdd() [3/3]

```
bool isOdd (
            const double & a)  [inline]
```

11.21.3.210 NonZeroRandomMatrix() [1/2]

```
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) [inline]
```

Random non-zero Matrix.

Creates a `m` x `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` x `lda` field elements)
	lda	leading dimension of `A`
	G	a random iterator

Returns

`A`.

11.21.3.211 NonZeroRandomMatrix() [2/2]

```
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) [inline]
```

Random non-zero Matrix.

Creates a `m` x `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` x `lda` field elements)
	lda	leading dimension of `A`

Returns

`A`.

11.21.3.212 RandomMatrix() [1/2]

```
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)  [inline]
```

Random Matrix.

Creates a m x 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 x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

11.21.3.213 RandomMatrix() [2/2]

```
template<class Field>
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 x 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 x lda field elements)
	lda	leading dimension of A

Returns

A.

11.21.3.214 RandomTriangularMatrix() [1/2]

```
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)  [inline]
```

Random Triangular Matrix.

Creates a `m` x `n` triangular matrix with random entries. The `UpLo` parameter defines wether 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` x `lda` field elements)
	lda	leading dimension of `A`
	G	a random iterator

Returns

`A`.

11.21.3.215 RandomTriangularMatrix() [2/2]

```
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)  [inline]
```

Random Triangular Matrix.

Creates a `m` x `n` triangular matrix with random entries. The `UpLo` parameter defines wether 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 x lda field elements)
	lda	leading dimension of A

Returns

A.

11.21.3.216 RandInt()

```
size_t RandInt (
            size_t a,
            size_t b)  [inline]
```

11.21.3.217 RandomSymmetricMatrix()

```
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)  [inline]
```

Random Symmetric Matrix.

Creates a m x n triangular matrix with random entries. The UpLo parameter defines wether it is upper or lower triangular.

Parameters

	F	field
	n	order of A
out	*A*	the matrix (preallocated to at least n x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

11.21.3.218 RandomMatrixWithRank() [1/2]

```
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)  [inline]
```

Random Matrix with prescribed rank.

Creates an m x 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 x lda field elements)
lda	leading dimension of A
G	a random iterator

Returns

A.

11.21.3.219 RandomMatrixWithRank() [2/2]

```
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)  [inline]
```

Random Matrix with prescribed rank.

Creates an m x 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 x lda field elements)
	lda	leading dimension of A

Returns

A.

11.21.3.220 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 $\{0,\ldots,N-1\}$ 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)

11.21.3.221 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)

11.21.3.222 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 x N : a matrix with only R non-zeros equal to one, in a random rook placement.

Parameters

	M	row dimension
	N	column dimension
out	*rows*	the row position of each non zero element (pre-allocated)
out	*cols*	the column position of each non zero element (pre-allocated)

11.21.3.223 swapval()

```
void swapval (
            size_t k,
            size_t N,
            size_t * P,
            size_t val)  [inline]
```

11.21.3.224 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 x N : a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.

Parameters

	N	matrix order
out	*rows*	the row position of each non zero element (pre-allocated)
out	*cols*	the column position of each non zero element (pre-allocated)

11.21.3.225 RandomLTQSRankProfileMatrix()

```
void RandomLTQSRankProfileMatrix (
            size_t n,
            size_t r,
            size_t t,
            size_t * rows,
            size_t * cols)  [inline]
```

11.21.3.226 RandomMatrixWithRankandRPM() [1/2]

```
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)  [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an m x 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 x 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.

11.21.3.227 RandomMatrixWithRankandRPM() [2/2]

```
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) [inline]
```

Random Matrix with prescribed rank and rank profile matrix Creates an m x 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 x 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

Returns

A.

11.21.3.228 RandomSymmetricMatrixWithRankandRPM() [1/2]

```
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) [inline]
```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an `n` x `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` x `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`.

11.21.3.229 RandomSymmetricMatrixWithRankandRPM() [2/2]

```
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) [inline]
```

Random Symmetric Matrix with prescribed rank and rank profile matrix Creates an `n` x `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` x `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

Returns

A.

11.21.3.230 RandomMatrixWithRankandRandomRPM() [1/2]

```
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) [inline]
```

Random Matrix with prescribed rank, with random rank profile matrix Creates an m x 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 x lda field elements)
lda	leading dimension of A
G	a random iterator

Returns

A.

11.21.3.231 RandomMatrixWithRankandRandomRPM() [2/2]

```
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) [inline]
```

Random Matrix with prescribed rank, with random rank profile matrix Creates an m x 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 x lda field elements)
lda	leading dimension of A

Returns

A.

11.21.3.232 RandomSymmetricMatrixWithRankandRandomRPM() [1/2]

```
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)  [inline]
```

Random Symmetric Matrix with prescribed rank, with random rank profile matrix Creates an n x 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 x lda field elements)
lda	leading dimension of A
G	a random iterator

Returns

A.

11.21.3.233 RandomSymmetricMatrixWithRankandRandomRPM() [2/2]

```
template<class Field>
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 x 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` x `lda` field elements)
lda	leading dimension of `A`

Returns

`A`.

11.21.3.234 RandomMatrixWithDet() [1/2]

```
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)  [inline]
```

Random Matrix with prescribed det.

Creates a `m` x `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` x `lda` field elements)
lda	leading dimension of `A`

Returns

`A`.

11.21.3.235 RandomMatrixWithDet() [2/2]

```
template<class Field, class RandIter>
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` x `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` x `lda` field elements)
lda	leading dimension of `A`

Returns

`A`.

11.21.3.236 RandomLTQSMatrixWithRankandQSorder()

```
template<class Field, class RandIter>
Field::Element_ptr RandomLTQSMatrixWithRankandQSorder (
            Field & F,
            size_t n,
            size_t r,
            size_t t,
            typename Field::Element_ptr A,
            size_t lda,
            RandIter & G)  [inline]
```

11.21.3.237 chooseField()

```
template<typename Field>
Field * chooseField (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed)
```

11.21.3.238 chooseField< Givaro::ZRing< int32_t > >()

```
template<>
Givaro::ZRing< int32_t > * chooseField< Givaro::ZRing< int32_t > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed)
```

11.21.3.239 chooseField< Givaro::ZRing< int64_t > >()

```
template<>
Givaro::ZRing< int64_t > * chooseField< Givaro::ZRing< int64_t > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed)
```

11.21.3.240 chooseField< Givaro::ZRing< float > >()

```
template<>
Givaro::ZRing< float > * chooseField< Givaro::ZRing< float > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed)
```

11.21.3.241 chooseField< Givaro::ZRing< double > >()

```
template<>
Givaro::ZRing< double > * chooseField< Givaro::ZRing< double > > (
            Givaro::Integer q,
            uint64_t b,
            uint64_t seed)
```

11.22 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, typename 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, 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 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, typename 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 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 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>
  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)

11.22.1 Function Documentation

11.22.1.1 LUdivine_construct() [1/2]

```
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)
```

11.22.1.2 GaussJordan()

```
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) [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

| I | A11 | A12 | | |—|-—|-—|—| | | |I | ∗| A22 | | | |0 | 0| A22 | | |—|-—|-—|—| | | 0 | A32 | | |—|-—|-—|—|
where the transformation matrix is stored at the pivot column position

11.22.1.3 KellerGehrig()

```
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)
```

11.22.1.4 KGFast()

```
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)
```

11.22.1.5 KGFast_generalized()

```
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)
```

11.22.1.6 fgemv_kgf()

```
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)
```

11.22.1.7 LUKrylov()

```
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)
```

11.22.1.8 Danilevski()

```
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)
```

11.22.1.9 RandomKrylovPrecond()

```
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)  [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

11.22.1.10 ArithProg()

```
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)  [inline]
```

11.22.1.11 LUKrylov_KGFast()

```
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)
```

11.22.1.12 MatVecMinPoly()

```
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) [inline]
```

11.22.1.13 Hybrid_KGF_LUK_MinPoly()

```
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)
```

11.22.1.14 updateD()

```
template<class Field>
size_t updateD (
            const Field & F,
            size_t * d,
            size_t k,
            std::vector< std::vector< typename Field::Element > > & minpt)
```

11.22.1.15 newD()

```
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)
```

11.22.1.16 CompressRows()

```
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)  [inline]
```

11.22.1.17 CompressRowsQK()

```
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)  [inline]
```

11.22.1.18 DeCompressRows()

```
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)  [inline]
```

11.22.1.19 DeCompressRowsQK()

```
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)  [inline]
```

11.22.1.20 CompressRowsQA()

```
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)  [inline]
```

11.22.1.21 DeCompressRowsQA()

```
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)  [inline]
```

11.22.1.22 LUdivine_construct() [2/2]

```
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)
```

11.23 Givaro Namespace Reference

Data Structures

- class ModularBalanced
- class Montgomery

11.24 MKL_CONFIG Namespace Reference

11.25 RecInt Namespace Reference

Data Structures

- class rint
- class ruint

Chapter 12

Data Structure Documentation

12.1 AlgoChooser< ModeT, ParSeq > Struct Template Reference

Public Types

- typedef MMHelperAlgo::Winograd value

12.1.1 Member Typedef Documentation

12.1.1.1 value

```
template<class ModeT, class ParSeq>
typedef MMHelperAlgo::Winograd value
```

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.2 AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq > Struct Template Reference

Public Types

- typedef MMHelperAlgo::Classic value
- typedef MMHelperAlgo::Winograd value

12.2.1 Member Typedef Documentation

12.2.1.1 value [1/2]

```
template<class ParSeq>
typedef MMHelperAlgo::Classic value
```

12.2.1.2 value [2/2]

```
typedef MMHelperAlgo::Winograd value
```

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.3 ALL< v > Struct Template Reference

The documentation for this struct was generated from the following file:

- test-simd.C

12.4 ALL< false, v... > Struct Template Reference

Static Public Attributes

- static constexpr bool value = false

12.4.1 Field Documentation

12.4.1.1 value

```
template<bool... v>
bool value = false [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.5 ALL< true, v... > Struct Template Reference

Static Public Attributes

- static constexpr bool value = ALL<v...>::value

12.5.1 Field Documentation

12.5.1.1 value

```
template<bool... v>
bool value = ALL<v...>::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.6 ALL<> Struct Reference

Static Public Attributes

- static constexpr bool value = true

12.6.1 Field Documentation

12.6.1.1 value

```
bool value = true [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.7 ArbitraryPrecIntTag Struct Reference

Arbitrary precision integers: GMP.

```
#include <field-traits.h>
```

12.7.1 Detailed Description

Arbitrary precision integers: GMP.

The documentation for this struct was generated from the following file:

- field-traits.h

12.8 AreEqual< X, Y > Class Template Reference

```
#include <fflas_enum.h>
```

Static Public Attributes

- static const bool value = false

12.8.1 Field Documentation

12.8.1.1 value

```
template<class X, class Y>
const bool value = false [static]
```

The documentation for this class was generated from the following file:

- fflas_enum.h

12.9 AreEqual< X, X > Class Template Reference

```
#include <fflas_enum.h>
```

Static Public Attributes

- static const bool value = true
- static const bool value

12.9.1 Field Documentation

12.9.1.1 value [1/2]

```
template<class X>
const bool value = true [static]
```

12.9.1.2 value [2/2]

```
const bool value [static]
```

The documentation for this class was generated from the following file:

- fflas_enum.h

12.10 Argument Struct Reference

```
#include <args-parser.h>
```

Data Fields

- char c
- const char ∗ example
- const char ∗ helpString
- ArgumentType type
- void ∗ data

12.10.1 Field Documentation

12.10.1.1 c

```
char c
```

12.10.1.2 example

`const char* example`

12.10.1.3 helpString

`const char* helpString`

12.10.1.4 type

`ArgumentType type`

12.10.1.5 data

`void* data`

The documentation for this struct was generated from the following file:

- args-parser.h

12.11 array< T > Class Template Reference

STL class.

Data Structures

- class const_iterator

  STL iterator class.
- class const_reverse_iterator

  STL iterator class.
- class iterator

  STL iterator class.
- class reverse_iterator

  STL iterator class.

Data Fields

- T elements

  STL member.

12.11.1 Detailed Description

STL class.

12.11.2 Field Documentation

12.11.2.1 elements

`T elements`

STL member.

The documentation for this class was generated from the following files:

12.12 associatedDelayedField< Field > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef Field field
- typedef Field & type

12.12.1 Member Typedef Documentation

12.12.1.1 field

```
template<class Field>
typedef Field field
```

12.12.1.2 type

```
template<class Field>
typedef Field& type
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.13 associatedDelayedField< const FFPACK::RNSIntegerMod< RNS > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FFPACK::RNSInteger< RNS > field
- typedef FFPACK::RNSInteger< RNS > type
- typedef const FFPACK::RNSIntegerMod< RNS > field
- typedef const FFPACK::RNSIntegerMod< RNS > & type

12.13.1 Member Typedef Documentation

12.13.1.1 field [1/2]

```
template<typename RNS>
typedef FFPACK::RNSInteger<RNS> field
```

12.13.1.2 type [1/2]

```
template<typename RNS>
typedef FFPACK::RNSInteger<RNS> type
```

12.13.1.3 field [2/2]

```
typedef const FFPACK::RNSIntegerMod< RNS > field
```

12.13.1.4 type [2/2]

```
typedef const FFPACK::RNSIntegerMod< RNS >& type
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.14 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
- typedef const Givaro::Modular< T, X > field
- typedef const Givaro::Modular< T, X > & type

12.14.1 Member Typedef Documentation

12.14.1.1 field [1/2]

```
template<typename T, typename X>
typedef Givaro::ZRing<T> field
```

12.14.1.2 type [1/2]

```
template<typename T, typename X>
typedef Givaro::ZRing<T> type
```

12.14.1.3 field [2/2]

```
typedef const Givaro::Modular< T, X > field
```

12.14.1.4 type [2/2]

```
typedef const Givaro::Modular< T, X >& type
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.15 associatedDelayedField< const Givaro::ModularBalanced< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Givaro::ZRing< T > field
- typedef Givaro::ZRing< T > type
- typedef const Givaro::ModularBalanced< T > field
- typedef const Givaro::ModularBalanced< T > & type

12.15.1 Member Typedef Documentation

12.15.1.1 field [1/2]

```
template<typename T>
typedef Givaro::ZRing<T> field
```

12.15.1.2 type [1/2]

```
template<typename T>
typedef Givaro::ZRing<T> type
```

12.15.1.3 field [2/2]

```
typedef const Givaro::ModularBalanced< T > field
```

12.15.1.4 type [2/2]

```
typedef const Givaro::ModularBalanced< T >& type
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.16 associatedDelayedField< const Givaro::ZRing< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef Givaro::ZRing< T > field
- typedef Givaro::ZRing< T > type
- typedef const Givaro::ZRing< T > field
- typedef const Givaro::ZRing< T > & type

12.16.1 Member Typedef Documentation

12.16.1.1 field [1/2]

```
template<typename T>
typedef Givaro::ZRing<T> field
```

12.16.1.2 type [1/2]

```
template<typename T>
typedef Givaro::ZRing<T> type
```

12.16.1.3 field [2/2]

```
typedef const Givaro::ZRing< T > field
```

12.16.1.4 type [2/2]

```
typedef const Givaro::ZRing< T >& type
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.17 Auto Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.18 Bench< Elt > Class Template Reference

Public Types

- using Field = Modular<Elt>
- using Elt_ptr = typename Field::Element_ptr
- using Residu = typename Field::Residu_t
- template<bool B, class T = void>
  using enable_if_t = typename std::enable_if<B, T>::type

- template<typename Simd>
  using is_same_element = typename Simd::template is_same_element<Field>
- template<typename E>
  using enable_if_no_simd_t = enable_if_t<Simd<E>::vect_size == 1>
- template<typename E>
  using enable_if_simd128_t = enable_if_t<sizeof(E)∗Simd<E>::vect_size == 16>
- template<typename E>
  using enable_if_simd256_t = enable_if_t<sizeof(E)∗Simd<E>::vect_size == 32>
- template<typename E>
  using enable_if_simd512_t = enable_if_t<sizeof(E)∗Simd<E>::vect_size == 64>

Public Member Functions

- Bench (size_t m, size_t n, size_t iters, bool inplace)
- template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > ∗ = nullptr>
  void doBenchs ()
- template<typename _E = Elt, enable_if_t< is_same< _E, Elt >::value > ∗ = nullptr, enable_if_no_simd_t< _E > ∗ = nullptr>
  void run (bool allsimd)

Static Public Member Functions

- template<typename _E = Elt, enable_if_t<!is_same< _E, Givaro::Integer >::value > ∗ = nullptr>
  static Residu cardinality ()
- template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > ∗ = nullptr>
  static Residu cardinality ()

Protected Attributes

- Field F
- const size_t m
- const size_t n
- const size_t iters
- const bool inplace

12.18.1 Member Typedef Documentation

12.18.1.1 Field

```
template<typename Elt>
using Field = Modular<Elt>
```

12.18.1.2 Elt_ptr

```
template<typename Elt>
using Elt_ptr = typename Field::Element_ptr
```

12.18.1.3 Residu

```
template<typename Elt>
using Residu = typename Field::Residu_t
```

12.18.1.4 enable_if_t

```
template<typename Elt>
template<bool B, class T = void>
using enable_if_t = typename std::enable_if<B, T>::type
```

12.18.1.5 is_same_element

```
template<typename Elt>
template<typename Simd>
using is_same_element = typename Simd::template is_same_element<Field>
```

12.18.1.6 enable_if_no_simd_t

```
template<typename Elt>
template<typename E>
using enable_if_no_simd_t = enable_if_t<Simd<E>::vect_size == 1>
```

12.18.1.7 enable_if_simd128_t

```
template<typename Elt>
template<typename E>
using enable_if_simd128_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 16>
```

12.18.1.8 enable_if_simd256_t

```
template<typename Elt>
template<typename E>
using enable_if_simd256_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 32>
```

12.18.1.9 enable_if_simd512_t

```
template<typename Elt>
template<typename E>
using enable_if_simd512_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 64>
```

12.18.2 Constructor & Destructor Documentation

12.18.2.1 Bench()

```
template<typename Elt>
Bench (
            size_t m,
            size_t n,
            size_t iters,
            bool inplace)   [inline]
```

12.18.3 Member Function Documentation

12.18.3.1 cardinality() [1/2]

```
template<typename Elt>
template<typename _E = Elt, enable_if_t<!is_same< _E, Givaro::Integer >::value > * = nullptr>
static Residu cardinality ()  [inline], [static]
```

12.18.3.2 cardinality() [2/2]

```
template<typename Elt>
template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > * = nullptr>
static Residu cardinality ()  [inline], [static]
```

12.18.3.3 doBenchs()

```
template<typename Elt>
template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > * =
nullptr>
void doBenchs ()  [inline]
```

12.18.3.4 run()

```
template<typename Elt>
template<typename _E = Elt, enable_if_t< is_same< _E, Elt >::value > * = nullptr, enable_if_no_simd_t<
_E > * = nullptr>
void run (
            bool allsimd) [inline]
```

12.18.4 Field Documentation

12.18.4.1 F

```
template<typename Elt>
Field F [protected]
```

12.18.4.2 m

```
template<typename Elt>
const size_t m [protected]
```

12.18.4.3 n

```
template<typename Elt>
const size_t n [protected]
```

12.18.4.4 iters

```
template<typename Elt>
const size_t iters [protected]
```

12.18.4.5 inplace

```
template<typename Elt>
const bool inplace [protected]
```

The documentation for this class was generated from the following file:

- benchmark-storage-transpose.C

12.19 Bini Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.20 Block Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.21 BlockTransposeSIMD< Field, Simd, > Struct Template Reference

```
#include <fflas_transpose.h>
```

Public Member Functions

- template<size_t _s = Simd::vect_size, IsSimdSize< _s, 1 > * = nullptr>
  void transpose (const Field &F, ConstElement_ptr A, size_t lda, Element_ptr B, size_t ldb) const
- template<size_t _s = Simd::vect_size, IsSimdSize< _s, 2 > * = nullptr>
  void transpose (const Field &F, ConstElement_ptr A, size_t lda, Element_ptr B, size_t ldb) const
- template<size_t _s = Simd::vect_size, IsSimdSize< _s, 4 > * = nullptr>
  void transpose (const Field &F, ConstElement_ptr A, size_t lda, Element_ptr B, size_t ldb) const
- template<size_t _s = Simd::vect_size, IsSimdSize< _s, 8 > * = nullptr>
  void transpose (const Field &F, ConstElement_ptr A, size_t lda, Element_ptr B, size_t ldb) const
- template<size_t _s = Simd::vect_size, IsSimdSize< _s, 16 > * = nullptr>
  void transpose (const Field &F, ConstElement_ptr A, size_t lda, Element_ptr B, size_t ldb) const

Static Public Member Functions

- static constexpr size_t size ()
- static const std::string info ()

12.21.1 Member Function Documentation

12.21.1.1 size()

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_↩
element< Field >::value >::type * = nullptr>
static constexpr size_t size ()  [inline], [static], [constexpr]
```

12.21.1.2 info()

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_↩
element< Field >::value >::type * = nullptr>
static const std::string info ()  [inline], [static]
```

12.21.1.3 transpose() [1/5]

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_↩
element< Field >::value >::type * = nullptr>
template<size_t _s = Simd::vect_size, IsSimdSize< _s, 1 > * = nullptr>
void transpose (
            const Field & F,
            ConstElement_ptr A,
            size_t lda,
            Element_ptr B,
            size_t ldb) const  [inline]
```

12.21.1.4 transpose() [2/5]

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_↩
element< Field >::value >::type * = nullptr>
template<size_t _s = Simd::vect_size, IsSimdSize< _s, 2 > * = nullptr>
void transpose (
            const Field & F,
            ConstElement_ptr A,
            size_t lda,
            Element_ptr B,
            size_t ldb) const  [inline]
```

12.21.1.5 transpose() [3/5]

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_←
element< Field >::value >::type * = nullptr>
template<size_t _s = Simd::vect_size, IsSimdSize< _s, 4 > * = nullptr>
void transpose (
            const Field & F,
            ConstElement_ptr A,
            size_t lda,
            Element_ptr B,
            size_t ldb) const  [inline]
```

12.21.1.6 transpose() [4/5]

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_←
element< Field >::value >::type * = nullptr>
template<size_t _s = Simd::vect_size, IsSimdSize< _s, 8 > * = nullptr>
void transpose (
            const Field & F,
            ConstElement_ptr A,
            size_t lda,
            Element_ptr B,
            size_t ldb) const  [inline]
```

12.21.1.7 transpose() [5/5]

```
template<typename Field, typename Simd, typename std::enable_if< Simd::template is_same_←
element< Field >::value >::type * = nullptr>
template<size_t _s = Simd::vect_size, IsSimdSize< _s, 16 > * = nullptr>
void transpose (
            const Field & F,
            ConstElement_ptr A,
            size_t lda,
            Element_ptr B,
            size_t ldb) const  [inline]
```

The documentation for this struct was generated from the following file:

- fflas_transpose.h

12.22 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)

12.22.1 Member Function Documentation

12.22.1.1 operator()()

```
template<class Element>
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)  [inline]
```

The documentation for this class was generated from the following file:

- ffpack_ludivine.inl

12.23 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)
- 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)

12.23.1 Member Function Documentation

12.23.1.1 operator()() [1/2]

```
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)  [inline]
```

12.23.1.2 operator()() [2/2]

```
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

12.24 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)
- 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)

12.24.1 Member Function Documentation

12.24.1.1 operator()() [1/2]

```
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)  [inline]
```

12.24.1.2 operator()() [2/2]

```
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

12.25 CharpolyFailed Class Reference

`#include <ffpack.h>`

The documentation for this class was generated from the following file:

- ffpack.h

12.26 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)

12.26.1 Constructor & Destructor Documentation

12.26.1.1 Checker_Empty()

```
template<class Field>
template<typename...  Params>
Checker_Empty (
             Params...  parameters)  [inline]
```

12.26.2 Member Function Documentation

12.26.2.1 check()

```
template<class Field>
template<typename...  Params>
bool check (
             Params...  parameters)  [inline]
```

The documentation for this struct was generated from the following file:

- checker_empty.h

12.27 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)

12.27.1 Constructor & Destructor Documentation

12.27.1.1 CheckerImplem_charpoly() [1/2]

```
template<class Field, class Polynomial>
CheckerImplem_charpoly (
             const Field & F_,
             const size_t n_,
             typename Field::ConstElement_ptr A,
             size_t lda_)  [inline]
```

12.27.1.2 CheckerImplem_charpoly() [2/2]

```
template<class Field, class Polynomial>
CheckerImplem_charpoly (
             typename Field::RandIter & G,
             const size_t n_,
             typename Field::ConstElement_ptr A,
             size_t lda_)  [inline]
```

12.27.1.3 ∼CheckerImplem_charpoly()

```
template<class Field, class Polynomial>
∼CheckerImplem_charpoly () [inline]
```

12.27.2 Member Function Documentation

12.27.2.1 check()

```
template<class Field, class Polynomial>
bool check (
            Polynomial & g) [inline]
```

The documentation for this class was generated from the following file:

- checker_charpoly.inl

12.28 CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial > Class Template Reference

Public Types

- typedef Givaro::ZRing< Givaro::Integer > Ring

Public Member Functions

- CheckerImplem_charpoly (const Ring &F_, const size_t n_, typename Ring::ConstElement_ptr A, size_t lda_)
- CheckerImplem_charpoly (typename Ring::RandIter &G, const size_t n_, typename Ring::ConstElement_ptr A, size_t lda_)
- ∼CheckerImplem_charpoly ()
- bool check (Polynomial &g)
- CheckerImplem_charpoly (const Givaro::ZRing< Givaro::Integer > &F_, const size_t n_, typename Givaro::ZRing< Givaro::Integer >::ConstElement_ptr A, size_t lda_)
- CheckerImplem_charpoly (typename Givaro::ZRing< Givaro::Integer >::RandIter &G, const size_t n_, typename Givaro::ZRing< Givaro::Integer >::ConstElement_ptr A, size_t lda_)
- ∼CheckerImplem_charpoly ()
- bool check (Polynomial &g)

12.28.1 Member Typedef Documentation

12.28.1.1 Ring

```
template<class Polynomial>
typedef Givaro::ZRing<Givaro::Integer> Ring
```

12.28.2 Constructor & Destructor Documentation

12.28.2.1 CheckerImplem_charpoly() [1/4]

```
template<class Polynomial>
CheckerImplem_charpoly (
            const Ring & F_,
            const size_t n_,
            typename Ring::ConstElement_ptr A,
            size_t lda_) [inline]
```

12.28.2.2 CheckerImplem_charpoly() [2/4]

```
template<class Polynomial>
CheckerImplem_charpoly (
            typename Ring::RandIter & G,
            const size_t n_,
            typename Ring::ConstElement_ptr A,
            size_t lda_) [inline]
```

12.28.2.3 ∼CheckerImplem_charpoly() [1/2]

```
template<class Polynomial>
~CheckerImplem_charpoly () [inline]
```

12.28.2.4 CheckerImplem_charpoly() [3/4]

```
CheckerImplem_charpoly (
            const Givaro::ZRing< Givaro::Integer > & F_,
            const size_t n_,
            typename Givaro::ZRing< Givaro::Integer >::ConstElement_ptr A,
            size_t lda_) [inline]
```

12.28.2.5 CheckerImplem_charpoly() [4/4]

```
CheckerImplem_charpoly (
            typename Givaro::ZRing< Givaro::Integer >::RandIter & G,
            const size_t n_,
            typename Givaro::ZRing< Givaro::Integer >::ConstElement_ptr A,
            size_t lda_) [inline]
```

12.28.2.6 ∼CheckerImplem_charpoly() [2/2]

```
~CheckerImplem_charpoly () [inline]
```

12.28.3 Member Function Documentation

12.28.3.1 check() [1/2]

```
template<class Polynomial>
bool check (
            Polynomial & g) [inline]
```

12.28.3.2 check() [2/2]

```
bool check (
            Polynomial & g) [inline]
```

The documentation for this class was generated from the following file:

- checker_charpoly.inl

12.29 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.

12.29.1 Constructor & Destructor Documentation

12.29.1.1 CheckerImplem_Det() [1/2]

```
template<class Field>
CheckerImplem_Det (
            const Field & F_,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda)  [inline]
```

12.29.1.2 CheckerImplem_Det() [2/2]

```
template<class Field>
CheckerImplem_Det (
            typename Field::RandIter & G,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda)  [inline]
```

12.29.1.3 ∼CheckerImplem_Det()

```
template<class Field>
∼CheckerImplem_Det ()  [inline]
```

12.29.2 Member Function Documentation

12.29.2.1 check()

```
template<class Field>
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

LU,storage	for L and U
det	
P	
Q	

The documentation for this class was generated from the following file:

- checker_det.inl

12.30 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)

12.30.1 Constructor & Destructor Documentation

12.30.1.1 CheckerImplem_fgemm() [1/2]

```
template<class Field>
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]
```

12.30.1.2 CheckerImplem_fgemm() [2/2]

```
template<class Field>
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]
```

12.30.1.3 ∼CheckerImplem_fgemm()

```
template<class Field>
~CheckerImplem_fgemm ( )  [inline]
```

12.30.2 Member Function Documentation

12.30.2.1 check()

```
template<class Field>
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

12.31 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)

12.31.1 Constructor & Destructor Documentation

12.31.1.1 CheckerImplem_ftrsm() [1/2]

```
template<class Field>
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]
```

12.31.1.2 CheckerImplem_ftrsm() [2/2]

```
template<class Field>
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]
```

12.31.1.3 ∼CheckerImplem_ftrsm()

```
template<class Field>
∼CheckerImplem_ftrsm () [inline]
```

12.31.2 Member Function Documentation

12.31.2.1 check()

```
template<class Field>
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_ftrsm.inl

12.32 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)

12.32.1 Constructor & Destructor Documentation

12.32.1.1 CheckerImplem_invert() [1/2]

```
template<class Field>
CheckerImplem_invert (
            const Field & F_,
            const size_t m_,
            typename Field::ConstElement_ptr A,
            const size_t lda_ )  [inline]
```

12.32.1.2 CheckerImplem_invert() [2/2]

```
template<class Field>
CheckerImplem_invert (
            typename Field::RandIter & G,
            const size_t m_,
            typename Field::ConstElement_ptr A,
            const size_t lda_ )  [inline]
```

12.32.1.3 ∼CheckerImplem_invert()

```
template<class Field>
~CheckerImplem_invert ( )  [inline]
```

12.32.2 Member Function Documentation

12.32.2.1 check()

```
template<class Field>
bool check (
            typename Field::ConstElement_ptr A,
            int nullity )  [inline]
```

The documentation for this class was generated from the following file:

- checker_invert.inl

12.33 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.

12.33.1 Constructor & Destructor Documentation

12.33.1.1 CheckerImplem_PLUQ() [1/2]

```
template<class Field>
CheckerImplem_PLUQ (
            const Field & F_,
            size_t m_,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda) [inline]
```

12.33.1.2 CheckerImplem_PLUQ() [2/2]

```
template<class Field>
CheckerImplem_PLUQ (
            typename Field::RandIter & G,
            size_t m_,
            size_t n_,
            typename Field::ConstElement_ptr A,
            size_t lda) [inline]
```

12.33.1.3 ∼CheckerImplem_PLUQ()

```
template<class Field>
∼CheckerImplem_PLUQ () [inline]
```

12.33.2 Member Function Documentation

12.33.2.1 check()

```
template<class Field>
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

A	
r	
P	
Q	

The documentation for this class was generated from the following file:

- checker_pluq.inl

12.34 Classic Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.35 Column Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.36 CompactElement< Element > Struct Template Reference

Public Types

- typedef Element type

12.36.1 Member Typedef Documentation

12.36.1.1 type

```
template<class Element>
typedef Element type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.37 CompactElement< double > Struct Reference

Public Types

- typedef int32_t type
- typedef double type

12.37.1 Member Typedef Documentation

12.37.1.1 type [1/2]

```
typedef int32_t type
```

12.37.1.2 type [2/2]

```
typedef double type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.38 CompactElement< float > Struct Reference

Public Types

- typedef int16_t type
- typedef float type

12.38.1 Member Typedef Documentation

12.38.1.1 type [1/2]

```
typedef int16_t type
```

12.38.1.2 type [2/2]

```
typedef float type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.39 CompactElement< int16_t > Struct Reference

Public Types

- typedef int8_t type
- typedef int16_t type

12.39.1 Member Typedef Documentation

12.39.1.1 type [1/2]

```
typedef int8_t type
```

12.39.1.2 type [2/2]

```
typedef int16_t type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.40 CompactElement< int32_t > Struct Reference

Public Types

- typedef int16_t type
- typedef int32_t type

12.40.1 Member Typedef Documentation

12.40.1.1 type [1/2]

```
typedef int16_t type
```

12.40.1.2 type [2/2]

```
typedef int32_t type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.41 CompactElement< int64_t > Struct Reference

Public Types

- typedef int32_t type
- typedef int64_t type

12.41.1 Member Typedef Documentation

12.41.1.1 type [1/2]

```
typedef int32_t type
```

12.41.1.2 type [2/2]

```
typedef int64_t type
```

The documentation for this struct was generated from the following file:

- test-io.C

12.42 compatible_data_type< Field > Struct Template Reference

Static Public Attributes

- static constexpr bool value = true

12.42.1 Field Documentation

12.42.1.1 value

```
template<typename Field>
bool value = true  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.43 compatible_data_type< Givaro::ZRing< double > > Struct Reference

Static Public Attributes

- static constexpr bool value = false
- static constexpr bool value

12.43.1 Field Documentation

12.43.1.1 value [1/2]

```
bool value = false  [static], [constexpr]
```

12.43.1.2 value [2/2]

```
bool value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.44 compatible_data_type< Givaro::ZRing< float > > Struct Reference

Static Public Attributes

- static constexpr bool value = false
- static constexpr bool value

12.44.1 Field Documentation

12.44.1.1 value [1/2]

```
bool value = false  [static], [constexpr]
```

12.44.1.2 value [2/2]

```
bool value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.45 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)

12.45.1 Constructor & Destructor Documentation

12.45.1.1 Compose() [1/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose ()  [inline]
```

12.45.1.2 Compose() [2/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
            const Compose< H1, H2 > & other)  [inline]
```

12.45.1.3 Compose() [3/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
            const Sequential & S)  [inline]
```

12.45.1.4 Compose() [4/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
            size_t th1,
            size_t th2)  [inline]
```

12.45.1.5 Compose() [5/5]

```
template<typename H1 = Sequential, typename H2 = Sequential>
Compose (
            const H1 & o1,
            const H2 & o2)  [inline]
```

12.45.2 Member Function Documentation

12.45.2.1 first_component()

```
template<typename H1 = Sequential, typename H2 = Sequential>
H1 first_component () const  [inline]
```

12.45.2.2 second_component()

```
template<typename H1 = Sequential, typename H2 = Sequential>
H2 second_component () const  [inline]
```

12.45.3 Friends And Related Symbol Documentation

12.45.3.1 operator<<

```
template<typename H1 = Sequential, typename H2 = Sequential>
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

12.46 array< T >::const_iterator Class Reference

STL iterator class.

12.46.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.47 string::const_iterator Class Reference

STL iterator class.

12.47.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.48 vector< T >::const_iterator Class Reference

STL iterator class.

12.48.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.49 array< T >::const_reverse_iterator Class Reference

STL iterator class.

12.49.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.50 string::const_reverse_iterator Class Reference

STL iterator class.

12.50.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.51 vector< T >::const_reverse_iterator Class Reference

STL iterator class.

12.51.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.52 Simd128_impl< true, true, false, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.52.1 Field Documentation

12.52.1.1 v

```
vect_t v
```

12.52.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int16.inl

12.53 Simd128_impl< true, true, false, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.53.1 Field Documentation

12.53.1.1 v

```
vect_t v
```

12.53.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int32.inl

12.54 Simd128_impl< true, true, false, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.54.1 Field Documentation

12.54.1.1 v

```
vect_t v
```

12.54.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int64.inl

12.55 Simd128_impl< true, true, true, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.55.1 Field Documentation

12.55.1.1 v

```
vect_t v
```

12.55.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int16.inl

12.56 Simd128_impl< true, true, true, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.56.1 Field Documentation

12.56.1.1 v

```
vect_t v
```

12.56.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int32.inl

12.57 Simd128_impl< true, true, true, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.57.1 Field Documentation

12.57.1.1 v

```
vect_t v
```

12.57.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd128_int64.inl

12.58 Simd256_impl< true, false, true, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.58.1 Field Documentation

12.58.1.1 v

```
vect_t v
```

12.58.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd256_double.inl

12.59 Simd256_impl< true, true, false, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.59.1 Field Documentation

12.59.1.1 v

```
vect_t v
```

12.59.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd256_int16.inl

12.60 Simd256_impl< true, true, false, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.60.1 Field Documentation

12.60.1.1 v

`vect_t v`

12.60.1.2 t

`scalar_t t`

The documentation for this union was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

12.61 Simd256_impl< true, true, false, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.61.1 Field Documentation

12.61.1.1 v

`vect_t v`

12.61.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- simd256_int64.inl

12.62 Simd256_impl< true, true, true, 2 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.62.1 Field Documentation

12.62.1.1 v

`vect_t v`

12.62.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- simd256_int16.inl

12.63 Simd256_impl< true, true, true, 4 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.63.1 Field Documentation

12.63.1.1 v

`vect_t v`

12.63.1.2 t

`scalar_t t`

The documentation for this union was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

12.64 Simd256_impl< true, true, true, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.64.1 Field Documentation

12.64.1.1 v

`vect_t v`

12.64.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- simd256_int64.inl

12.65 Simd512_impl< true, true, false, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.65.1 Field Documentation

12.65.1.1 v

`vect_t v`

12.65.1.2 t

`scalar_t t[vect_size]`

The documentation for this union was generated from the following file:

- simd512_int64.inl

12.66 Simd512_impl< true, true, true, 8 >::Converter Union Reference

Data Fields

- vect_t v
- scalar_t t [vect_size]

12.66.1 Field Documentation

12.66.1.1 v

```
vect_t v
```

12.66.1.2 t

```
scalar_t t[vect_size]
```

The documentation for this union was generated from the following file:

- simd512_int64.inl

12.67 ConvertTo< T > Struct Template Reference

Force conversion to appropriate element type of ElementCategory T.

```
#include <field-traits.h>
```

12.67.1 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

12.68 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

12.68.1 Member Typedef Documentation

12.68.1.1 Self

```
template<class ValT, class IdxT>
using Self = Coo<ValT, IdxT>
```

12.68.2 Constructor & Destructor Documentation

12.68.2.1 Coo() [1/4]

```
template<class ValT, class IdxT>
Coo (
            ValT v,
            IdxT r,
            IdxT c)  [inline]
```

12.68.2.2 Coo() [2/4]

```
template<class ValT, class IdxT>
Coo ()  [default]
```

12.68.2.3 Coo() [3/4]

```
template<class ValT, class IdxT>
Coo (
            const Self & )  [default]
```

12.68.2.4 Coo() [4/4]

```
template<class ValT, class IdxT>
Coo (
            Self && )  [default]
```

12.68.3 Member Function Documentation

12.68.3.1 operator=() [1/2]

```
template<class ValT, class IdxT>
Self & operator= (
            const Self & )  [default]
```

12.68.3.2 operator=() [2/2]

```
template<class ValT, class IdxT>
Self & operator= (
            Self && )  [default]
```

12.68.4 Field Documentation

12.68.4.1 val

```
template<class ValT, class IdxT>
ValT val = 0
```

12.68.4.2 row

```
template<class ValT, class IdxT>
IdxT row = 0
```

12.68.4.3 col

```
template<class ValT, class IdxT>
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- csr_hyb_utils.inl

12.69 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

12.69.1 Constructor & Destructor Documentation

12.69.1.1 Coo() [1/4]

```
template<class Field>
Coo ()  [default]
```

12.69.1.2 Coo() [2/4]

```
template<class Field>
Coo (
            typename Field::Element v,
            index_t r,
            index_t c)  [inline]
```

12.69.1.3 Coo() [3/4]

```
template<class Field>
Coo (
            const Self & )  [default]
```

12.69.1.4 Coo() [4/4]

```
template<class Field>
Coo (
            Self && )  [default]
```

12.69.2 Member Function Documentation

12.69.2.1 operator=() [1/2]

```
template<class Field>
Self & operator= (
            const Self & )  [default]
```

12.69.2.2 operator=() [2/2]

```
template<class Field>
Self & operator= (
            Self && )  [default]
```

12.69.3 Field Documentation

12.69.3.1 val

```
template<class Field>
Field::Element val = 0
```

12.69.3.2 col

```
template<class Field>
index_t col = 0
```

12.69.3.3 row

```
template<class Field>
index_t row = 0
```

12.69.3.4 deleted

```
template<class Field>
bool deleted = false
```

The documentation for this struct was generated from the following file:

- read_sparse.h

12.70 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

12.70.1 Member Typedef Documentation

12.70.1.1 Self

```
template<class ValT, class IdxT>
using Self = Coo<ValT, IdxT>
```

12.70.2 Constructor & Destructor Documentation

12.70.2.1 Coo() [1/4]

```
template<class ValT, class IdxT>
Coo (
            ValT v,
            IdxT r,
            IdxT c)  [inline]
```

12.70.2.2 Coo() [2/4]

```
template<class ValT, class IdxT>
Coo ()  [default]
```

12.70.2.3 Coo() [3/4]

```
template<class ValT, class IdxT>
Coo (
            const Self & )  [default]
```

12.70.2.4 Coo() [4/4]

```
template<class ValT, class IdxT>
Coo (
            Self && )  [default]
```

12.70.3 Member Function Documentation

12.70.3.1 operator=() [1/2]

```
template<class ValT, class IdxT>
Self & operator= (
            const Self & )  [default]
```

12.70.3.2 operator=() [2/2]

```
template<class ValT, class IdxT>
Self & operator= (
            Self && )  [default]
```

12.70.4 Field Documentation

12.70.4.1 val

```
template<class ValT, class IdxT>
ValT val = 0
```

12.70.4.2 row

```
template<class ValT, class IdxT>
IdxT row = 0
```

12.70.4.3 col

```
template<class ValT, class IdxT>
IdxT col = 0
```

The documentation for this struct was generated from the following file:

- sell_utils.inl

12.71 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

12.71.1 Field Documentation

12.71.1.1 _coo16

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::COO,int16_t>* _coo16 = nullptr
```

12.71.1.2 _coo32

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::COO,int32_t>* _coo32 = nullptr
```

12.71.1.3 _coo64

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::COO,int64_t>* _coo64 = nullptr
```

12.71.1.4 _coo16_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::COO_ZO,int16_t>* _coo16_zo = nullptr
```

12.71.1.5 _coo32_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::COO_ZO,int32_t>* _coo32_zo = nullptr
```

12.71.1.6 _coo64_zo

```
template<class Field>
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

12.72 count_nonconst_lvalue_reference< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- test-simd.C

12.73 count_nonconst_lvalue_reference< const T &, O... > Struct Template Reference

Static Public Attributes

- static constexpr size_t n = count_nonconst_lvalue_reference<O...>::n

12.73.1 Field Documentation

12.73.1.1 n

```
template<typename T, typename... O>
size_t n = count_nonconst_lvalue_reference<O...>::n  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.74 count_nonconst_lvalue_reference< T &, O... > Struct Template Reference

Static Public Attributes

- static constexpr size_t n

12.74.1 Field Documentation

12.74.1.1 n

```
template<typename T, typename... O>
size_t n  [static], [constexpr]
```

Initial value:

```
= std::integral_constant<size_t, 1>::value
                                                    + count_nonconst_lvalue_reference<O...>::n
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.75 count_nonconst_lvalue_reference< T, O... > Struct Template Reference

Static Public Attributes

- static constexpr size_t n = count_nonconst_lvalue_reference<O...>::n

12.75.1 Field Documentation

12.75.1.1 n

```
template<typename T, typename... O>
size_t n = count_nonconst_lvalue_reference<O...>::n  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.76 count_nonconst_lvalue_reference<> Struct Reference

Static Public Attributes

- static constexpr size_t n = std::integral_constant<size_t, 0>::value

12.76.1 Field Documentation

12.76.1.1 n

```
size_t n = std::integral_constant<size_t, 0>::value [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.77 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

12.77.1 Field Documentation

12.77.1.1 _csr16

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::CSR,int16_t>* _csr16 = nullptr
```

12.77.1.2 _csr32

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::CSR,int32_t>* _csr32 = nullptr
```

12.77.1.3 _csr64

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::CSR,int64_t>* _csr64 = nullptr
```

12.77.1.4 _csr16_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::CSR_ZO,int16_t>* _csr16_zo = nullptr
```

12.77.1.5 _csr32_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::CSR_ZO,int32_t>* _csr32_zo = nullptr
```

12.77.1.6 _csr64_zo

```
template<class Field>
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

12.78 DefaultBoundedTag Struct Reference

Use standard field operations, but keeps track of bounds on input and output.
`#include <field-traits.h>`

12.78.1 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

12.79 DefaultTag Struct Reference

No specific mode of action: use standard field operations.
`#include <field-traits.h>`

12.79.1 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

12.80 DelayedTag Struct Reference

Performs field operations with delayed mod reductions. Ensures result is reduced.
`#include <field-traits.h>`

12.80.1 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

12.81 DivideAndConquer Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.82 ElementTraits< Element > Struct Template Reference

ElementTraits.
`#include <field-traits.h>`

Public Types

- typedef ElementCategories::GenericTag value

12.82.1 Detailed Description

template< class Element >
struct FFLAS::ElementTraits< Element >

ElementTraits.

12.82.2 Member Typedef Documentation

12.82.2.1 value

```
template<class Element>
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.83 ElementTraits< double > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineFloatTag value
- typedef ElementCategories::GenericTag value

12.83.1 Member Typedef Documentation

12.83.1.1 value [1/2]

```
typedef ElementCategories::MachineFloatTag value
```

12.83.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.84 ElementTraits< FFPACK::rns_double_elt > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::RNSElementTag value
- typedef ElementCategories::GenericTag value

12.84.1 Member Typedef Documentation

12.84.1.1 value [1/2]

```
typedef ElementCategories::RNSElementTag value
```

12.84.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.85 ElementTraits< float > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineFloatTag value
- typedef ElementCategories::GenericTag value

12.85.1 Member Typedef Documentation

12.85.1.1 value [1/2]

```
typedef ElementCategories::MachineFloatTag value
```

12.85.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.86 ElementTraits< Givaro::Integer > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::ArbitraryPrecIntTag value
- typedef ElementCategories::GenericTag value

12.86.1 Member Typedef Documentation

12.86.1.1 value [1/2]

```
typedef ElementCategories::ArbitraryPrecIntTag value
```

12.86.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.87 ElementTraits< int16_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.87.1 Member Typedef Documentation

12.87.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.87.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.88 ElementTraits< int32_t > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.88.1 Member Typedef Documentation

12.88.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.88.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.89 ElementTraits< int64_t > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.89.1 Member Typedef Documentation

12.89.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.89.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.90 ElementTraits< int8_t > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.90.1 Member Typedef Documentation

12.90.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.90.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.91 ElementTraits< RecInt::rint< K > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::FixedPrecIntTag value
- typedef ElementCategories::GenericTag value

12.91.1 Member Typedef Documentation

12.91.1.1 value [1/2]

```
template<size_t K>
typedef ElementCategories::FixedPrecIntTag value
```

12.91.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.92 ElementTraits< RecInt::rmint< K, MG > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::FixedPrecIntTag value
- typedef ElementCategories::GenericTag value

12.92.1 Member Typedef Documentation

12.92.1.1 value [1/2]

```
template<size_t K, int MG>
typedef ElementCategories::FixedPrecIntTag value
```

12.92.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.93 ElementTraits< RecInt::ruint< K > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::FixedPrecIntTag value
- typedef ElementCategories::GenericTag value

12.93.1 Member Typedef Documentation

12.93.1.1 value [1/2]

```
template<size_t K>
typedef ElementCategories::FixedPrecIntTag value
```

12.93.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.94 ElementTraits< uint16_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.94.1 Member Typedef Documentation

12.94.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.94.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.95 ElementTraits< uint32_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.95.1 Member Typedef Documentation

12.95.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.95.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.96 ElementTraits< uint64_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.96.1 Member Typedef Documentation

12.96.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.96.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.97 ElementTraits< uint8_t > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ElementCategories::MachineIntTag value
- typedef ElementCategories::GenericTag value

12.97.1 Member Typedef Documentation

12.97.1.1 value [1/2]

```
typedef ElementCategories::MachineIntTag value
```

12.97.1.2 value [2/2]

```
typedef ElementCategories::GenericTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.98 EllMat< 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

12.98.1 Field Documentation

12.98.1.1 _ell16

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::ELL,int16_t>* _ell16 = nullptr
```

12.98.1.2 _ell32

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::ELL,int32_t>* _ell32 = nullptr
```

12.98.1.3 _ell64

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::ELL,int64_t>* _ell64 = nullptr
```

12.98.1.4 _ell16_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::ELL_ZO,int16_t>* _ell16_zo = nullptr
```

12.98.1.5 _ell32_zo

```
template<class Field>
FFLAS::Sparse<Field,SparseMatrix_t::ELL_ZO,int32_t>* _ell32_zo = nullptr
```

12.98.1.6 _ell64_zo

```
template<class Field>
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

12.99 Failure Class Reference

A precondtion 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

12.99.1 Detailed Description

A precondtion 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.

12.99.2 Constructor & Destructor Documentation

12.99.2.1 Failure()

```
Failure ()  [inline]
```

12.99.3 Member Function Documentation

12.99.3.1 operator()() [1/2]

```
void operator() (
            const char * function,
            int line,
            const char * check)  [inline]
```

A precondtion failed.

Parameters

function	usually `__func__`, the function that threw the error
line	usually `__LINE__`, the line where it happened
check	a string telling what failed.

12.99.3.2 operator()() [2/2]

```
void operator() (
            const char * function,
            const char * file,
            int line,
            const char * check)  [inline]
```

A precondtion 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

function	usually `__func__`, the function that threw the error
file	usually `__FILE__`, the file where this function is
line	usually `__LINE__`, the line where it happened
check	a string telling what failed.

12.99.3.3 setErrorStream()

```
void setErrorStream (
            std::ostream & stream)
```

12.99.3.4 print()

```
std::ostream & print (
            std::ostream & o) const  [inline]
```

overload the virtual print of LinboxError.

Parameters

o	output stream

12.99.4 Field Documentation

12.99.4.1 _errorStream

`std::ostream* _errorStream [protected]`

The documentation for this class was generated from the following file:

- debug.h

12.100 FailureCharpolyCheck Class Reference

`#include <checkers_ffpack.h>`

The documentation for this class was generated from the following file:

- checkers_ffpack.h

12.101 FailureDetCheck Class Reference

`#include <checkers_ffpack.h>`

The documentation for this class was generated from the following file:

- checkers_ffpack.h

12.102 FailureFgemmCheck Class Reference

`#include <checkers_fflas.h>`

The documentation for this class was generated from the following file:

- checkers_fflas.h

12.103 FailureInvertCheck Class Reference

`#include <checkers_ffpack.h>`

The documentation for this class was generated from the following file:

- checkers_ffpack.h

12.104 FailurePLUQCheck Class Reference

`#include <checkers_ffpack.h>`

The documentation for this class was generated from the following file:

- checkers_ffpack.h

12.105 FailureTrsmCheck Class Reference

`#include <checkers_fflas.h>`

The documentation for this class was generated from the following file:

- checkers_fflas.h

12.106 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

12.106.1 Member Typedef Documentation

12.106.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.106.1.2 Element

```
template<class _Field>
using Element = typename Field::Element
```

12.106.1.3 simd

```
template<class _Field>
using simd = Simd<typename _Field::Element>
```

12.106.1.4 vect_t

```
template<class _Field>
using vect_t = typename simd::vect_t
```

12.106.1.5 scalar_t

```
template<class _Field>
using scalar_t = typename simd::scalar_t
```

12.106.2 Constructor & Destructor Documentation

12.106.2.1 FieldSimd() [1/3]

```
template<class _Field>
FieldSimd (
            const Field & f)   [inline]
```

12.106.2.2 FieldSimd() [2/3]

```
template<class _Field>
FieldSimd (
            const Self & )   [default]
```

12.106.2.3 FieldSimd() [3/3]

```
template<class _Field>
FieldSimd (
            Self && )   [default]
```

12.106.3 Member Function Documentation

12.106.3.1 operator=() [1/2]

```
template<class _Field>
Self & operator= (
            const Self & )   [default]
```

12.106.3.2 operator=() [2/2]

```
template<class _Field>
Self & operator= (
            Self && )   [default]
```

12.106.3.3 init() [1/2]

```
template<class _Field>
INLINE vect_t init (
            vect_t & x,
            const vect_t a) const  [inline]
```

12.106.3.4 init() [2/2]

```
template<class _Field>
INLINE vect_t init (
            const vect_t a) const  [inline]
```

12.106.3.5 add() [1/2]

```
template<class _Field>
INLINE vect_t add (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline]
```

12.106.3.6 add() [2/2]

```
template<class _Field>
INLINE vect_t add (
            const vect_t a,
            const vect_t b)  [inline]
```

12.106.3.7 addin()

```
template<class _Field>
INLINE vect_t addin (
            vect_t & a,
            const vect_t b) const  [inline]
```

12.106.3.8 add_r() [1/2]

```
template<class _Field>
INLINE vect_t add_r (
            vect_t & c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.9 add_r() [2/2]

```
template<class _Field>
INLINE vect_t add_r (
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.10 addin_r()

```
template<class _Field>
INLINE vect_t addin_r (
            vect_t & a,
            const vect_t b) const  [inline]
```

12.106.3.11 sub() [1/2]

```
template<class _Field>
INLINE vect_t sub (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline]
```

12.106.3.12 sub() [2/2]

```
template<class _Field>
INLINE vect_t sub (
            const vect_t a,
            const vect_t b) [inline]
```

12.106.3.13 subin()

```
template<class _Field>
INLINE vect_t subin (
            vect_t & a,
            const vect_t b) const [inline]
```

12.106.3.14 sub_r() [1/2]

```
template<class _Field>
INLINE vect_t sub_r (
            vect_t & c,
            const vect_t a,
            const vect_t b) const [inline]
```

12.106.3.15 sub_r() [2/2]

```
template<class _Field>
INLINE vect_t sub_r (
            const vect_t a,
            const vect_t b) const [inline]
```

12.106.3.16 subin_r()

```
template<class _Field>
INLINE vect_t subin_r (
            vect_t & a,
            const vect_t b) const [inline]
```

12.106.3.17 zero() [1/2]

```
template<class _Field>
INLINE vect_t zero (
            vect_t & x) const [inline]
```

12.106.3.18 zero() [2/2]

```
template<class _Field>
INLINE vect_t zero () const [inline]
```

12.106.3.19 mod()

```
template<class _Field>
INLINE vect_t mod (
            vect_t & c) const [inline]
```

12.106.3.20 mul() [1/2]

```
template<class _Field>
INLINE vect_t mul (
            vect_t & c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.21 mul() [2/2]

```
template<class _Field>
INLINE vect_t mul (
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.22 mulin()

```
template<class _Field>
INLINE vect_t mulin (
            vect_t & a,
            const vect_t b) const  [inline]
```

12.106.3.23 mul_r() [1/2]

```
template<class _Field>
INLINE vect_t mul_r (
            vect_t & c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.24 mul_r() [2/2]

```
template<class _Field>
INLINE vect_t mul_r (
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.25 axpy() [1/2]

```
template<class _Field>
INLINE vect_t axpy (
            vect_t & r,
            const vect_t a,
            const vect_t b,
            const vect_t c) const  [inline]
```

12.106.3.26 axpy() [2/2]

```
template<class _Field>
INLINE vect_t axpy (
            const vect_t c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.27 axpyin()

```
template<class _Field>
INLINE vect_t axpyin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.28 axpy_r() [1/2]

```
template<class _Field>
INLINE vect_t axpy_r (
            vect_t & r,
            const vect_t a,
            const vect_t b,
            const vect_t c) const  [inline]
```

12.106.3.29 axpy_r() [2/2]

```
template<class _Field>
INLINE vect_t axpy_r (
            const vect_t c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.30 axpyin_r()

```
template<class _Field>
INLINE vect_t axpyin_r (
            vect_t & c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.31 maxpy() [1/2]

```
template<class _Field>
INLINE vect_t maxpy (
            vect_t & r,
            const vect_t a,
            const vect_t b,
            const vect_t c) const  [inline]
```

12.106.3.32 maxpy() [2/2]

```
template<class _Field>
INLINE vect_t maxpy (
            const vect_t c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.3.33 maxpyin()

```
template<class _Field>
INLINE vect_t maxpyin (
            vect_t & c,
            const vect_t a,
            const vect_t b) const  [inline]
```

12.106.4 Field Documentation

12.106.4.1 vect_size

```
template<class _Field>
const constexpr size_t vect_size = simd::vect_size  [static], [constexpr]
```

12.106.4.2 alignment

```
template<class _Field>
const constexpr size_t alignment = simd::alignment  [static], [constexpr]
```

The documentation for this class was generated from the following file:

- simd_modular.inl

12.107 FieldTraits< Field > Struct Template Reference

FieldTrait.

`#include <field-traits.h>`

Public Types

- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false

12.107.1 Detailed Description

template<class Field>
struct FFLAS::FieldTraits< Field >

FieldTrait.

12.107.2 Member Typedef Documentation

12.107.2.1 category

```
template<class Field>
typedef FieldCategories::GenericTag category
```

12.107.3 Field Documentation

12.107.3.1 balanced

```
template<class Field>
const bool balanced = false  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.108 FieldTraits< FFPACK::RNSInteger< T > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.108.1 Member Typedef Documentation

12.108.1.1 category [1/2]

```
template<typename T>
typedef FieldCategories::UnparametricTag category
```

12.108.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.108.2 Field Documentation

12.108.2.1 balanced [1/2]

```
template<typename T>
const bool balanced = false  [static]
```

12.108.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.109 FieldTraits< FFPACK::RNSIntegerMod< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::ModularTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.109.1 Member Typedef Documentation

12.109.1.1 category [1/2]

```
template<typename T>
typedef FieldCategories::ModularTag category
```

12.109.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.109.2 Field Documentation

12.109.2.1 balanced [1/2]

```
template<typename T>
const bool balanced = false  [static]
```

12.109.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.110 FieldTraits< Givaro::Modular< Element > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::ModularTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.110.1 Member Typedef Documentation

12.110.1.1 category [1/2]

```
template<class Element>
typedef FieldCategories::ModularTag category
```

12.110.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.110.2 Field Documentation

12.110.2.1 balanced [1/2]

```
template<class Element>
const bool balanced = false  [static]
```

12.110.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.111 FieldTraits< Givaro::ModularBalanced< Element > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::ModularTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = true
- static const bool balanced

12.111.1 Member Typedef Documentation

12.111.1.1 category [1/2]

```
template<class Element>
typedef FieldCategories::ModularTag category
```

12.111.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.111.2 Field Documentation

12.111.2.1 balanced [1/2]

```
template<class Element>
const bool balanced = true [static]
```

12.111.2.2 balanced [2/2]

```
const bool balanced [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.112 FieldTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.112.1 Member Typedef Documentation

12.112.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.112.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.112.2 Field Documentation

12.112.2.1 balanced [1/2]

```
const bool balanced = false [static]
```

12.112.2.2 balanced [2/2]

`const bool balanced [static]`

The documentation for this struct was generated from the following file:

- field-traits.h

12.113 FieldTraits< Givaro::ZRing< float > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.113.1 Member Typedef Documentation

12.113.1.1 category [1/2]

`typedef FieldCategories::UnparametricTag category`

12.113.1.2 category [2/2]

`typedef FieldCategories::GenericTag category`

12.113.2 Field Documentation

12.113.2.1 balanced [1/2]

`const bool balanced = false [static]`

12.113.2.2 balanced [2/2]

`const bool balanced [static]`

The documentation for this struct was generated from the following file:

- field-traits.h

12.114 FieldTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.114.1 Member Typedef Documentation

12.114.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.114.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.114.2 Field Documentation

12.114.2.1 balanced [1/2]

```
const bool balanced = false [static]
```

12.114.2.2 balanced [2/2]

```
const bool balanced [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.115 FieldTraits< Givaro::ZRing< int16_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.115.1 Member Typedef Documentation

12.115.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.115.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.115.2 Field Documentation

12.115.2.1 balanced [1/2]

```
const bool balanced = false [static]
```

12.115.2.2 balanced [2/2]

```
const bool balanced [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.116 FieldTraits< Givaro::ZRing< int32_t > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.116.1 Member Typedef Documentation

12.116.1.1 category [1/2]

`typedef FieldCategories::UnparametricTag category`

12.116.1.2 category [2/2]

`typedef FieldCategories::GenericTag category`

12.116.2 Field Documentation

12.116.2.1 balanced [1/2]

`const bool balanced = false [static]`

12.116.2.2 balanced [2/2]

`const bool balanced [static]`

The documentation for this struct was generated from the following file:

- field-traits.h

12.117 FieldTraits< Givaro::ZRing< int64_t > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.117.1 Member Typedef Documentation

12.117.1.1 category [1/2]

`typedef FieldCategories::UnparametricTag category`

12.117.1.2 category [2/2]

`typedef FieldCategories::GenericTag category`

12.117.2 Field Documentation

12.117.2.1 balanced [1/2]

```
const bool balanced = false  [static]
```

12.117.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.118 FieldTraits< Givaro::ZRing< RecInt::ruint< K > > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.118.1 Member Typedef Documentation

12.118.1.1 category [1/2]

```
template<size_t K>
typedef FieldCategories::UnparametricTag category
```

12.118.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.118.2 Field Documentation

12.118.2.1 balanced [1/2]

```
template<size_t K>
const bool balanced = false  [static]
```

12.118.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.119 FieldTraits< Givaro::ZRing< uint16_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.119.1 Member Typedef Documentation

12.119.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.119.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.119.2 Field Documentation

12.119.2.1 balanced [1/2]

```
const bool balanced = false  [static]
```

12.119.2.2 balanced [2/2]

```
const bool balanced  [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.120 FieldTraits< Givaro::ZRing< uint32_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.120.1 Member Typedef Documentation

12.120.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.120.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.120.2 Field Documentation

12.120.2.1 balanced [1/2]

```
const bool balanced = false [static]
```

12.120.2.2 balanced [2/2]

```
const bool balanced [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.121 FieldTraits< Givaro::ZRing< uint64_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef FieldCategories::UnparametricTag category
- typedef FieldCategories::GenericTag category

Static Public Attributes

- static const bool balanced = false
- static const bool balanced

12.121.1 Member Typedef Documentation

12.121.1.1 category [1/2]

```
typedef FieldCategories::UnparametricTag category
```

12.121.1.2 category [2/2]

```
typedef FieldCategories::GenericTag category
```

12.121.2 Field Documentation

12.121.2.1 balanced [1/2]

```
const bool balanced = false [static]
```

12.121.2.2 balanced [2/2]

```
const bool balanced [static]
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.122 Fixed Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.123 FixedPrecIntTag Struct Reference

Fixed precision integers above machine precision: Givaro::recInt.

```
#include <field-traits.h>
```

12.123.1 Detailed Description

Fixed precision integers above machine precision: Givaro::recInt.
The documentation for this struct was generated from the following file:

- field-traits.h

12.124 ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestDistribution Class Reference

Public Types

- using IntType = typename make_unsigned_int<Element>::type

Public Member Functions

- FloatingPointTestDistribution ()
- template<class Generator>
  Element operator() (Generator &g)

12.124.1 Member Typedef Documentation

12.124.1.1 IntType

```
template<typename Element>
using IntType = typename make_unsigned_int<Element>::type
```

12.124.2 Constructor & Destructor Documentation

12.124.2.1 FloatingPointTestDistribution()

```
template<typename Element>
FloatingPointTestDistribution () [inline]
```

12.124.3 Member Function Documentation

12.124.3.1 operator()()

```
template<typename Element>
template<class Generator>
Element operator() (
            Generator & g) [inline]
```

The documentation for this class was generated from the following file:

- test-simd.C

12.125 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

12.125.1 Constructor & Destructor Documentation

12.125.1.1 ForStrategy1D() [1/2]

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
ForStrategy1D (
            const blocksize_t n,
            const ParSeqHelper::Parallel< Cut, Param > H)  [inline]
```

12.125.1.2 ForStrategy1D() [2/2]

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
ForStrategy1D (
            const blocksize_t b,
            const blocksize_t e,
            const ParSeqHelper::Parallel< Cut, Param > H)  [inline]
```

12.125.2 Member Function Documentation

12.125.2.1 build()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
void build (
            const blocksize_t n,
            const ParSeqHelper::Parallel< Cut, Param > H)  [inline]
```

12.125.2.2 initialize()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t initialize ()  [inline]
```

12.125.2.3 isTerminated()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
bool isTerminated () const  [inline]
```

12.125.2.4 begin()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t begin () const [inline]
```

12.125.2.5 end()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t end () const [inline]
```

12.125.2.6 numblocks()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numblocks () const [inline]
```

12.125.2.7 blockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t blockindex () const [inline]
```

12.125.2.8 operator++()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t operator++ () [inline]
```

12.125.3 Field Documentation

12.125.3.1 ibeg

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t ibeg [protected]
```

12.125.3.2 iend

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t iend [protected]
```

12.125.3.3 current

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t current [protected]
```

12.125.3.4 firstBlockSize

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t firstBlockSize [protected]
```

12.125.3.5 lastBlockSize

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t lastBlockSize [protected]
```

12.125.3.6 changeBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t changeBS  [protected]
```

12.125.3.7 numBlock

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numBlock  [protected]
```

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.126 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)

12.126.1 Constructor & Destructor Documentation

12.126.1.1 ForStrategy2D()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
ForStrategy2D (
             const blocksize_t m,
             const blocksize_t n,
             const ParSeqHelper::Parallel< Cut, Param > H)  [inline]
```

12.126.2 Member Function Documentation

12.126.2.1 initialize()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t initialize ()  [inline]
```

12.126.2.2 isTerminated()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
bool isTerminated () const  [inline]
```

12.126.2.3 ibegin()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t ibegin () const  [inline]
```

12.126.2.4 jbegin()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t jbegin () const  [inline]
```

12.126.2.5 iend()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t iend () const  [inline]
```

12.126.2.6 jend()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t jend () const  [inline]
```

12.126.2.7 operator++()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t operator++ ()  [inline]
```

12.126.2.8 rownumblocks()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t rownumblocks () const  [inline]
```

12.126.2.9 colnumblocks()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t colnumblocks () const  [inline]
```

12.126.2.10 blockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t blockindex () const  [inline]
```

12.126.2.11 rowblockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t rowblockindex () const  [inline]
```

12.126.2.12 colblockindex()

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t colblockindex () const  [inline]
```

12.126.3 Friends And Related Symbol Documentation

12.126.3.1 operator<<

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
std::ostream & operator<< (
            std::ostream & out,
            const ForStrategy2D< blocksize_t, Cut, Param > & FS2D)  [friend]
```

12.126.4 Field Documentation

12.126.4.1 _ibeg

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _ibeg  [protected]
```

12.126.4.2 _iend

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _iend  [protected]
```

12.126.4.3 _jbeg

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _jbeg  [protected]
```

12.126.4.4 _jend

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t _jend  [protected]
```

12.126.4.5 rowBlockSize

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t rowBlockSize  [protected]
```

12.126.4.6 colBlockSize

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t colBlockSize  [protected]
```

12.126.4.7 current

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t current  [protected]
```

12.126.4.8 lastRBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t lastRBS  [protected]
```

12.126.4.9 lastCBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t lastCBS  [protected]
```

12.126.4.10 changeRBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t changeRBS  [protected]
```

12.126.4.11 changeCBS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t changeCBS  [protected]
```

12.126.4.12 numRowBlock

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numRowBlock  [protected]
```

12.126.4.13 numColBlock

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t numColBlock  [protected]
```

12.126.4.14 BLOCKS

```
template<typename blocksize_t = size_t, typename Cut = CuttingStrategy::Block, typename Param
= StrategyParameter::Threads>
blocksize_t BLOCKS  [protected]
```

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.127 ftrmmLeftLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.128 ftrmmLeftLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.129 ftrmmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.130 ftrmmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.131 ftrmmLeftUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.132 ftrmmLeftUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.133 ftrmmLeftUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.134 ftrmmLeftUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.135 ftrmmRightLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.136 ftrmmRightLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.137 ftrmmRightLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.138 ftrmmRightLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.139 ftrmmRightUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.140 ftrmmRightUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.141 ftrmmRightUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.142 ftrmmRightUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.143 ftrsmLeftLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.144 ftrsmLeftLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.145 ftrsmLeftLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.146 ftrsmLeftLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.147 ftrsmLeftUpperNoTransNonUnit< Element > Class Template Reference

Computes the maximal size for delaying the modular reduction in a triangular system resolution.

12.147.1 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 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

12.148 ftrsmLeftUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.149 ftrsmLeftUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.150 ftrsmLeftUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.151 ftrsmRightLowerNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.152 ftrsmRightLowerNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.153 ftrsmRightLowerTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.154 ftrsmRightLowerTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.155 ftrsmRightUpperNoTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.156 ftrsmRightUpperNoTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.157 ftrsmRightUpperTransNonUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.158 ftrsmRightUpperTransUnit< Element > Class Template Reference

The documentation for this class was generated from the following file:

- fflas_level3.inl

12.159 GenericTag Struct Reference

default is generic
`#include <field-traits.h>`

12.159.1 Detailed Description

default is generic
The documentation for this struct was generated from the following file:

- field-traits.h

12.160 GenericTag Struct Reference

generic ring.
`#include <field-traits.h>`

12.160.1 Detailed Description

generic ring.
The documentation for this struct was generated from the following file:

- field-traits.h

12.161 Grain Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.162 has_minus_eq_impl< C > Struct Template Reference

`#include <sparse_matrix_traits.h>`

Static Public Attributes

- static constexpr bool value = type::value

12.162.1 Field Documentation

12.162.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.163 has_minus_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

12.163.1 Field Documentation

12.163.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.164 has_mul_eq_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

12.164.1 Field Documentation

12.164.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.165 has_mul_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

12.165.1 Field Documentation

12.165.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.166 has_operation< T > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value

12.166.1 Field Documentation

12.166.1.1 value

```
template<class T>
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

12.167 has_plus_eq_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

12.167.1 Field Documentation

12.167.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.168 has_plus_impl< C > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Static Public Attributes

- static constexpr bool value = type::value

12.168.1 Field Documentation

12.168.1.1 value

```
template<typename C>
bool value = type::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.169 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

12.169.1 Field Documentation

12.169.1.1 none

```
uint64_t none = 0_ui64  [static], [constexpr]
```

12.169.1.2 coo

```
uint64_t coo = 1_ui64  [static], [constexpr]
```

12.169.1.3 csr

```
uint64_t csr = 1_ui64 << 1  [static], [constexpr]
```

12.169.1.4 ell

```
uint64_t ell = 1_ui64 << 2  [static], [constexpr]
```

12.169.1.5 aut

```
uint64_t aut = 1_ui64 << 32  [static], [constexpr]
```

12.169.1.6 pm1

```
uint64_t pm1 = 1_ui64 << 33  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- fflas_sparse.h

12.170 HelperMod< Field, ElementTraits > Struct Template Reference

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

12.171 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

12.171.1 Constructor & Destructor Documentation

12.171.1.1 HelperMod() [1/2]

```
template<class Field>
HelperMod ( ) [inline]
```

12.171.1.2 HelperMod() [2/2]

```
template<class Field>
HelperMod (
            const Field & F) [inline]
```

12.171.2 Field Documentation

12.171.2.1 p

```
template<class Field>
Field::Element p
```

12.171.2.2 invp

```
template<class Field>
double invp
```

12.171.2.3 min

```
template<class Field>
double min
```

12.171.2.4 max

```
template<class Field>
double max
```

12.171.2.5 pow50rem

```
template<class Field>
int64_t pow50rem
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

12.172 HelperMod< Field, FFLAS::ElementCategories::ArbitraryPrecIntTag > Struct Template Reference

Public Member Functions

- HelperMod ()
- HelperMod (const Field &F)

Data Fields

- Field::Element p

12.172.1 Constructor & Destructor Documentation

12.172.1.1 HelperMod() [1/2]

```
template<class Field>
HelperMod ()  [inline]
```

12.172.1.2 HelperMod() [2/2]

```
template<class Field>
HelperMod (
            const Field & F)  [inline]
```

12.172.2 Field Documentation

12.172.2.1 p

```
template<class Field>
Field::Element p
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

12.173 HelperMod< Field, FFLAS::ElementCategories::FixedPrecIntTag > Struct Template Reference

Public Member Functions

- HelperMod ()
- HelperMod (const Field &F)

Data Fields

- Field::Element p

12.173.1 Constructor & Destructor Documentation

12.173.1.1 HelperMod() [1/2]

```
template<class Field>
HelperMod ()  [inline]
```

12.173.1.2 HelperMod() [2/2]

```
template<class Field>
HelperMod (
            const Field & F)  [inline]
```

12.173.2 Field Documentation

12.173.2.1 p

```
template<class Field>
Field::Element p
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

12.174 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

12.174.1 Constructor & Destructor Documentation

12.174.1.1 HelperMod() [1/2]

```
template<class Field>
HelperMod ()  [inline]
```

12.174.1.2 HelperMod() [2/2]

```
template<class Field>
HelperMod (
            const Field & F)  [inline]
```

12.174.2 Field Documentation

12.174.2.1 p

```
template<class Field>
Field::Element p
```

12.174.2.2 invp

```
template<class Field>
Field::Element invp
```

12.174.2.3 min

```
template<class Field>
Field::Element min
```

12.174.2.4 max

```
template<class Field>
Field::Element max
```

The documentation for this struct was generated from the following file:

- fflas_freduce.inl

12.175 Hybrid Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.176 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

12.176.1 Constructor & Destructor Documentation

12.176.1.1 Info() [1/4]

```
Info (
            uint64_t it,
            uint64_t s,
            uint64_t p)  [inline]
```

12.176.1.2 Info() [2/4]

```
Info ()  [default]
```

12.176.1.3 Info() [3/4]

```
Info (
            const Info & )  [default]
```

12.176.1.4 Info() [4/4]

```
Info (
            Info && )  [default]
```

12.176.2 Member Function Documentation

12.176.2.1 operator=() [1/2]

```
Info & operator= (
            const Info & )  [default]
```

12.176.2.2 operator=() [2/2]

```
Info & operator= (
            Info && )  [default]
```

12.176.3 Field Documentation

12.176.3.1 size

```
uint64_t size = 0
```

12.176.3.2 perm

```
uint64_t perm = 0
```

12.176.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- csr_hyb_utils.inl

12.177 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

12.177.1 Constructor & Destructor Documentation

12.177.1.1 Info() [1/4]

```
Info (
            uint64_t it,
            uint64_t s,
            uint64_t p)  [inline]
```

12.177.1.2 Info() [2/4]

```
Info ()  [default]
```

12.177.1.3 Info() [3/4]

```
Info (
            const Info & )  [default]
```

12.177.1.4 Info() [4/4]

```
Info (
            Info && )  [default]
```

12.177.2 Member Function Documentation

12.177.2.1 operator=() [1/2]

```
Info & operator= (
            const Info & )  [default]
```

12.177.2.2 operator=() [2/2]

```
Info & operator= (
            Info && )  [default]
```

12.177.3 Field Documentation

12.177.3.1 size

```
uint64_t size = 0
```

12.177.3.2 perm

```
uint64_t perm = 0
```

12.177.3.3 begin

```
uint64_t begin = 0
```

The documentation for this struct was generated from the following file:

- sell_utils.inl

12.178 is_all_same< Args > Struct Template Reference

The documentation for this struct was generated from the following file:

- test-simd.C

12.179 is_all_same< T, Args... > Struct Template Reference

Static Public Attributes

- static constexpr bool value = ALL<std::is_same<typename decay<T>::type, typename decay<Args>↩ ::type>::value...>::value

12.179.1 Field Documentation

12.179.1.1 value

```
template<typename T, typename...  Args>
bool value = ALL<std::is_same<typename decay<T>::type, typename decay<Args>::type>::value...>↩
::value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.180 is_all_same<> Struct Reference

Static Public Attributes

- static constexpr bool value = true

12.180.1 Field Documentation

12.180.1.1 value

```
bool value = true [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.181 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

12.181.1 Member Typedef Documentation

12.181.1.1 type

```
template<class T>
using type = std::integral_constant<bool, false>
```

12.181.2 Field Documentation

12.181.2.1 value

```
template<class T>
const constexpr bool value = false [static], [constexpr]
```

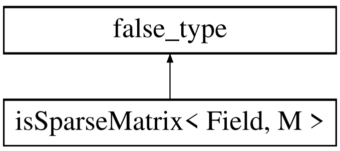
The documentation for this struct was generated from the following file:

- fflas_simd.h

12.182 isSparseMatrix< Field, M > Struct Template Reference

```
#include <sparse_matrix_traits.h>
```

Inheritance diagram for isSparseMatrix< Field, M >:



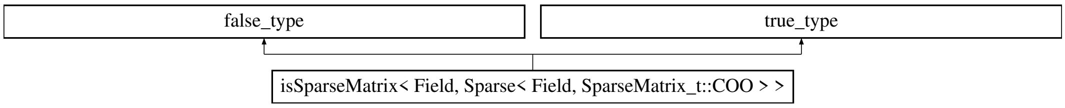
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.183 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 > >:



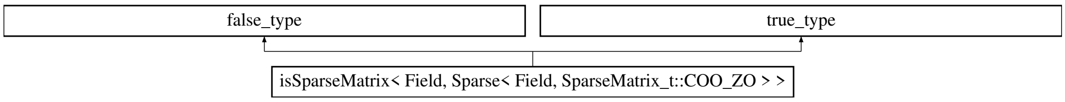
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.184 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 > >:



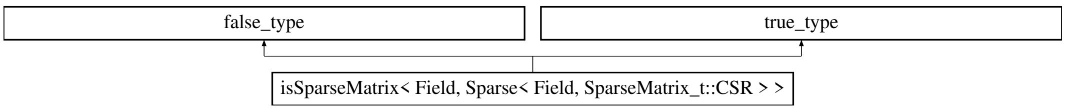
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.185 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 > >:



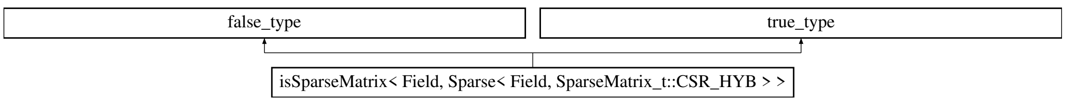
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.186 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 > >:



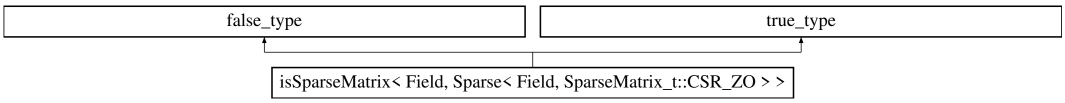
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.187 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 > >:



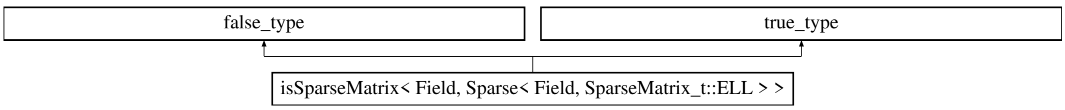
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.188 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 > >:



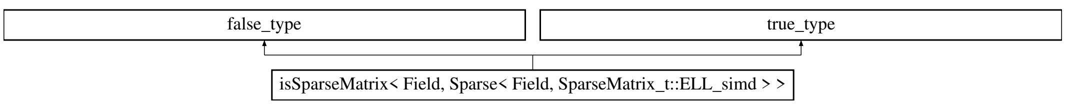
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.189 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 > >:



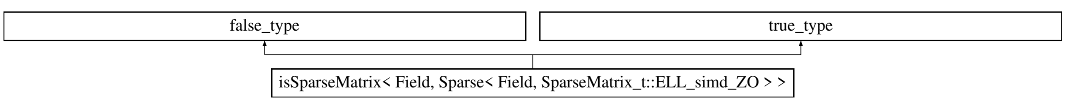
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.190 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 > >:



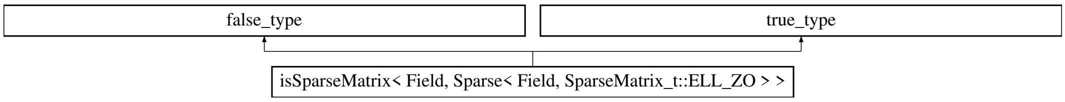
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.191 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 > >:



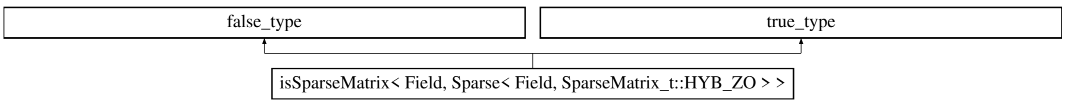
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.192 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 > >:



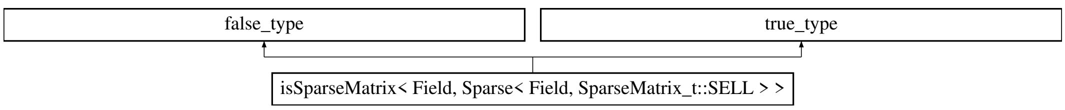
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.193 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 > >:



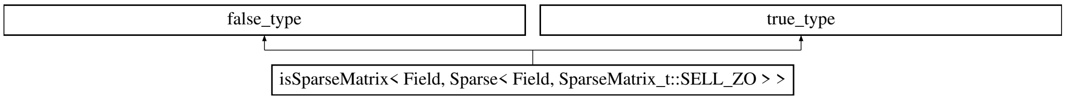
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.194 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 > >:



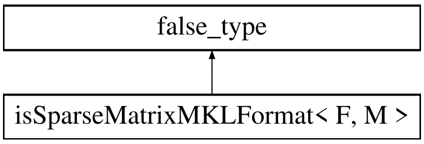
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.195 isSparseMatrixMKLFormat< F, M > Struct Template Reference

`#include <sparse_matrix_traits.h>`

Inheritance diagram for isSparseMatrixMKLFormat< F, M >:



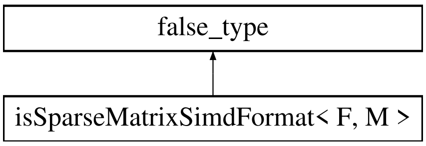
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.196 isSparseMatrixSimdFormat< F, M > Struct Template Reference

`#include <sparse_matrix_traits.h>`

Inheritance diagram for isSparseMatrixSimdFormat< F, M >:



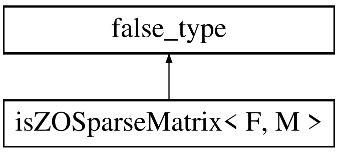
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.197 isZOSparseMatrix< F, M > Struct Template Reference

`#include <sparse_matrix_traits.h>`

Inheritance diagram for isZOSparseMatrix< F, M >:



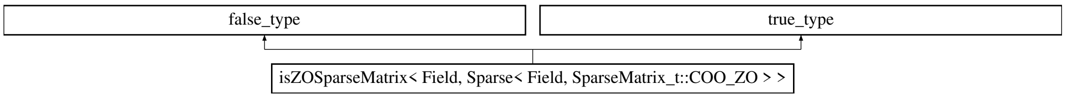
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.198 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 > >:



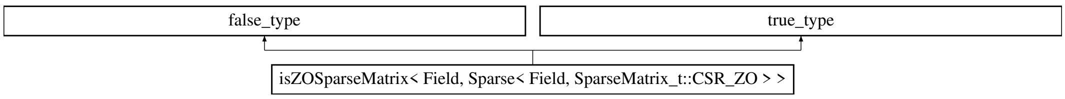
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.199 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 > >:



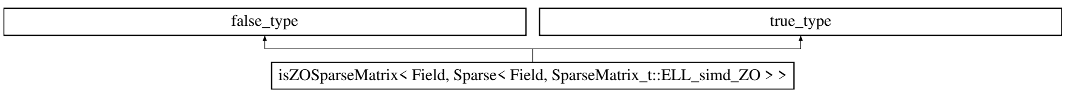
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.200 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 > >:



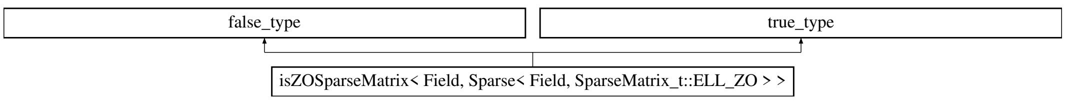
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.201 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 > >:



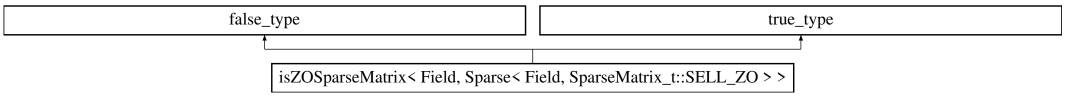
The documentation for this struct was generated from the following file:

- sparse_matrix_traits.h

12.202 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

12.203 Iterative Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.204 array$<$ T $>$::iterator Class Reference

STL iterator class.

12.204.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.205 string::iterator Class Reference

STL iterator class.

12.205.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.206 vector$<$ T $>$::iterator Class Reference

STL iterator class.

12.206.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.207 LazyTag Struct Reference

Performs field operations with delayed mod only when necessary. Result may not be reduced.
#include <field-traits.h>

12.207.1 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

12.208 limits< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- flimits.h

12.209 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

12.209.1 Member Typedef Documentation

12.209.1.1 T

```
typedef char T
```

12.209.2 Member Function Documentation

12.209.2.1 max()

```
static constexpr char max () [inline], [static], [constexpr], [noexcept]
```

12.209.2.2 min()

```
static constexpr char min () [inline], [static], [constexpr], [noexcept]
```

12.209.2.3 digits()

```
static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.210 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

12.210.1 Member Typedef Documentation

12.210.1.1 T

```
typedef double T
```

12.210.2 Member Function Documentation

12.210.2.1 max()

```
static constexpr int64_t max () [inline], [static], [constexpr], [noexcept]
```

12.210.2.2 min()

```
static constexpr int64_t min () [inline], [static], [constexpr], [noexcept]
```

12.210.2.3 digits()

```
static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.211 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

12.211.1 Member Typedef Documentation

12.211.1.1 T

```
typedef float T
```

12.211.2 Member Function Documentation

12.211.2.1 max()

```
static constexpr int32_t max () [inline], [static], [constexpr], [noexcept]
```

12.211.2.2 min()

```
static constexpr int32_t min () [inline], [static], [constexpr], [noexcept]
```

12.211.2.3 digits()

`static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]`

The documentation for this struct was generated from the following file:

- flimits.h

12.212 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

12.212.1 Member Typedef Documentation

12.212.1.1 T

`typedef Givaro::Integer T`

12.212.2 Member Function Documentation

12.212.2.1 max()

`static constexpr int max () [inline], [static], [constexpr], [noexcept]`

12.212.2.2 min()

`static constexpr int min () [inline], [static], [constexpr], [noexcept]`

The documentation for this struct was generated from the following file:

- flimits.h

12.213 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

12.213.1 Member Typedef Documentation

12.213.1.1 T

`typedef int T`

12.213.2 Member Function Documentation

12.213.2.1 max()

```
static constexpr int max () [inline], [static], [constexpr], [noexcept]
```

12.213.2.2 min()

```
static constexpr int min () [inline], [static], [constexpr], [noexcept]
```

12.213.2.3 digits()

```
static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.214 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

12.214.1 Member Typedef Documentation

12.214.1.1 T

```
typedef long T
```

12.214.2 Member Function Documentation

12.214.2.1 max()

```
static constexpr long max () [inline], [static], [constexpr], [noexcept]
```

12.214.2.2 min()

```
static constexpr long min () [inline], [static], [constexpr], [noexcept]
```

12.214.2.3 digits()

```
static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.215 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

12.215.1 Member Typedef Documentation

12.215.1.1 T

```
typedef long long T
```

12.215.2 Member Function Documentation

12.215.2.1 max()

```
static constexpr long long max ()  [inline], [static], [constexpr], [noexcept]
```

12.215.2.2 min()

```
static constexpr long long min ()  [inline], [static], [constexpr], [noexcept]
```

12.215.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.216 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

12.216.1 Member Typedef Documentation

12.216.1.1 T

```
template<size_t K>
typedef RecInt::ruint<K> T
```

12.216.2 Member Function Documentation

12.216.2.1 max()

```
template<size_t K>
static constexpr RecInt::rint< K > max ()  [inline], [static], [constexpr], [noexcept]
```

12.216.2.2 min()

```
template<size_t K>
static constexpr RecInt::rint< K > min ()  [inline], [static], [constexpr], [noexcept]
```
The documentation for this struct was generated from the following file:

- flimits.h

12.217 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

12.217.1 Member Typedef Documentation

12.217.1.1 T

```
template<size_t K>
typedef RecInt::ruint<K> T
```

12.217.2 Member Function Documentation

12.217.2.1 max()

```
template<size_t K>
static constexpr RecInt::ruint< K > max ()  [inline], [static], [constexpr], [noexcept]
```

12.217.2.2 min()

```
template<size_t K>
static constexpr RecInt::ruint< K > min ()  [inline], [static], [constexpr], [noexcept]
```
The documentation for this struct was generated from the following file:

- flimits.h

12.218 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

12.218.1 Member Typedef Documentation

12.218.1.1 T

```
typedef short int T
```

12.218.2 Member Function Documentation

12.218.2.1 max()

```
static constexpr short int max () [inline], [static], [constexpr], [noexcept]
```

12.218.2.2 min()

```
static constexpr short int min () [inline], [static], [constexpr], [noexcept]
```

12.218.2.3 digits()

```
static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.219 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

12.219.1 Member Typedef Documentation

12.219.1.1 T

```
typedef signed char T
```

12.219.2 Member Function Documentation

12.219.2.1 max()

```
static constexpr signed char max () [inline], [static], [constexpr], [noexcept]
```

12.219.2.2 min()

```
static constexpr signed char min () [inline], [static], [constexpr], [noexcept]
```

12.219.2.3 digits()

```
static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.220 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

12.220.1 Member Typedef Documentation

12.220.1.1 T

```
typedef unsigned char T
```

12.220.2 Member Function Documentation

12.220.2.1 max()

```
static constexpr unsigned char max ()  [inline], [static], [constexpr], [noexcept]
```

12.220.2.2 min()

```
static constexpr unsigned char min ()  [inline], [static], [constexpr], [noexcept]
```

12.220.2.3 digits()

```
static constexpr int32_t digits ()  [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.221 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

12.221.1 Member Typedef Documentation

12.221.1.1 T

```
typedef unsigned int T
```

12.221.2 Member Function Documentation

12.221.2.1 max()

`static constexpr unsigned int max () [inline], [static], [constexpr], [noexcept]`

12.221.2.2 min()

`static constexpr unsigned int min () [inline], [static], [constexpr], [noexcept]`

12.221.2.3 digits()

`static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]`

The documentation for this struct was generated from the following file:

- flimits.h

12.222 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

12.222.1 Member Typedef Documentation

12.222.1.1 T

`typedef unsigned long T`

12.222.2 Member Function Documentation

12.222.2.1 max()

`static constexpr unsigned long max () [inline], [static], [constexpr], [noexcept]`

12.222.2.2 min()

`static constexpr unsigned long min () [inline], [static], [constexpr], [noexcept]`

12.222.2.3 digits()

`static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]`

The documentation for this struct was generated from the following file:

- flimits.h

12.223 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

12.223.1 Member Typedef Documentation

12.223.1.1 T

```
typedef unsigned long long T
```

12.223.2 Member Function Documentation

12.223.2.1 max()

```
static constexpr unsigned long long max () [inline], [static], [constexpr], [noexcept]
```

12.223.2.2 min()

```
static constexpr unsigned long long min () [inline], [static], [constexpr], [noexcept]
```

12.223.2.3 digits()

```
static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]
```

The documentation for this struct was generated from the following file:

- flimits.h

12.224 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

12.224.1 Member Typedef Documentation

12.224.1.1 T

```
typedef unsigned short int T
```

12.224.2 Member Function Documentation

12.224.2.1 max()

```
static constexpr unsigned short int max () [inline], [static], [constexpr], [noexcept]
```

12.224.2.2 min()

```
static constexpr unsigned short int min () [inline], [static], [constexpr], [noexcept]
```

12.224.2.3 digits()

`static constexpr int32_t digits () [inline], [static], [constexpr], [noexcept]`

The documentation for this struct was generated from the following file:

- flimits.h

12.225 MachineFloatTag Struct Reference

float or double

`#include <field-traits.h>`

12.225.1 Detailed Description

float or double

The documentation for this struct was generated from the following file:

- field-traits.h

12.226 MachineIntTag Struct Reference

short, int, long, long long, and unsigned variants

`#include <field-traits.h>`

12.226.1 Detailed Description

short, int, long, long long, and unsigned variants

The documentation for this struct was generated from the following file:

- field-traits.h

12.227 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)

- bool checkOut (const Field &F, FFLAS_UPLO uplo, 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)

12.227.1 Member Typedef Documentation

12.227.1.1 Self_t

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
typedef MMHelper<Field,AlgoTrait,ModeTrait,ParSeqTrait> Self_t
```

12.227.1.2 DelayedField_t

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
typedef associatedDelayedField<constField>::type DelayedField_t
```

12.227.1.3 DelayedField

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
typedef associatedDelayedField<constField>::field DelayedField
```

12.227.1.4 DFElt

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
typedef DelayedField::Element DFElt
```

12.227.2 Constructor & Destructor Documentation

12.227.2.1 MMHelper() [1/5]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
MMHelper () [inline]
```

12.227.2.2 MMHelper() [2/5]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
MMHelper (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            ParSeqTrait _PS)  [inline]
```

12.227.2.3 MMHelper() [3/5]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
MMHelper (
            const Field & F,
            int w,
            ParSeqTrait _PS = ParSeqTrait())  [inline]
```

12.227.2.4 MMHelper() [4/5]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
template<class F2, typename AlgoT2, typename FT2, typename PS2>
MMHelper (
            MMHelper< F2, AlgoT2, FT2, PS2 > & WH)  [inline]
```

12.227.2.5 MMHelper() [5/5]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
MMHelper (
            const Field & F,
            int w,
            DFElt _Amin,
            DFElt _Amax,
            DFElt _Bmin,
            DFElt _Bmax,
            DFElt _Cmin,
            DFElt _Cmax,
            ParSeqTrait _PS = ParSeqTrait())  [inline]
```

12.227.3 Member Function Documentation

12.227.3.1 initC()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
void initC ()  [inline]
```

12.227.3.2 initA()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
void initA ()  [inline]
```

12.227.3.3 initB()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
void initB ()  [inline]
```

12.227.3.4 initOut()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
void initOut ()  [inline]
```

12.227.3.5 MaxDelayedDim()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
size_t MaxDelayedDim (
            DFElt beta)  [inline]
```

12.227.3.6 Aunfit()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
bool Aunfit ()  [inline]
```

12.227.3.7 Bunfit()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
bool Bunfit ()  [inline]
```

12.227.3.8 setOutBounds()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
void setOutBounds (
            const size_t k,
            const DFElt alpha,
            const DFElt beta)  [inline]
```

12.227.3.9 checkA()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
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]
```

12.227.3.10 checkB()

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
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]
```

12.227.3.11 checkOut() [1/2]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
bool checkOut (
            const Field & F,
            const size_t M,
            const size_t N,
            typename Field::ConstElement_ptr A,
            const size_t lda)  [inline]
```

12.227.3.12 checkOut() [2/2]

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
bool checkOut (
```

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

12.227.4 Friends And Related Symbol Documentation

12.227.4.1 operator<<

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
std::ostream & operator<< (
            std::ostream & out,
            const Self_t & M)  [friend]
```

12.227.5 Field Documentation

12.227.5.1 recLevel

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
int recLevel
```

12.227.5.2 FieldMin

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt FieldMin
```

12.227.5.3 FieldMax

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt FieldMax
```

12.227.5.4 Amin

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Amin
```

12.227.5.5 Amax

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Amax
```

12.227.5.6 Bmin

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Bmin
```

12.227.5.7 Bmax

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Bmax
```

12.227.5.8 Cmin

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Cmin
```

12.227.5.9 Cmax

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Cmax
```

12.227.5.10 Outmin

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Outmin
```

12.227.5.11 Outmax

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt Outmax
```

12.227.5.12 MaxStorableValue

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
DFElt MaxStorableValue
```

12.227.5.13 delayedField

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
const DelayedField_t delayedField
```

12.227.5.14 parseq

```
template<class Field, typename AlgoTrait, typename ModeTrait, typename ParSeqTrait>
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.228 ModeTraits< Field > Struct Template Reference

ModeTraits.

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::DefaultTag value

12.228.1 Detailed Description

template<class Field>
struct FFLAS::ModeTraits< Field >

ModeTraits.

12.228.2 Member Typedef Documentation

12.228.2.1 value

```
template<class Field>
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.229 ModeTraits< Givaro::Modular< Element, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::DelayedTag value
- typedef ModeCategories::DefaultTag value

12.229.1 Member Typedef Documentation

12.229.1.1 value [1/2]

```
template<typename Element, typename Compute>
typedef ModeCategories::DelayedTag value
```

12.229.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.230 ModeTraits< Givaro::Modular< Givaro::Integer, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value
- typedef ModeCategories::DefaultTag value

12.230.1 Member Typedef Documentation

12.230.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

12.230.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.231 ModeTraits< Givaro::Modular< int16_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.231.1 Member Typedef Documentation

12.231.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.231.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.232 ModeTraits< Givaro::Modular< int32_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.232.1 Member Typedef Documentation

12.232.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.232.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.233 ModeTraits< Givaro::Modular< int64_t, uint64_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::DefaultTag value
- typedef ModeCategories::DefaultTag value

12.233.1 Member Typedef Documentation

12.233.1.1 value [1/2]

```
typedef ModeCategories::DefaultTag value
```

12.233.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.234 ModeTraits< Givaro::Modular< int8_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.234.1 Member Typedef Documentation

12.234.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.234.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.235 ModeTraits< Givaro::Modular< RecInt::ruint< K >, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value
- typedef ModeCategories::DefaultTag value

12.235.1 Member Typedef Documentation

12.235.1.1 value [1/2]

```
template<typename Compute, size_t K>
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

12.235.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.236 ModeTraits< Givaro::Modular< uint16_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.236.1 Member Typedef Documentation

12.236.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.236.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.237 ModeTraits< Givaro::Modular< uint32_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.237.1 Member Typedef Documentation

12.237.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.237.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.238 ModeTraits< Givaro::Modular< uint8_t, Compute > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.238.1 Member Typedef Documentation

12.238.1.1 value [1/2]

```
template<typename Compute>
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.238.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.239 ModeTraits< Givaro::ModularBalanced< Element > > Struct Template Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::DelayedTag value
- typedef ModeCategories::DefaultTag value

12.239.1 Member Typedef Documentation

12.239.1.1 value [1/2]

```
template<typename Element>
typedef ModeCategories::DelayedTag value
```

12.239.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.240 ModeTraits< Givaro::ModularBalanced< Givaro::Integer > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value
- typedef ModeCategories::DefaultTag value

12.240.1 Member Typedef Documentation

12.240.1.1 value [1/2]

```
typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value
```

12.240.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.241 ModeTraits< Givaro::ModularBalanced< int16_t > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.241.1 Member Typedef Documentation

12.241.1.1 value [1/2]

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.241.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.242 ModeTraits< Givaro::ModularBalanced< int32_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.242.1 Member Typedef Documentation

12.242.1.1 value [1/2]

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.242.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.243 ModeTraits< Givaro::ModularBalanced< int8_t > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::MachineFloatTag > value
- typedef ModeCategories::DefaultTag value

12.243.1 Member Typedef Documentation

12.243.1.1 value [1/2]

```
typedef ModeCategories::ConvertTo<ElementCategories::MachineFloatTag> value
```

12.243.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.244 ModeTraits< Givaro::Montgomery< T > > Struct Template Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::DefaultBoundedTag value
- typedef ModeCategories::DefaultTag value

12.244.1 Member Typedef Documentation

12.244.1.1 value [1/2]

```
template<class T>
typedef ModeCategories::DefaultBoundedTag value
```

12.244.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.245 ModeTraits< Givaro::ZRing< double > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::DefaultBoundedTag value
- typedef ModeCategories::DefaultTag value

12.245.1 Member Typedef Documentation

12.245.1.1 value [1/2]

```
typedef ModeCategories::DefaultBoundedTag value
```

12.245.1.2 value [2/2]

```
typedef ModeCategories::DefaultTag value
```

The documentation for this struct was generated from the following file:

- field-traits.h

12.246 ModeTraits< Givaro::ZRing< float > > Struct Reference

```
#include <field-traits.h>
```

Public Types

- typedef ModeCategories::DefaultBoundedTag value
- typedef ModeCategories::DefaultTag value

12.246.1 Member Typedef Documentation

12.246.1.1 value [1/2]

```
typedef ModeCategories::DefaultBoundedTag value
```

12.246.1.2 value [2/2]

`typedef ModeCategories::DefaultTag value`

The documentation for this struct was generated from the following file:

- field-traits.h

12.247 ModeTraits< Givaro::ZRing< Givaro::Integer > > Struct Reference

`#include <field-traits.h>`

Public Types

- typedef ModeCategories::ConvertTo< ElementCategories::RNSElementTag > value
- typedef ModeCategories::DefaultTag value

12.247.1 Member Typedef Documentation

12.247.1.1 value [1/2]

`typedef ModeCategories::ConvertTo<ElementCategories::RNSElementTag> value`

12.247.1.2 value [2/2]

`typedef ModeCategories::DefaultTag value`

The documentation for this struct was generated from the following file:

- field-traits.h

12.248 ModularBalanced< T > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

12.249 ModularBalanced< T > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

12.250 ModularTag Struct Reference

This is a modular field like e.g. `Modular<T>` or `ModularBalanced<T>`
`#include <field-traits.h>`

12.250.1 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

12.251 Montgomery< T > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

12.252 need_field_characteristic< Field > Struct Template Reference

Static Public Attributes

- static constexpr bool value = false

12.252.1 Field Documentation

12.252.1.1 value

```
template<typename Field>
bool value = false  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.253 need_field_characteristic< Givaro::Modular< Field > > Struct Template Reference

Static Public Attributes

- static constexpr bool value = true
- static constexpr bool value

12.253.1 Field Documentation

12.253.1.1 value [1/2]

```
template<typename Field>
bool value = true  [static], [constexpr]
```

12.253.1.2 value [2/2]

```
bool value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.254 need_field_characteristic< Givaro::ModularBalanced< Field > > Struct Template Reference

Static Public Attributes

- static constexpr bool value = true
- static constexpr bool value

12.254.1 Field Documentation

12.254.1.1 value [1/2]

```
template<typename Field>
bool value = true  [static], [constexpr]
```

12.254.1.2 value [2/2]

```
bool value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- benchmark-fgemv.C

12.255 NoSimd< T > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using vect_t = T*
- using scalar_t = T
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, 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
- static const constexpr size_t alignment = static_cast<size_t>(Alignment::Normal)

12.255.1 Member Typedef Documentation

12.255.1.1 vect_t

```
template<typename T>
using vect_t = T*
```

12.255.1.2 scalar_t

```
template<typename T>
using scalar_t = T
```

12.255.1.3 aligned_allocator

```
template<typename T>
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.255.1.4 aligned_vector

```
template<typename T>
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.255.1.5 is_same_element

```
template<typename T>
template<class Field>
using is_same_element = std::is_same<typename Field::Element, T>
```

12.255.2 Member Function Documentation

12.255.2.1 type_string()

```
template<typename T>
static const std::string type_string () [inline], [static]
```

12.255.2.2 valid()

```
template<typename T>
template<class TT>
static constexpr bool valid (
            TT p)  [inline], [static], [constexpr]
```

12.255.2.3 compliant()

```
template<typename T>
template<class TT>
static constexpr bool compliant (
            TT n)  [inline], [static], [constexpr]
```

12.255.3 Field Documentation

12.255.3.1 vect_size

```
template<typename T>
const constexpr size_t vect_size = 1  [static], [constexpr]
```

12.255.3.2 alignment

```
template<typename T>
const constexpr size_t alignment = static_cast<size_t>(Alignment::Normal)  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

12.256 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)

12.256.1 Member Typedef Documentation

12.256.1.1 Cut

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
typedef C Cut
```

12.256.1.2 Param

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
typedef P Param
```

12.256.2 Constructor & Destructor Documentation

12.256.2.1 Parallel()

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
Parallel (
            size_t n = NUM_THREADS)  [inline]
```

12.256.3 Member Function Documentation

12.256.3.1 numthreads()

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
size_t numthreads () const  [inline]
```

12.256.3.2 set_numthreads()

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
size_t & set_numthreads (
            size_t n)  [inline]
```

12.256.4 Friends And Related Symbol Documentation

12.256.4.1 operator<<

```
template<typename C = CuttingStrategy::Block, typename P = StrategyParameter::Threads>
std::ostream & operator<< (
            std::ostream & out,
            const Parallel< C, P > & p)  [friend]
```

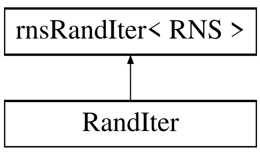
The documentation for this struct was generated from the following file:

- blockcuts.inl

12.257 RNSInteger< RNS >::RandIter Class Reference

```
#include <rns-integer.h>
```

Inheritance diagram for RNSInteger< RNS >::RandIter:



Public Member Functions

- RandIter (const RNSInteger< RNS > &F, 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

12.257.1 Constructor & Destructor Documentation

12.257.1.1 RandIter()

```
template<typename RNS>
RandIter (
            const RNSInteger< RNS > & F,
            uint64_t seed = 0)  [inline]
```

12.257.2 Member Function Documentation

12.257.2.1 random() [1/2]

```
template<typename RNS>
RNS::Element & random (
            typename RNS::Element & elt) const  [inline], [inherited]
```

RNS ring Element random assignement.
Element is supposed to be initialized

Returns

random ring Element

12.257.2.2 random() [2/2]

```
template<typename RNS>
RNS::Element random () const  [inline], [inherited]
```

12.257.2.3 operator()() [1/2]

```
template<typename RNS>
RNS::Element & operator() (
            typename RNS::Element & elt) const  [inline], [inherited]
```

12.257.2.4 operator()() [2/2]

```
template<typename RNS>
RNS::Element operator() () const  [inline], [inherited]
```

12.257.2.5 ring()

```
template<typename RNS>
const RNS & ring () const  [inline], [inherited]
```

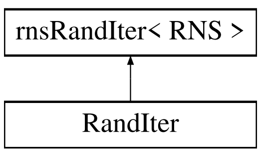
The documentation for this class was generated from the following file:

- rns-integer.h

12.258 RNSIntegerMod< RNS >::RandIter Class Reference

```
#include <rns-integer-mod.h>
```

Inheritance diagram for RNSIntegerMod< RNS >::RandIter:



Public Member Functions

- RandIter (const RNSIntegerMod< RNS > &F, 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

12.258.1 Constructor & Destructor Documentation

12.258.1.1 RandIter()

```
template<typename RNS>
RandIter (
            const RNSIntegerMod< RNS > & F,
            uint64_t seed = 0) [inline]
```

12.258.2 Member Function Documentation

12.258.2.1 random() [1/2]

```
template<typename RNS>
RNS::Element & random (
            typename RNS::Element & elt) const [inline]
```

12.258.2.2 random() [2/2]

```
template<typename RNS>
RNS::Element random () const [inline], [inherited]
```

12.258.2.3 operator()() [1/2]

```
template<typename RNS>
RNS::Element & operator() (
            typename RNS::Element & elt) const [inline], [inherited]
```

12.258.2.4 operator()() [2/2]

```
template<typename RNS>
RNS::Element operator() () const [inline], [inherited]
```

12.258.2.5 ring()

```
template<typename RNS>
const RNS & ring () const [inline], [inherited]
```

The documentation for this class was generated from the following file:

- rns-integer-mod.h

12.259 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)

12.259.1 Member Typedef Documentation

12.259.1.1 Element

```
template<class Field, class T>
typedef Field::Element Element
```

12.259.1.2 Element_ptr

```
template<class Field, class T>
typedef Field::Element_ptr Element_ptr
```

12.259.2 Member Function Documentation

12.259.2.1 operator()()

```
template<class Field, typename T>
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

12.260 readMyMachineType< Field, mpz_t > Struct Template Reference

```
#include <read_sparse.h>
```

Public Types

- typedef Field::Element Element
- typedef Field::Element_ptr Element_ptr
- 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)
- 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)

12.260.1 Member Typedef Documentation

12.260.1.1 Element [1/2]

```
template<class Field>
typedef Field::Element Element
```

12.260.1.2 Element_ptr [1/2]

```
template<class Field>
typedef Field::Element_ptr Element_ptr
```

12.260.1.3 Element [2/2]

```
typedef Field::Element Element
```

12.260.1.4 Element_ptr [2/2]

```
typedef Field::Element_ptr Element_ptr
```

12.260.2 Member Function Documentation

12.260.2.1 operator()() [1/2]

```
template<class Field>
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)
```

12.260.2.2 operator()() [2/2]

```
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

12.261 Recursive Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.262 Recursive Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.263 array< T >::reverse_iterator Class Reference

STL iterator class.

12.263.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.264 string::reverse_iterator Class Reference

STL iterator class.

12.264.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.265 vector< T >::reverse_iterator Class Reference

STL iterator class.

12.265.1 Detailed Description

STL iterator class.
The documentation for this class was generated from the following files:

12.266 rint< K > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

12.267 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 RecInt::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, RecInt::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

12.267.1 Member Typedef Documentation

12.267.1.1 integer

```
typedef Givaro::Integer integer
```

12.267.1.2 ModField

```
typedef Givaro::Modular<double> ModField
```

12.267.1.3 BasisElement

```
typedef double BasisElement
```

12.267.1.4 Element

```
typedef rns_double_elt Element
```

12.267.1.5 Element_ptr

```
typedef rns_double_elt_ptr Element_ptr
```

12.267.1.6 ConstElement_ptr

```
typedef rns_double_elt_cstptr ConstElement_ptr
```

12.267.2 Constructor & Destructor Documentation

12.267.2.1 rns_double() [1/4]

```
rns_double (
            const integer & bound,
            size_t pbits,
            bool rnsmod = false,
            long seed = time(NULL)) [inline]
```

12.267.2.2 rns_double() [2/4]

```
rns_double (
            size_t pbits,
            size_t size,
            long seed = time(NULL)) [inline]
```

12.267.2.3 rns_double() [3/4]

```
template<typename Vect>
rns_double (
            const Vect & basis,
            bool rnsmod = false,
            long seed = time(NULL)) [inline]
```

12.267.2.4 rns_double() [4/4]

```
rns_double (
            const RNSIntegerMod< rns_double > & basis,
            bool rnsmod = false,
            long seed = time(NULL)) [inline]
```

12.267.3 Member Function Documentation

12.267.3.1 precompute_cst()

```
void precompute_cst (
            size_t K = 0) [inline]
```

12.267.3.2 init() [1/3]

```
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 [inline]
```

12.267.3.3 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]
```

12.267.3.4 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]
```

12.267.3.5 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]
```

12.267.3.6 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]
```

12.267.3.7 reduce()

```
void reduce (
            size_t n,
            double * Arns,
            size_t rda,
            bool RNS_MAJOR = false) const  [inline]
```

12.267.3.8 init() [3/3]

```
template<size_t K>
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]
```

12.267.3.9 convert() [2/2]

```
template<size_t K>
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]
```

12.267.4 Field Documentation

12.267.4.1 _basis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

12.267.4.2 _basisMax

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

12.267.4.3 _negbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

12.267.4.4 _invbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis
```

12.267.4.5 _field_rns

```
std::vector<ModField> _field_rns
```

12.267.4.6 _M

```
integer _M
```

12.267.4.7 _Mi

```
std::vector<integer> _Mi
```

12.267.4.8 _MMi

```
std::vector<double> _MMi
```

12.267.4.9 _crt_in

```
std::vector<double> _crt_in
```

12.267.4.10 _crt_out

```
std::vector<double> _crt_out
```

12.267.4.11 _size

```
size_t _size
```

12.267.4.12 _pbits

```
size_t _pbits
```

12.267.4.13 _ldm

```
size_t _ldm
```

12.267.4.14 _mi_sum

```
integer _mi_sum
```

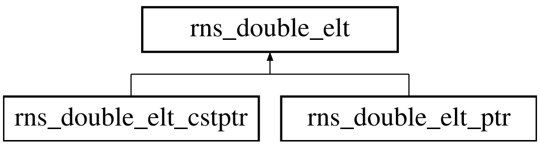
The documentation for this struct was generated from the following files:

- rns-double.h
- rns-double-recint.inl
- rns-double.inl

12.268 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

12.268.1 Constructor & Destructor Documentation

12.268.1.1 rns_double_elt() [1/3]

```
rns_double_elt () [inline]
```

12.268.1.2 ∼rns_double_elt()

```
∼rns_double_elt () [inline]
```

12.268.1.3 rns_double_elt() [2/3]

```
rns_double_elt (
            double * p,
            size_t r,
            size_t a = false) [inline]
```

12.268.1.4 rns_double_elt() [3/3]

```
rns_double_elt (
            const rns_double_elt & x) [inline]
```

12.268.2 Member Function Documentation

12.268.2.1 operator&() [1/2]

```
rns_double_elt_ptr operator& () [inline]
```

12.268.2.2 operator&() [2/2]

```
rns_double_elt_cstptr operator& () const [inline]
```

12.268.3 Field Documentation

12.268.3.1 _ptr

```
double* _ptr
```

12.268.3.2 _stride

```
size_t _stride
```

12.268.3.3 _alloc

```
bool _alloc
```

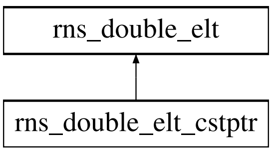
The documentation for this struct was generated from the following file:

- rns-double-elt.h

12.269 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

12.269.1 Constructor & Destructor Documentation

12.269.1.1 rns_double_elt_cstptr() [1/5]

```
rns_double_elt_cstptr ( ) [inline]
```

12.269.1.2 rns_double_elt_cstptr() [2/5]

```
rns_double_elt_cstptr (
            double * p,
            size_t r ) [inline]
```

12.269.1.3 rns_double_elt_cstptr() [3/5]

```
rns_double_elt_cstptr (
            const rns_double_elt_ptr & x ) [inline]
```

12.269.1.4 rns_double_elt_cstptr() [4/5]

```
rns_double_elt_cstptr (
            const rns_double_elt_cstptr & x ) [inline]
```

12.269.1.5 rns_double_elt_cstptr() [5/5]

```
rns_double_elt_cstptr (
            rns_double_elt_cstptr && ) [default]
```

12.269.2 Member Function Documentation

12.269.2.1 operator&() [1/2]

```
rns_double_elt_cstptr * operator& ( ) [inline]
```

12.269.2.2 operator∗()

```
rns_double_elt & operator* ( ) const [inline]
```

12.269.2.3 operator[]() [1/2]

```
rns_double_elt operator[] (
            size_t i) const [inline]
```

12.269.2.4 operator[]() [2/2]

```
rns_double_elt & operator[] (
            size_t i) [inline]
```

12.269.2.5 operator++()

```
rns_double_elt_cstptr operator++ () [inline]
```

12.269.2.6 operator--()

```
rns_double_elt_cstptr operator-- () [inline]
```

12.269.2.7 operator+()

```
rns_double_elt_cstptr operator+ (
            size_t inc) const [inline]
```

12.269.2.8 operator-()

```
rns_double_elt_cstptr operator- (
            size_t inc) const [inline]
```

12.269.2.9 operator+=()

```
rns_double_elt_cstptr & operator+= (
            size_t inc) [inline]
```

12.269.2.10 operator-=()

```
rns_double_elt_cstptr & operator-= (
            size_t inc) [inline]
```

12.269.2.11 operator=()

```
rns_double_elt_cstptr & operator= (
            const rns_double_elt_cstptr & x) [inline]
```

12.269.2.12 operator<()

```
bool operator< (
            const rns_double_elt_cstptr & x) [inline]
```

12.269.2.13 operator"!=()

```
bool operator!= (
            const rns_double_elt_cstptr & x) [inline]
```

12.269.2.14 operator&() [2/2]

```
rns_double_elt_cstptr operator& () const [inline], [inherited]
```

12.269.3 Field Documentation

12.269.3.1 other

`rns_double_elt other`

12.269.3.2 _ptr

`double* _ptr [inherited]`

12.269.3.3 _stride

`size_t _stride [inherited]`

12.269.3.4 _alloc

`bool _alloc [inherited]`

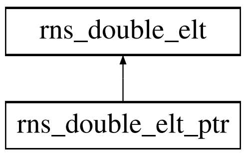
The documentation for this struct was generated from the following file:

- rns-double-elt.h

12.270 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

12.270.1 Constructor & Destructor Documentation

12.270.1.1 rns_double_elt_ptr() [1/5]

```
rns_double_elt_ptr ( ) [inline]
```

12.270.1.2 rns_double_elt_ptr() [2/5]

```
rns_double_elt_ptr (
            double * p,
            size_t r ) [inline]
```

12.270.1.3 rns_double_elt_ptr() [3/5]

```
rns_double_elt_ptr (
            const rns_double_elt_ptr & x ) [inline]
```

12.270.1.4 rns_double_elt_ptr() [4/5]

```
rns_double_elt_ptr (
            const rns_double_elt_cstptr & x ) [inline]
```

12.270.1.5 rns_double_elt_ptr() [5/5]

```
rns_double_elt_ptr (
            rns_double_elt_ptr && ) [default]
```

12.270.2 Member Function Documentation

12.270.2.1 operator&() [1/2]

```
rns_double_elt_ptr * operator& ( ) [inline]
```

12.270.2.2 operator*()

```
rns_double_elt & operator* ( ) [inline]
```

12.270.2.3 operator[]() [1/2]

```
rns_double_elt operator[ ] (
            size_t i ) const [inline]
```

12.270.2.4 operator[]() [2/2]

```
rns_double_elt & operator[ ] (
            size_t i ) [inline]
```

12.270.2.5 operator++()

```
rns_double_elt_ptr operator++ ( ) [inline]
```

12.270.2.6 operator--()

```
rns_double_elt_ptr operator-- ( ) [inline]
```

12.270.2.7 operator+()

```
rns_double_elt_ptr operator+ (
            size_t inc) [inline]
```

12.270.2.8 operator-()

```
rns_double_elt_ptr operator- (
            size_t inc) [inline]
```

12.270.2.9 operator+=()

```
rns_double_elt_ptr & operator+= (
            size_t inc) [inline]
```

12.270.2.10 operator-=()

```
rns_double_elt_ptr & operator-= (
            size_t inc) [inline]
```

12.270.2.11 operator=()

```
rns_double_elt_ptr & operator= (
            const rns_double_elt_ptr & x) [inline]
```

12.270.2.12 operator<()

```
bool operator< (
            const rns_double_elt_ptr & x) [inline]
```

12.270.2.13 operator"!=()

```
bool operator!= (
            const rns_double_elt_ptr & x) [inline]
```

12.270.2.14 operator&() [2/2]

```
rns_double_elt_cstptr operator& () const [inline], [inherited]
```

12.270.3 Field Documentation

12.270.3.1 other

```
rns_double_elt other
```

12.270.3.2 _ptr

```
double* _ptr [inherited]
```

12.270.3.3 _stride

```
size_t _stride [inherited]
```

12.270.3.4 _alloc

```
bool _alloc [inherited]
```

The documentation for this struct was generated from the following file:

- rns-double-elt.h

12.271 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

12.271.1 Member Typedef Documentation

12.271.1.1 integer

`typedef Givaro::Integer integer`

12.271.1.2 ModField

`typedef Givaro::ModularExtended<double> ModField`

12.271.1.3 BasisElement

```
typedef double BasisElement
```

12.271.1.4 Element

```
typedef rns_double_elt Element
```

12.271.1.5 Element_ptr

```
typedef rns_double_elt_ptr Element_ptr
```

12.271.1.6 ConstElement_ptr

```
typedef rns_double_elt_cstptr ConstElement_ptr
```

12.271.2 Constructor & Destructor Documentation

12.271.2.1 rns_double_extended() [1/3]

```
rns_double_extended (
            const integer & bound,
            size_t pbits,
            bool rnsmod = false,
            long seed = time(NULL))  [inline]
```

12.271.2.2 rns_double_extended() [2/3]

```
rns_double_extended (
            size_t pbits,
            size_t size,
            long seed = time(NULL))  [inline]
```

12.271.2.3 rns_double_extended() [3/3]

```
template<typename Vect>
rns_double_extended (
            const Vect & basis,
            bool rnsmod = false,
            long seed = time(NULL))  [inline]
```

12.271.3 Member Function Documentation

12.271.3.1 precompute_cst()

```
void precompute_cst ()  [inline]
```

12.271.3.2 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]
```

12.271.3.3 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]
```

12.271.3.4 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]
```

12.271.3.5 init() [3/3]

```
void init (
            size_t m,
            double * Arns,
            const integer * A,
            size_t lda) const  [inline]
```

12.271.3.6 convert() [2/2]

```
void convert (
            size_t m,
            integer * A,
            const double * Arns) const  [inline]
```

12.271.3.7 reduce()

```
void reduce (
            size_t n,
            double * Arns,
            size_t rda,
            bool RNS_MAJOR = false) const  [inline]
```

12.271.4 Field Documentation

12.271.4.1 _basis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basis
```

12.271.4.2 _basisMax

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _basisMax
```

12.271.4.3 _negbasis

```
std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _negbasis
```

12.271.4.4 _invbasis

`std::vector<double, AlignedAllocator<double, Alignment::CACHE_LINE> > _invbasis`

12.271.4.5 _field_rns

`std::vector<ModField> _field_rns`

12.271.4.6 _M

`integer _M`

12.271.4.7 _Mi

`std::vector<integer> _Mi`

12.271.4.8 _MMi

`std::vector<double> _MMi`

12.271.4.9 _crt_in

`std::vector<double> _crt_in`

12.271.4.10 _crt_out

`std::vector<double> _crt_out`

12.271.4.11 _size

`size_t _size`

12.271.4.12 _pbits

`size_t _pbits`

12.271.4.13 _ldm

`size_t _ldm`

The documentation for this struct was generated from the following files:

- rns-double.h
- rns-double.inl

12.272 RNSElementTag Struct Reference

Representation in a Residue Number System.

`#include <field-traits.h>`

12.272.1 Detailed Description

Representation in a Residue Number System.

The documentation for this struct was generated from the following file:

- field-traits.h

12.273 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

12.273.1 Member Typedef Documentation

12.273.1.1 BasisElement

```
template<typename RNS>
typedef RNS::BasisElement BasisElement  [protected]
```

12.273.1.2 integer

```
template<typename RNS>
typedef Givaro::Integer integer  [protected]
```

12.273.1.3 Element

```
template<typename RNS>
typedef RNS::Element Element
```

12.273.1.4 Element_ptr

```
template<typename RNS>
typedef RNS::Element_ptr Element_ptr
```

12.273.1.5 ConstElement_ptr

```
template<typename RNS>
typedef RNS::ConstElement_ptr ConstElement_ptr
```

12.273.2 Constructor & Destructor Documentation

12.273.2.1 RNSInteger() [1/2]

```
template<typename RNS>
RNSInteger (
            const RNS & myrns)  [inline]
```

12.273.2.2 RNSInteger() [2/2]

```
template<typename RNS>
template<typename T>
RNSInteger (
            const T & F)  [inline]
```

12.273.3 Member Function Documentation

12.273.3.1 rns()

```
template<typename RNS>
const RNS & rns () const  [inline]
```

12.273.3.2 size()

```
template<typename RNS>
size_t size () const  [inline]
```

12.273.3.3 isOne()

```
template<typename RNS>
bool isOne (
            const Element & x) const  [inline]
```

12.273.3.4 isMOne()

```
template<typename RNS>
bool isMOne (
            const Element & x) const  [inline]
```

12.273.3.5 isZero()

```
template<typename RNS>
bool isZero (
            const Element & x) const  [inline]
```

12.273.3.6 characteristic()

```
template<typename RNS>
integer characteristic (
            integer & p) const  [inline]
```

12.273.3.7 cardinality()

```
template<typename RNS>
integer cardinality (
            integer & p) const  [inline]
```

12.273.3.8 init() [1/2]

```
template<typename RNS>
Element & init (
            Element & x) const  [inline]
```

12.273.3.9 init() [2/2]

```
template<typename RNS>
Element & init (
            Element & x,
            const Givaro::Integer & y) const  [inline]
```

12.273.3.10 reduce() [1/2]

```
template<typename RNS>
Element & reduce (
            Element & x,
            const Element & y) const  [inline]
```

12.273.3.11 reduce() [2/2]

```
template<typename RNS>
Element & reduce (
            Element & x) const  [inline]
```

12.273.3.12 convert()

```
template<typename RNS>
Givaro::Integer convert (
            Givaro::Integer & x,
            const Element & y) const  [inline]
```

12.273.3.13 assign()

```
template<typename RNS>
Element & assign (
            Element & x,
            const Element & y) const  [inline]
```

12.273.3.14 write() [1/2]

```
template<typename RNS>
std::ostream & write (
            std::ostream & os,
            const Element & y) const  [inline]
```

12.273.3.15 write() [2/2]

```
template<typename RNS>
std::ostream & write (
            std::ostream & os) const  [inline]
```

12.273.4 Field Documentation

12.273.4.1 _rns

```
template<typename RNS>
const RNS* _rns  [protected]
```

12.273.4.2 one

```
template<typename RNS>
Element one
```

12.273.4.3 mOne

```
template<typename RNS>
Element mOne
```

12.273.4.4 zero

```
template<typename RNS>
Element zero
```

The documentation for this class was generated from the following files:

- field-traits.h
- rns-integer.h

12.274 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

12.274.1 Member Typedef Documentation

12.274.1.1 Element

```
template<typename RNS>
typedef RNS::Element Element
```

12.274.1.2 Element_ptr

```
template<typename RNS>
typedef RNS::Element_ptr Element_ptr
```

12.274.1.3 ConstElement_ptr

```
template<typename RNS>
typedef RNS::ConstElement_ptr ConstElement_ptr
```

12.274.1.4 BasisElement

```
template<typename RNS>
typedef RNS::BasisElement BasisElement  [protected]
```

12.274.1.5 ModField

```
template<typename RNS>
typedef Givaro::Modular<BasisElement> ModField  [protected]
```

12.274.1.6 integer

```
template<typename RNS>
typedef Givaro::Integer integer  [protected]
```

12.274.2 Constructor & Destructor Documentation

12.274.2.1 RNSIntegerMod()

```
template<typename RNS>
RNSIntegerMod (
            const integer & p,
            const RNS & myrns)  [inline]
```

12.274.3 Member Function Documentation

12.274.3.1 rns()

```
template<typename RNS>
const rns_double & rns () const  [inline]
```

12.274.3.2 delayed()

```
template<typename RNS>
const RNSInteger< RNS > & delayed () const  [inline]
```

12.274.3.3 size()

```
template<typename RNS>
size_t size () const  [inline]
```

12.274.3.4 isOne()

```
template<typename RNS>
bool isOne (
            const Element & x) const  [inline]
```

12.274.3.5 isMOne()

```
template<typename RNS>
bool isMOne (
            const Element & x) const  [inline]
```

12.274.3.6 isZero()

```
template<typename RNS>
bool isZero (
            const Element & x) const  [inline]
```

12.274.3.7 characteristic() [1/2]

```
template<typename RNS>
integer & characteristic (
            integer & p) const  [inline]
```

12.274.3.8 characteristic() [2/2]

```
template<typename RNS>
integer characteristic () const  [inline]
```

12.274.3.9 cardinality() [1/2]

```
template<typename RNS>
integer & cardinality (
            integer & p) const  [inline]
```

12.274.3.10 cardinality() [2/2]

```
template<typename RNS>
integer cardinality () const  [inline]
```

12.274.3.11 minElement()

```
template<typename RNS>
integer minElement () const  [inline]
```

12.274.3.12 maxElement()

```
template<typename RNS>
integer maxElement () const  [inline]
```

12.274.3.13 init() [1/3]

```
template<typename RNS>
Element & init (
            Element & x) const  [inline]
```

12.274.3.14 init() [2/3]

```
template<typename RNS>
Element & init (
            Element & x,
            const Givaro::Integer & y) const  [inline]
```

12.274.3.15 reduce() [1/2]

```
template<typename RNS>
Element & reduce (
            Element & x,
            const Element & y) const  [inline]
```

12.274.3.16 reduce() [2/2]

```
template<typename RNS>
Element & reduce (
            Element & x) const  [inline]
```

12.274.3.17 init() [3/3]

```
template<typename RNS>
Element & init (
            Element & x,
            const Element & y) const  [inline]
```

12.274.3.18 convert()

```
template<typename RNS>
Givaro::Integer convert (
            Givaro::Integer & x,
            const Element & y) const  [inline]
```

12.274.3.19 assign()

```
template<typename RNS>
Element & assign (
            Element & x,
            const Element & y) const  [inline]
```

12.274.3.20 add()

```
template<typename RNS>
Element & add (
            Element & x,
            const Element & y,
            const Element & z) const  [inline]
```

12.274.3.21 sub()

```
template<typename RNS>
Element & sub (
            Element & x,
            const Element & y,
            const Element & z) const  [inline]
```

12.274.3.22 neg()

```
template<typename RNS>
Element & neg (
            Element & x,
            const Element & y) const  [inline]
```

12.274.3.23 mul()

```
template<typename RNS>
Element & mul (
            Element & x,
            const Element & y,
            const Element & z) const  [inline]
```

12.274.3.24 axpyin()

```
template<typename RNS>
Element & axpyin (
            Element & x,
            const Element & y,
            const Element & z) const  [inline]
```

12.274.3.25 inv()

```
template<typename RNS>
Element & inv (
            Element & x,
            const Element & y) const  [inline]
```

12.274.3.26 areEqual()

```
template<typename RNS>
bool areEqual (
            const Element & x,
            const Element & y) const  [inline]
```

12.274.3.27 write() [1/2]

```
template<typename RNS>
std::ostream & write (
            std::ostream & os,
            const Element & y) const  [inline]
```

12.274.3.28 write() [2/2]

```
template<typename RNS>
std::ostream & write (
            std::ostream & os) const  [inline]
```

12.274.3.29 reduce_modp() [1/2]

```
template<typename RNS>
void reduce_modp (
            size_t n,
            Element_ptr B) const  [inline]
```

12.274.3.30 write_matrix()

```
template<typename RNS>
std::ostream & write_matrix (
            std::ostream & c,
            const double * E,
            int n,
            int m,
            int lda) const  [inline]
```

12.274.3.31 write_matrix_long()

```
template<typename RNS>
std::ostream & write_matrix_long (
            std::ostream & c,
            const double * E,
            int n,
```

```
int m,
int lda) const [inline]
```

12.274.3.32 reduce_modp() [2/2]

```
template<typename RNS>
void reduce_modp (
            size_t m,
            size_t n,
            Element_ptr B,
            size_t lda) const [inline]
```

12.274.3.33 reduce_modp_rnsmajor()

```
template<typename RNS>
void reduce_modp_rnsmajor (
            size_t n,
            Element_ptr B) const [inline]
```

12.274.4 Field Documentation

12.274.4.1 _p

```
template<typename RNS>
integer _p [protected]
```

12.274.4.2 _Mi_modp_rns

```
template<typename RNS>
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _Mi_modp_↩
rns [protected]
```

12.274.4.3 _iM_modp_rns

```
template<typename RNS>
std::vector<BasisElement, AlignedAllocator<BasisElement, Alignment::CACHE_LINE> > _iM_modp_↩
rns [protected]
```

12.274.4.4 _rns

```
template<typename RNS>
const RNS* _rns [protected]
```

12.274.4.5 _F

```
template<typename RNS>
Givaro::Modular<Givaro::Integer> _F [protected]
```

12.274.4.6 _RNSdelayed

```
template<typename RNS>
RNSInteger<RNS> _RNSdelayed [protected]
```

12.274.4.7 one

```
template<typename RNS>
Element one
```

12.274.4.8 mOne

```
template<typename RNS>
Element mOne
```

12.274.4.9 zero

```
template<typename RNS>
Element zero
```

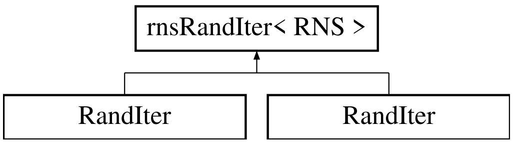
The documentation for this class was generated from the following files:

- field-traits.h
- rns-integer-mod.h

12.275 rnsRandIter< RNS > Class Template Reference

```
#include <rns-double.h>
```

Inheritance diagram for rnsRandIter< RNS >:



Public Member Functions

- rnsRandIter (const RNS &R, 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

12.275.1 Constructor & Destructor Documentation

12.275.1.1 rnsRandIter()

```
template<typename RNS>
rnsRandIter (
        const RNS & R,
        uint64_t seed = 0) [inline]
```

12.275.2 Member Function Documentation

12.275.2.1 random() [1/2]

```
template<typename RNS>
RNS::Element & random (
        typename RNS::Element & elt) const [inline]
```

RNS ring Element random assignement.

Element is supposed to be initialized

Returns

random ring Element

12.275.2.2 operator()() [1/2]

```
template<typename RNS>
RNS::Element & operator() (
            typename RNS::Element & elt) const [inline]
```

12.275.2.3 operator()() [2/2]

```
template<typename RNS>
RNS::Element operator() () const [inline]
```

12.275.2.4 random() [2/2]

```
template<typename RNS>
RNS::Element random () const [inline]
```

12.275.2.5 ring()

```
template<typename RNS>
const RNS & ring () const [inline]
```

The documentation for this class was generated from the following file:

- rns-double.h

12.276 Row Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

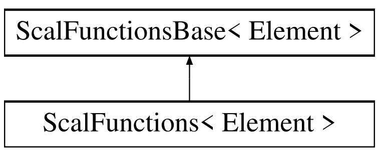
12.277 ruint< K > Class Template Reference

The documentation for this class was generated from the following file:

- field-traits.h

12.278 ScalFunctions< Element > Struct Template Reference

Inheritance diagram for ScalFunctions< Element >:



Public Types

- using vectElt = vector<Element>

Static Public Member Functions

- static void genInputs (vector< vectElt > &inputs)
- static void genInputsWithZero (vector< vectElt > &inputs)
- 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 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)
- static vectElt unpacklo (vectElt a, vectElt b)
- static vectElt unpackhi (vectElt a, vectElt b)
- static void unpacklohi (vectElt &lo, vectElt &hi, vectElt a, vectElt b)
- static vectElt pack_even (vectElt a, vectElt b)
- static vectElt pack_odd (vectElt a, vectElt b)
- static void pack (vectElt &even, vectElt &odd, vectElt a, vectElt b)
- template<uint16_t s>
  static vectElt blend (vectElt a, vectElt b)

12.278.1 Member Typedef Documentation

12.278.1.1 vectElt

```
template<typename Element>
using vectElt = vector<Element>
```

12.278.2 Member Function Documentation

12.278.2.1 genInputs()

```
template<typename Element>
static void genInputs (
            vector< vectElt > & inputs)  [inline], [static]
```

12.278.2.2 genInputsWithZero()

```
template<typename Element>
static void genInputsWithZero (
            vector< vectElt > & inputs)  [inline], [static]
```

12.278.2.3 zero()

```
template<typename Element>
static Element zero ()  [inline], [static]
```

12.278.2.4 vand()

```
template<typename Element>
static Element vand (
            Element x1,
            Element x2) [inline], [static]
```

12.278.2.5 vor()

```
template<typename Element>
static Element vor (
            Element x1,
            Element x2) [inline], [static]
```

12.278.2.6 vxor()

```
template<typename Element>
static Element vxor (
            Element x1,
            Element x2) [inline], [static]
```

12.278.2.7 vandnot()

```
template<typename Element>
static Element vandnot (
            Element x1,
            Element x2) [inline], [static]
```

12.278.2.8 add()

```
template<typename Element>
static Element add (
            Element x1,
            Element x2) [inline], [static]
```

12.278.2.9 addin()

```
template<typename Element>
static Element addin (
            Element & x1,
            Element x2) [inline], [static]
```

12.278.2.10 sub()

```
template<typename Element>
static Element sub (
            Element x1,
            Element x2) [inline], [static]
```

12.278.2.11 subin()

```
template<typename Element>
static Element subin (
            Element & x1,
            Element x2) [inline], [static]
```

12.278.2.12 mul()

```
template<typename Element>
static Element mul (
```

```
            Element x1,
            Element x2)  [inline], [static]
```

12.278.2.13 mulin()

```
template<typename Element>
static Element mulin (
            Element & x1,
            Element x2)  [inline], [static]
```

12.278.2.14 div()

```
template<typename Element>
static Element div (
            Element x1,
            Element x2)  [inline], [static]
```

12.278.2.15 fmadd()

```
template<typename Element>
static Element fmadd (
            Element x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.278.2.16 fmaddin()

```
template<typename Element>
static Element fmaddin (
            Element & x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.278.2.17 fmsub()

```
template<typename Element>
static Element fmsub (
            Element x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.278.2.18 fmsubin()

```
template<typename Element>
static Element fmsubin (
            Element & x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.278.2.19 fnmadd()

```
template<typename Element>
static Element fnmadd (
            Element x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.278.2.20 fnmaddin()

```
template<typename Element>
static Element fnmaddin (
            Element & x1,
            Element x2,
            Element x3) [inline], [static]
```

12.278.2.21 lesser()

```
template<typename Element>
static Element lesser (
            Element x1,
            Element x2) [inline], [static]
```

12.278.2.22 lesser_eq()

```
template<typename Element>
static Element lesser_eq (
            Element x1,
            Element x2) [inline], [static]
```

12.278.2.23 greater()

```
template<typename Element>
static Element greater (
            Element x1,
            Element x2) [inline], [static]
```

12.278.2.24 greater_eq()

```
template<typename Element>
static Element greater_eq (
            Element x1,
            Element x2) [inline], [static]
```

12.278.2.25 eq()

```
template<typename Element>
static Element eq (
            Element x1,
            Element x2) [inline], [static]
```

12.278.2.26 unpacklo()

```
template<typename Element>
static vectElt unpacklo (
            vectElt a,
            vectElt b) [inline], [static]
```

12.278.2.27 unpackhi()

```
template<typename Element>
static vectElt unpackhi (
            vectElt a,
            vectElt b) [inline], [static]
```

12.278.2.28 unpacklohi()

```
template<typename Element>
static void unpacklohi (
            vectElt & lo,
            vectElt & hi,
            vectElt a,
            vectElt b) [inline], [static]
```

12.278.2.29 pack_even()

```
template<typename Element>
static vectElt pack_even (
            vectElt a,
            vectElt b) [inline], [static]
```

12.278.2.30 pack_odd()

```
template<typename Element>
static vectElt pack_odd (
            vectElt a,
            vectElt b) [inline], [static]
```

12.278.2.31 pack()

```
template<typename Element>
static void pack (
            vectElt & even,
            vectElt & odd,
            vectElt a,
            vectElt b) [inline], [static]
```

12.278.2.32 blend()

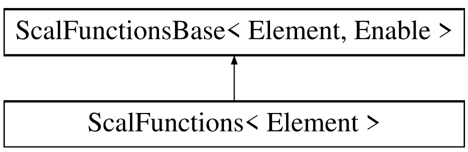
```
template<typename Element>
template<uint16_t s>
static vectElt blend (
            vectElt a,
            vectElt b) [inline], [static]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.279 ScalFunctionsBase< Element, Enable > Struct Template Reference

Inheritance diagram for ScalFunctionsBase< Element, Enable >:



The documentation for this struct was generated from the following file:

- test-simd.C

12.280 ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type > Struct Template Reference

Data Structures

- class FloatingPointTestDistribution

Static Public Member Functions

- static FloatingPointTestDistribution get_default_random_generator ()
- static Element ceil (Element x)
- static Element floor (Element x)
- static Element round (Element x)
- static Element blendv (Element a, Element b, Element mask)
- static Element fma (Element x, Element y, Element z)

Static Public Attributes

- static constexpr Element _zero = 0.0
- static constexpr Element cmp_true = NAN
- static constexpr Element cmp_false = _zero

12.280.1 Member Function Documentation

12.280.1.1 get_default_random_generator()

```
template<typename Element>
static FloatingPointTestDistribution get_default_random_generator ()  [inline], [static]
```

12.280.1.2 ceil()

```
template<typename Element>
static Element ceil (
            Element x)  [inline], [static]
```

12.280.1.3 floor()

```
template<typename Element>
static Element floor (
            Element x)  [inline], [static]
```

12.280.1.4 round()

```
template<typename Element>
static Element round (
            Element x)  [inline], [static]
```

12.280.1.5 blendv()

```
template<typename Element>
static Element blendv (
            Element a,
            Element b,
            Element mask)  [inline], [static]
```

12.280.1.6 fma()

```
template<typename Element>
static Element fma (
            Element x,
            Element y,
            Element z)  [inline], [static]
```

12.280.2 Field Documentation

12.280.2.1 _zero

```
template<typename Element>
Element _zero = 0.0  [static], [constexpr]
```

12.280.2.2 cmp_true

```
template<typename Element>
Element cmp_true = NAN  [static], [constexpr]
```

12.280.2.3 cmp_false

```
template<typename Element>
Element cmp_false = _zero  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.281 ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type > Struct Template Reference

Static Public Member Functions

- static uniform_int_distribution< Element > get_default_random_generator ()
- static Element round (Element x)
- static Element fma (Element x, Element y, Element z)
- static Element mullo (Element x1, Element x2)
- static Element mulhi (Element x1, Element x2)
- static Element mulx (Element x1, Element x2)
- static Element fmaddx (Element x1, Element x2, Element x3)
- static Element fmaddxin (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 fnmaddx (Element x1, Element x2, Element x3)
- static Element fnmaddxin (Element &x1, Element x2, Element x3)
- template< int s >
  static Element sra (Element x1)
- template< int s >
  static Element srl (Element x1)
- template< int s >
  static Element sll (Element x1)

Static Public Attributes

- static constexpr Element _zero = 0
- static constexpr Element cmp_true = -1
- static constexpr Element cmp_false = _zero

12.281.1 Member Function Documentation

12.281.1.1 get_default_random_generator()

```
template<typename Element>
static uniform_int_distribution< Element > get_default_random_generator ()  [inline], [static]
```

12.281.1.2 round()

```
template<typename Element>
static Element round (
            Element x)  [inline], [static]
```

12.281.1.3 fma()

```
template<typename Element>
static Element fma (
            Element x,
            Element y,
            Element z)  [inline], [static]
```

12.281.1.4 mullo()

```
template<typename Element>
static Element mullo (
            Element x1,
            Element x2)  [inline], [static]
```

12.281.1.5 mulhi()

```
template<typename Element>
static Element mulhi (
            Element x1,
            Element x2)  [inline], [static]
```

12.281.1.6 mulx()

```
template<typename Element>
static Element mulx (
            Element x1,
            Element x2)  [inline], [static]
```

12.281.1.7 fmaddx()

```
template<typename Element>
static Element fmaddx (
            Element x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.281.1.8 fmaddxin()

```
template<typename Element>
static Element fmaddxin (
            Element & x1,
            Element x2,
            Element x3)  [inline], [static]
```

12.281.1.9 fmsubx()

```
template<typename Element>
static Element fmsubx (
            Element x1,
            Element x2,
            Element x3) [inline], [static]
```

12.281.1.10 fmsubxin()

```
template<typename Element>
static Element fmsubxin (
            Element & x1,
            Element x2,
            Element x3) [inline], [static]
```

12.281.1.11 fnmaddx()

```
template<typename Element>
static Element fnmaddx (
            Element x1,
            Element x2,
            Element x3) [inline], [static]
```

12.281.1.12 fnmaddxin()

```
template<typename Element>
static Element fnmaddxin (
            Element & x1,
            Element x2,
            Element x3) [inline], [static]
```

12.281.1.13 sra()

```
template<typename Element>
template<int s>
static Element sra (
            Element x1) [inline], [static]
```

12.281.1.14 srl()

```
template<typename Element>
template<int s>
static Element srl (
            Element x1) [inline], [static]
```

12.281.1.15 sll()

```
template<typename Element>
template<int s>
static Element sll (
            Element x1) [inline], [static]
```

12.281.2 Field Documentation

12.281.2.1 _zero

```
template<typename Element>
Element _zero = 0 [static], [constexpr]
```

12.281.2.2 cmp_true

```
template<typename Element>
Element cmp_true = -1  [static], [constexpr]
```

12.281.2.3 cmp_false

```
template<typename Element>
Element cmp_false = _zero  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.282 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 &)

12.282.1 Constructor & Destructor Documentation

12.282.1.1 Sequential() [1/3]

```
Sequential ()  [inline]
```

12.282.1.2 Sequential() [2/3]

```
Sequential (
            size_t nth)  [inline]
```

12.282.1.3 Sequential() [3/3]

```
template<class Cut, class Param>
Sequential (
            Parallel< Cut, Param > & )  [inline]
```

12.282.2 Member Function Documentation

12.282.2.1 numthreads()

```
size_t numthreads () const  [inline]
```

12.282.3 Friends And Related Symbol Documentation

12.282.3.1 operator<<

```
std::ostream & operator<< (
            std::ostream & out,
            const Sequential & )  [friend]
```

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.283 Simd128_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- simd128.inl

12.284 Simd128_impl< true, false, true, 4 > Struct Reference

The documentation for this struct was generated from the following file:

- simd128_float.inl

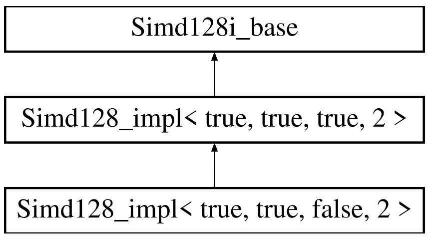
12.285 Simd128_impl< true, false, true, 8 > Struct Reference

The documentation for this struct was generated from the following file:

- simd128_double.inl

12.286 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 aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using vect_t = __m128i

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- 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 __m64 &INVP, const vect_t &NEGP, const vect_t &MIN, const vect_t &MAX, vect_t &Q, vect_t &T)
- 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

12.286.1 Member Typedef Documentation

12.286.1.1 scalar_t

```
using scalar_t = uint16_t
```

12.286.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.286.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.286.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.286.1.5 vect_t

```
using vect_t = __m128i  [inherited]
```

12.286.2 Member Function Documentation

12.286.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.286.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.286.2.3 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]
```

12.286.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx) [inline], [static]
```

12.286.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p) [inline], [static]
```

12.286.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p) [inline], [static]
```

12.286.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.286.2.8 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.286.2.9 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v) [inline], [static]
```

12.286.2.10 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a) [inline], [static]
```

12.286.2.11 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b) [inline], [static]
```

12.286.2.12 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.286.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.15 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.16 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.17 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.18 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.19 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.20 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.21 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.22 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.286.2.23 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.286.2.24 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr], [inherited]
```

12.286.2.25 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr], [inherited]
```

12.286.2.26 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static], [inherited]
```

12.286.2.27 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]
```

12.286.2.28 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.286.2.29 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.286.2.30 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.286.2.31 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.286.2.32 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.286.2.33 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.286.2.34 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.35 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.36 shuffle()

```
template<uint32_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.37 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.38 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.39 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.40 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.41 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.42 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.43 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.44 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.45 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static], [inherited]
```

12.286.2.46 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.47 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.48 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.2.49 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.2.50 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.2.51 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.2.52 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.2.53 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.2.54 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.2.55 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.2.56 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.2.57 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.58 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.59 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.60 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.61 mod()

```
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)  [inline], [static], [inherited]
```

12.286.2.62 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.286.2.63 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.64 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.286.2.65 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.286.2.66 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.2.67 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.2.68 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.286.3 Field Documentation

12.286.3.1 vect_size

```
const constexpr size_t vect_size = 8 [static], [constexpr], [inherited]
```

12.286.3.2 alignment

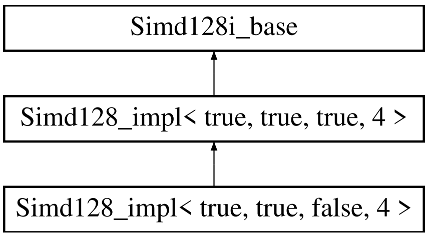
```
const constexpr size_t alignment = 16 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd128_int16.inl

12.287 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 aligned_allocator = AlignedAllocator< scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector< scalar_t, aligned_allocator >
- template< class Field >
  using is_same_element = std::is_same< typename Field::Element, scalar_t >
- using vect_t = __m128i

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3)

- 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 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

12.287.1 Member Typedef Documentation

12.287.1.1 scalar_t

```
using scalar_t = uint32_t
```

12.287.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.287.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.287.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.287.1.5 vect_t

```
using vect_t = __m128i  [inherited]
```

12.287.2 Member Function Documentation

12.287.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.287.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.287.2.3 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]
```

12.287.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.287.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.287.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.287.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.287.2.8 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.287.2.9 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.287.2.10 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.287.2.11 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.287.2.12 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.287.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.15 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.16 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.17 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.18 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.19 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.20 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.21 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.22 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.287.2.23 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.287.2.24 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr], [inherited]
```

12.287.2.25 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr], [inherited]
```

12.287.2.26 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static], [inherited]
```

12.287.2.27 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]
```

12.287.2.28 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.287.2.29 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.287.2.30 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.287.2.31 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.287.2.32 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.287.2.33 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.287.2.34 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.35 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.36 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.37 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.38 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.39 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.287.2.40 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.287.2.41 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.287.2.42 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.287.2.43 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.287.2.44 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.287.2.45 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3) [inline], [static], [inherited]
```

12.287.2.46 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.287.2.47 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.287.2.48 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.49 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.50 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.51 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.52 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.53 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.54 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.55 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.56 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.57 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.58 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.59 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.60 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.61 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]
```

12.287.2.62 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.287.2.63 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.64 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.287.2.65 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.66 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.67 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.2.68 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.287.3 Field Documentation

12.287.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr], [inherited]
```

12.287.3.2 alignment

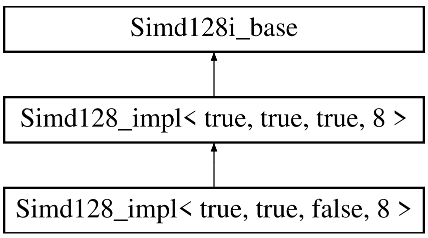
```
const constexpr size_t alignment = 16  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd128_int32.inl

12.288 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 aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using vect_t = __m128i

Static Public Member Functions

- static const std::string type_string ()
- 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 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)
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)

- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1)
- 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 __m128d &P, const __m128d &INVP, const __m128d &NEGP, const vect_t &POW50REM, const __m128d &MIN, const __m128d &MAX, __m128d &Q, __m128d &T)
- 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)

12.288.1 Member Typedef Documentation

12.288.1.1 scalar_t

```
using scalar_t = uint64_t
```

12.288.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.288.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.288.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.288.1.5 vect_t

```
using vect_t = __m128i  [inherited]
```

12.288.2 Member Function Documentation

12.288.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.288.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.288.2.3 set() [1/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1)  [inline], [static]
```

12.288.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.288.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.288.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.288.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.288.2.8 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.288.2.9 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.288.2.10 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.288.2.11 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b) [inline], [static]
```

12.288.2.12 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b) [inline], [static]
```

12.288.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.288.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.288.2.15 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t x0,
            const vect_t x1) [inline], [static]
```

12.288.2.16 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.288.2.17 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.288.2.18 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.288.2.19 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.288.2.20 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.21 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.22 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.23 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.288.2.24 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.288.2.25 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr], [inherited]
```

12.288.2.26 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr], [inherited]
```

12.288.2.27 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static], [inherited]
```

12.288.2.28 set() [2/2]

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1)  [inline], [static], [inherited]
```

12.288.2.29 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx) [inline], [static], [inherited]
```

12.288.2.30 get()

```
template<int idx>
static INLINE CONST scalar_t get (
            vect_t v) [inline], [static], [inherited]
```

12.288.2.31 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p) [inline], [static], [inherited]
```

12.288.2.32 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p) [inline], [static], [inherited]
```

12.288.2.33 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v) [inline], [static], [inherited]
```

12.288.2.34 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v) [inline], [static], [inherited]
```

12.288.2.35 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v) [inline], [static], [inherited]
```

12.288.2.36 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a) [inline], [static], [inherited]
```

12.288.2.37 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a) [inline], [static], [inherited]
```

12.288.2.38 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a) [inline], [static], [inherited]
```

12.288.2.39 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.40 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.41 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.42 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.43 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.44 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.45 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.46 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.47 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1)  [inline], [static], [inherited]
```

12.288.2.48 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.288.2.49 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.288.2.50 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b) [inline], [static], [inherited]
```

12.288.2.51 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.288.2.52 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b) [inline], [static], [inherited]
```

12.288.2.53 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.288.2.54 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.288.2.55 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.288.2.56 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.288.2.57 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.58 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.59 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.60 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.61 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.288.2.62 mask_high()

```
static INLINE CONST vect_t mask_high ()  [inline], [static], [inherited]
```

12.288.2.63 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y)  [static], [inherited]
```

12.288.2.64 mod()

```
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], [inherited]
```

12.288.2.65 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected], [inherited]
```

12.288.2.66 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.288.2.67 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.288.2.68 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.288.2.69 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.70 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.71 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.2.72 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.288.3 Field Documentation

12.288.3.1 vect_size

```
const constexpr size_t vect_size = 2  [static], [constexpr], [inherited]
```

12.288.3.2 alignment

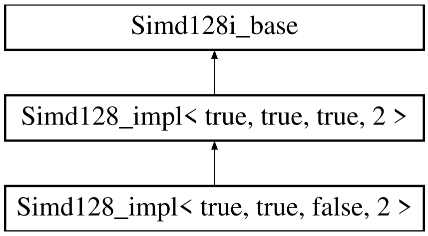
```
const constexpr size_t alignment = 16  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd128_int64.inl

12.289 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
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)

- 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 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

12.289.1 Member Typedef Documentation

12.289.1.1 vect_t

```
using vect_t = __m128i
```

12.289.1.2 scalar_t

```
using scalar_t = int16_t
```

12.289.1.3 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.289.1.4 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.289.1.5 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.289.2 Member Function Documentation

12.289.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.289.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.289.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.289.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.289.2.5 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]
```

12.289.2.6 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.289.2.7 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.289.2.8 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p) [inline], [static]
```

12.289.2.9 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.289.2.10 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.289.2.11 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v) [inline], [static]
```

12.289.2.12 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a) [inline], [static]
```

12.289.2.13 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a) [inline], [static]
```

12.289.2.14 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a) [inline], [static]
```

12.289.2.15 shuffle()

```
template<uint32_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a) [inline], [static]
```

12.289.2.16 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.289.2.17 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.289.2.18 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.19 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.20 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.21 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.22 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.23 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.24 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static]
```

12.289.2.25 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.26 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.27 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.289.2.28 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.29 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.289.2.30 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.31 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.32 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.33 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.34 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.35 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.36 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.289.2.37 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.289.2.38 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.289.2.39 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.289.2.40 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.289.2.41 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.289.2.42 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.289.2.43 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.289.2.44 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.45 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.46 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.47 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.48 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.49 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.50 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.289.2.51 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.289.2.52 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.289.2.53 mod()

```
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)  [inline], [static]
```

12.289.2.54 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.289.2.55 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.289.2.56 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.289.2.57 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.289.2.58 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.289.2.59 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.289.2.60 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.289.3 Field Documentation

12.289.3.1 vect_size

```
const constexpr size_t vect_size = 8  [static], [constexpr]
```

12.289.3.2 alignment

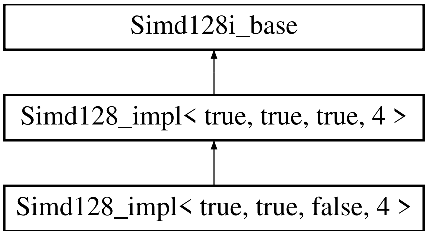
```
const constexpr size_t alignment = 16  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd128_int16.inl

12.290 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
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)

- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3)
- 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 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

12.290.1 Member Typedef Documentation

12.290.1.1 vect_t

```
using vect_t = __m128i
```

12.290.1.2 scalar_t

```
using scalar_t = int32_t
```

12.290.1.3 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.290.1.4 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.290.1.5 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.290.2 Member Function Documentation

12.290.2.1 type_string()

```
static const std::string type_string () [inline], [static]
```

12.290.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p) [inline], [static], [constexpr]
```

12.290.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n) [inline], [static], [constexpr]
```

12.290.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x) [inline], [static]
```

12.290.2.5 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]
```

12.290.2.6 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx) [inline], [static]
```

12.290.2.7 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p) [inline], [static]
```

12.290.2.8 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p) [inline], [static]
```

12.290.2.9 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.290.2.10 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.290.2.11 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v) [inline], [static]
```

12.290.2.12 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a) [inline], [static]
```

12.290.2.13 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a) [inline], [static]
```

12.290.2.14 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a) [inline], [static]
```

12.290.2.15 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a) [inline], [static]
```

12.290.2.16 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.17 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.18 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.19 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.20 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.21 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.22 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.23 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.24 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3) [inline], [static]
```

12.290.2.25 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.26 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.27 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b) [inline], [static]
```

12.290.2.28 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.29 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b) [inline], [static]
```

12.290.2.30 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.31 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.32 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.33 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.34 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.35 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.36 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.37 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.38 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.39 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.40 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.41 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.42 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.43 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.44 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.290.2.45 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.46 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.47 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.48 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.49 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.50 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.290.2.51 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.290.2.52 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.290.2.53 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]
```

12.290.2.54 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.290.2.55 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a) [inline], [static], [inherited]
```

12.290.2.56 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a) [inline], [static], [inherited]
```

12.290.2.57 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.290.2.58 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.290.2.59 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.290.2.60 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.290.3 Field Documentation

12.290.3.1 vect_size

```
const constexpr size_t vect_size = 4 [static], [constexpr]
```

12.290.3.2 alignment

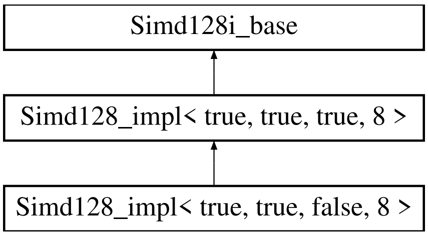
```
const constexpr size_t alignment = 16 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd128_int32.inl

12.291 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
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1)

- 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 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 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 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)

12.291.1 Member Typedef Documentation

12.291.1.1 vect_t

```
using vect_t = __m128i
```

12.291.1.2 scalar_t

```
using scalar_t = int64_t
```

12.291.1.3 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.291.1.4 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.291.1.5 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.291.2 Member Function Documentation

12.291.2.1 type_string()

```
static const std::string type_string ( ) [inline], [static]
```

12.291.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p) [inline], [static], [constexpr]
```

12.291.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n) [inline], [static], [constexpr]
```

12.291.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x) [inline], [static]
```

12.291.2.5 set()

```
static INLINE CONST vect_t set (
            const scalar_t x0,
            const scalar_t x1) [inline], [static]
```

12.291.2.6 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx) [inline], [static]
```

12.291.2.7 get()

```
template<int idx>
static INLINE CONST scalar_t get (
            vect_t v) [inline], [static]
```

12.291.2.8 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.291.2.9 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.291.2.10 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.291.2.11 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.291.2.12 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.291.2.13 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.291.2.14 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.291.2.15 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.291.2.16 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.291.2.17 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.18 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.19 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.20 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.21 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.22 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.23 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.24 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.25 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1) [inline], [static]
```

12.291.2.26 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.27 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.28 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.291.2.29 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.30 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.291.2.31 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t x0,
            const vect_t x1)  [inline], [static]
```

12.291.2.32 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.33 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.34 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.35 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.36 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.37 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.38 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.39 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.40 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.41 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.42 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.43 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.44 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.291.2.45 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.46 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.47 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.48 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.49 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.50 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.51 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.291.2.52 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.291.2.53 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.291.2.54 mask_high()

```
static INLINE CONST vect_t mask_high ()  [inline], [static]
```

12.291.2.55 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y)  [static]
```

12.291.2.56 mod()

```
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]
```

12.291.2.57 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected]
```

12.291.2.58 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.291.2.59 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.291.2.60 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static], [inherited]
```

12.291.2.61 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.291.2.62 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.291.2.63 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.291.2.64 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.291.3 Field Documentation

12.291.3.1 vect_size

```
const constexpr size_t vect_size = 2  [static], [constexpr]
```

12.291.3.2 alignment

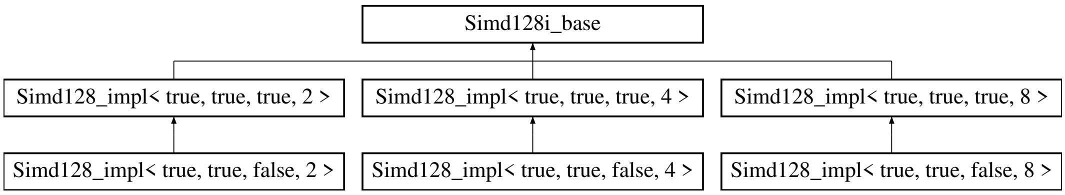
```
const constexpr size_t alignment = 16  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd128_int64.inl

12.292 Simd128i_base Struct Reference

Inheritance diagram for Simd128i_base:



Public Types

- using vect_t = __m128i

Static Public Member Functions

- 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)

12.292.1 Member Typedef Documentation

12.292.1.1 vect_t

```
using vect_t = __m128i
```

12.292.2 Member Function Documentation

12.292.2.1 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static]
```

12.292.2.2 sll128()

```
template<uint8_t s>
static INLINE CONST vect_t sll128 (
            const vect_t a)  [inline], [static]
```

12.292.2.3 srl128()

```
template<uint8_t s>
static INLINE CONST vect_t srl128 (
            const vect_t a)  [inline], [static]
```

12.292.2.4 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.292.2.5 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.292.2.6 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.292.2.7 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

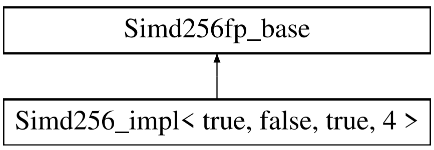
12.293 Simd256_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- simd256.inl

12.294 Simd256_impl< true, false, true, 4 > Struct Reference

Inheritance diagram for Simd256_impl< true, false, true, 4 >:

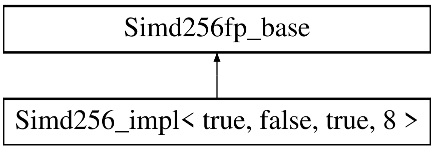


The documentation for this struct was generated from the following file:

- simd256_float.inl

12.295 Simd256_impl< true, false, true, 8 > Struct Reference

Inheritance diagram for Simd256_impl< true, false, true, 8 >:



Data Structures

- union Converter

Public Types

- using vect_t = __m256d
- using scalar_t = double
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3)
- 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

12.295.1 Member Typedef Documentation

12.295.1.1 vect_t

```
using vect_t = __m256d
```

12.295.1.2 scalar_t

```
using scalar_t = double
```

12.295.1.3 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.295.1.4 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.295.1.5 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.295.2 Member Function Documentation

12.295.2.1 type_string()

```
static const std::string type_string ( ) [inline], [static]
```

12.295.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.295.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.295.2.4 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static]
```

12.295.2.5 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.295.2.6 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]
```

12.295.2.7 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.295.2.8 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.295.2.9 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.295.2.10 store()

```
static INLINE void store (
            const scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.295.2.11 storeu()

```
static INLINE void storeu (
            const scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.295.2.12 stream()

```
static INLINE void stream (
            const scalar_t * p,
            const vect_t v) [inline], [static]
```

12.295.2.13 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.14 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.15 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.16 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.17 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.18 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.19 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.20 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.21 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3) [inline], [static]
```

12.295.2.22 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.23 blendv()

```
static INLINE CONST vect_t blendv (
            const vect_t a,
            const vect_t b,
            const vect_t mask) [inline], [static]
```

12.295.2.24 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.25 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b) [inline], [static]
```

12.295.2.26 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.27 subin()

```
static INLINE CONST vect_t subin (
            vect_t & a,
            const vect_t b) [inline], [static]
```

12.295.2.28 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.29 mulin()

```
static INLINE CONST vect_t mulin (
            vect_t & a,
            const vect_t b) [inline], [static]
```

12.295.2.30 div()

```
static INLINE CONST vect_t div (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.31 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.32 fmaddin()

```
static INLINE CONST vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.33 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.34 fnmaddin()

```
static INLINE CONST vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.35 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.36 fmsubin()

```
static INLINE CONST vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.37 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.38 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.295.2.39 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.40 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.41 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.42 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.43 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.44 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.45 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.46 floor()

```
static INLINE CONST vect_t floor (
            const vect_t a)  [inline], [static]
```

12.295.2.47 ceil()

```
static INLINE CONST vect_t ceil (
            const vect_t a)  [inline], [static]
```

12.295.2.48 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.295.2.49 hadd()

```
static INLINE CONST vect_t hadd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.295.2.50 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.295.2.51 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]
```

12.295.3 Field Documentation

12.295.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr]
```

12.295.3.2 alignment

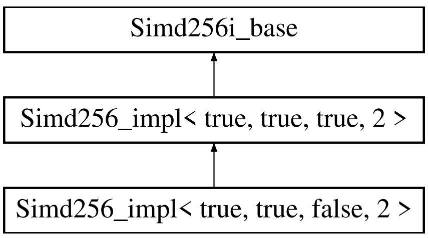
```
const constexpr size_t alignment = 32  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd256_double.inl

12.296 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 aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>

- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using simdHalf = Simd128<scalar_t>
- using vect_t = __m256i
- using half_t = __m128i

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7, vect_t &r8, vect_t &r9, vect_t &r10, vect_t &r11, vect_t &r12, vect_t &r13, vect_t &r14, vect_t &r15)
- template<uint16_t s, typename std::enable_if<(s &0x00ff)==(s > > 8)>::type ∗ = nullptr>
  static INLINE vect_t blend (const vect_t a, const vect_t b)
- template<uint16_t s, typename std::enable_if<(s &0x00ff) !=(s > > 8)>::type ∗ = nullptr>
  static INLINE 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 INLINE CONST vect_t zero ()

Static Public Attributes

- static const constexpr size_t vect_size = 16
- static const constexpr size_t alignment = 32

12.296.1 Member Typedef Documentation

12.296.1.1 scalar_t

```
using scalar_t = uint16_t
```

12.296.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.296.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.296.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.296.1.5 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

12.296.1.6 vect_t

```
using vect_t = __m256i  [inherited]
```

12.296.1.7 half_t

```
using half_t = __m128i  [inherited]
```

12.296.2 Member Function Documentation

12.296.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.296.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.296.2.3 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]
```

12.296.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.296.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.296.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.296.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.296.2.8 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.296.2.9 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v) [inline], [static]
```

12.296.2.10 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a) [inline], [static]
```

12.296.2.11 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b) [inline], [static]
```

12.296.2.12 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b) [inline], [static]
```

12.296.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.296.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.296.2.15 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.296.2.16 mulx()

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b) [inline], [static]
```

12.296.2.17 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.296.2.18 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.296.2.19 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.296.2.20 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.296.2.21 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.296.2.22 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.296.2.23 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a) [inline], [static]
```

12.296.2.24 valid()

```
template<class T>
static constexpr bool valid (
            T * p) [inline], [static], [constexpr], [inherited]
```

12.296.2.25 compliant()

```
template<class T>
static constexpr bool compliant (
            T n) [inline], [static], [constexpr], [inherited]
```

12.296.2.26 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static], [inherited]
```

12.296.2.27 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]
```

12.296.2.28 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.296.2.29 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.296.2.30 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.296.2.31 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.296.2.32 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.296.2.33 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.296.2.34 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a) [inline], [static], [inherited]
```

12.296.2.35 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a) [inline], [static], [inherited]
```

12.296.2.36 shuffle()

```
template<uint64_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a) [inline], [static], [inherited]
```

12.296.2.37 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.38 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.39 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.40 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.41 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.42 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.43 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.44 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.45 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7,
            vect_t & r8,
            vect_t & r9,
            vect_t & r10,
            vect_t & r11,
            vect_t & r12,
            vect_t & r13,
            vect_t & r14,
            vect_t & r15) [inline], [static], [inherited]
```

12.296.2.46 blend() [1/2]

```
template<uint16_t s, typename std::enable_if<(s &0x00ff)==(s > > 8)>::type * = nullptr>
static INLINE vect_t blend (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.47 blend() [2/2]

```
template<uint16_t s, typename std::enable_if<(s &0x00ff) !=(s > > 8)>::type * = nullptr>
static INLINE vect_t blend (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.48 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.49 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.50 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.51 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.52 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.53 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.54 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.55 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.56 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.57 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.58 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.59 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.296.2.60 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.296.2.61 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.296.2.62 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]
```

12.296.2.63 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.296.3 Field Documentation

12.296.3.1 vect_size

```
const constexpr size_t vect_size = 16  [static], [constexpr], [inherited]
```

12.296.3.2 alignment

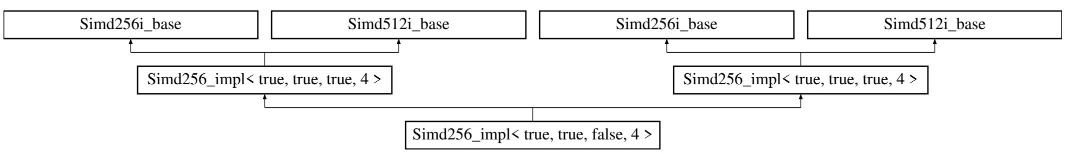
```
const constexpr size_t alignment = 32  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd256_int16.inl

12.297 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 aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>

- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using simdHalf = Simd128<scalar_t>
- using scalar_t = uint32_t
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_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 const std::string type_string ()
- 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 const std::string type_string ()
- 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 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<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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 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 CONST vect_t unpackhi (const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE void unpacklohi (vect_t &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7, vect_t &r8, vect_t &r9, vect_t &r10, vect_t &r11, vect_t &r12, vect_t &r13, vect_t &r14, vect_t &r15)
- template<uint8_t s>
  static INLINE CONST vect_t blend (const vect_t a, const vect_t b)
- template<uint16_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 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

12.297.1 Member Typedef Documentation

12.297.1.1 scalar_t [1/2]

```
using scalar_t = uint32_t
```

12.297.1.2 aligned_allocator [1/2]

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.297.1.3 aligned_vector [1/2]

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.297.1.4 is_same_element [1/2]

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.297.1.5 simdHalf [1/2]

```
using simdHalf = Simd128<scalar_t>
```

12.297.1.6 scalar_t [2/2]

```
using scalar_t = uint32_t
```

12.297.1.7 aligned_allocator [2/2]

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.297.1.8 aligned_vector [2/2]

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.297.1.9 is_same_element [2/2]

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.297.1.10 simdHalf [2/2]

```
using simdHalf = Simd128<scalar_t>
```

12.297.1.11 vect_t [1/2]

```
using vect_t = __m256i  [inherited]
```

12.297.1.12 vect_t [2/2]

```
using vect_t = __m512i  [inherited]
```

12.297.1.13 half_t [1/2]

```
using half_t = __m128i  [inherited]
```

12.297.1.14 half_t [2/2]

```
using half_t = __m256i  [inherited]
```

12.297.2 Member Function Documentation

12.297.2.1 type_string() [1/2]

```
static const std::string type_string ()  [inline], [static]
```

12.297.2.2 set1() [1/3]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.297.2.3 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]
```

12.297.2.4 gather() [1/3]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.297.2.5 load() [1/3]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.297.2.6 loadu() [1/3]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.297.2.7 store() [1/3]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.297.2.8 storeu() [1/3]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.297.2.9 stream() [1/3]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.297.2.10 sra() [1/2]

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.297.2.11 greater() [1/2]

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.297.2.12 lesser() [1/2]

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.297.2.13 greater_eq() [1/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.14 lesser_eq() [1/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.15 mulhi() [1/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.16 mulx() [1/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.297.2.17 fmaddx() [1/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.18 fmaddxin() [1/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.297.2.19 fnmaddx() [1/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.20 fnmaddxin() [1/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.21 fmsubx() [1/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.22 fmsubxin() [1/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.23 hadd_to_scal() [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a) [inline], [static]
```

12.297.2.24 type_string() [2/2]

```
static const std::string type_string () [inline], [static]
```

12.297.2.25 set1() [2/3]

```
static INLINE CONST vect_t set1 (
            const scalar_t x) [inline], [static]
```

12.297.2.26 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]
```

12.297.2.27 gather() [2/3]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx) [inline], [static]
```

12.297.2.28 load() [2/3]

```
static INLINE PURE vect_t load (
            const scalar_t *const p) [inline], [static]
```

12.297.2.29 loadu() [2/3]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p) [inline], [static]
```

12.297.2.30 store() [2/3]

```
static INLINE void store (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.297.2.31 storeu() [2/3]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.297.2.32 stream() [2/3]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v) [inline], [static]
```

12.297.2.33 sra() [2/2]

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a) [inline], [static]
```

12.297.2.34 greater() [2/2]

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b) [inline], [static]
```

12.297.2.35 lesser() [2/2]

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b) [inline], [static]
```

12.297.2.36 greater_eq() [2/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.37 lesser_eq() [2/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.38 mulhi() [2/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.39 mulx() [2/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b) [inline], [static]
```

12.297.2.40 fmaddx() [2/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.41 fmaddxin() [2/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.42 fnmaddx() [2/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.43 fnmaddxin() [2/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.44 fmsubx() [2/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.45 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.297.2.46 hadd_to_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a) [inline], [static]
```

12.297.2.47 valid() [1/2]

```
template<class T>
static constexpr bool valid (
            T * p) [inline], [static], [constexpr], [inherited]
```

12.297.2.48 valid() [2/2]

```
template<class T>
static constexpr bool valid (
            T * p) [inline], [static], [constexpr], [inherited]
```

12.297.2.49 compliant() [1/2]

```
template<class T>
static constexpr bool compliant (
            T n) [inline], [static], [constexpr], [inherited]
```

12.297.2.50 compliant() [2/2]

```
template<class T>
static constexpr bool compliant (
            T n) [inline], [static], [constexpr], [inherited]
```

12.297.2.51 set1() [3/3]

```
static INLINE CONST vect_t set1 (
            const scalar_t x) [inline], [static], [inherited]
```

12.297.2.52 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]
```

12.297.2.53 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]
```

12.297.2.54 gather() [3/3]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.297.2.55 load() [3/3]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.297.2.56 loadu() [3/3]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.297.2.57 store() [3/3]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.297.2.58 storeu() [3/3]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.297.2.59 stream() [3/3]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.297.2.60 sll() [1/2]

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.61 sll() [2/2]

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.62 srl() [1/2]

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.63 srl() [2/2]

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.297.2.64 shuffle_twice() [1/2]

```
template<uint8_t s>
static INLINE CONST vect_t shuffle_twice (
            const vect_t a) [inline], [static], [inherited]
```

12.297.2.65 shuffle_twice() [2/2]

```
template<uint8_t s>
static INLINE CONST vect_t shuffle_twice (
            const vect_t a) [inline], [static], [inherited]
```

12.297.2.66 shuffle() [1/2]

```
template<uint32_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a) [inline], [static], [inherited]
```

12.297.2.67 shuffle() [2/2]

```
template<uint64_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a) [inline], [static], [inherited]
```

12.297.2.68 unpacklo_intrinsic() [1/2]

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.69 unpacklo_intrinsic() [2/2]

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.70 unpackhi_intrinsic() [1/2]

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.71 unpackhi_intrinsic() [2/2]

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.72 unpacklo() [1/2]

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.73 unpacklo() [2/2]

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.74 unpackhi() [1/2]

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.75 unpackhi() [2/2]

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.76 unpacklohi() [1/2]

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.77 unpacklohi() [2/2]

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.78 pack_even() [1/2]

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.79 pack_even() [2/2]

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.80 pack_odd() [1/2]

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.81 pack_odd() [2/2]

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.82 pack() [1/2]

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.83 pack() [2/2]

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.84 transpose() [1/2]

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static], [inherited]
```

12.297.2.85 transpose() [2/2]

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7,
            vect_t & r8,
            vect_t & r9,
            vect_t & r10,
            vect_t & r11,
            vect_t & r12,
            vect_t & r13,
            vect_t & r14,
            vect_t & r15)  [inline], [static], [inherited]
```

12.297.2.86 blend() [1/2]

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.87 blend() [2/2]

```
template<uint16_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.88 add() [1/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.89 add() [2/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.90 addin() [1/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.91 addin() [2/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.92 sub() [1/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.93 sub() [2/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.94 subin() [1/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.95 subin() [2/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.96 mullo() [1/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.97 mullo() [2/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.98 mul() [1/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.99 mul() [2/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.100 fmadd() [1/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.101 fmadd() [2/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.102 fmaddin() [1/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.103 fmaddin() [2/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.104 fnmadd() [1/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.105 fnmadd() [2/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.106 fnmaddin() [1/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.297.2.107 fnmaddin() [2/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.108 fmsub() [1/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.109 fmsub() [2/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.110 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.111 fmsubin() [2/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.112 eq() [1/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.113 eq() [2/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.114 round() [1/2]

```
static INLINE CONST vect_t round (
            const vect_t a) [inline], [static], [inherited]
```

12.297.2.115 round() [2/2]

```
static INLINE CONST vect_t round (
            const vect_t a) [inline], [static], [inherited]
```

12.297.2.116 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]
```

12.297.2.117 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]
```

12.297.2.118 zero() [1/2]

```
static INLINE CONST vect_t zero () [inline], [static], [inherited]
```

12.297.2.119 zero() [2/2]

```
static INLINE CONST vect_t zero () [inline], [static], [inherited]
```

12.297.2.120 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.121 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.122 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.2.123 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.297.3 Field Documentation

12.297.3.1 vect_size

```
static const constexpr size_t vect_size = 8 [static], [constexpr], [inherited]
```

12.297.3.2 alignment

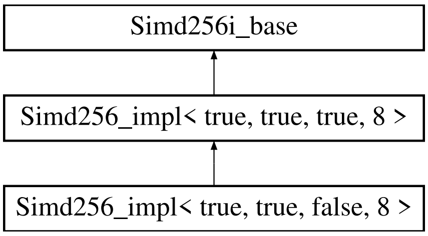
```
static const constexpr size_t alignment = 32 [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

12.298 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 aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using simdHalf = Simd128<scalar_t>
- using vect_t = __m256i
- using half_t = __m128i

Static Public Member Functions

- static const std::string type_string ()
- 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 mulhi (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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3)
- 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 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)

12.298.1 Member Typedef Documentation

12.298.1.1 scalar_t

```
using scalar_t = uint64_t
```

12.298.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.298.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.298.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.298.1.5 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

12.298.1.6 vect_t

```
using vect_t = __m256i  [inherited]
```

12.298.1.7 half_t

```
using half_t = __m128i  [inherited]
```

12.298.2 Member Function Documentation

12.298.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.298.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.298.2.3 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]
```

12.298.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx) [inline], [static]
```

12.298.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p) [inline], [static]
```

12.298.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p) [inline], [static]
```

12.298.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.298.2.8 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.298.2.9 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v) [inline], [static]
```

12.298.2.10 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a) [inline], [static]
```

12.298.2.11 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b) [inline], [static]
```

12.298.2.12 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b) [inline], [static]
```

12.298.2.13 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.298.2.14 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.15 mullo()

```
static INLINE CONST vect_t mullo (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.298.2.16 mulhi()

```
static INLINE CONST vect_t mulhi (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.298.2.17 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.18 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.19 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.20 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.21 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.22 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.298.2.23 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.298.2.24 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a) [inline], [static]
```

12.298.2.25 valid()

```
template<class T>
static constexpr bool valid (
            T * p) [inline], [static], [constexpr], [inherited]
```

12.298.2.26 compliant()

```
template<class T>
static constexpr bool compliant (
            T n) [inline], [static], [constexpr], [inherited]
```

12.298.2.27 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x) [inline], [static], [inherited]
```

12.298.2.28 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]
```

12.298.2.29 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx) [inline], [static], [inherited]
```

12.298.2.30 get()

```
template<int idx>
static INLINE CONST scalar_t get (
            vect_t v) [inline], [static], [inherited]
```

12.298.2.31 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p) [inline], [static], [inherited]
```

12.298.2.32 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p) [inline], [static], [inherited]
```

12.298.2.33 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v) [inline], [static], [inherited]
```

12.298.2.34 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v) [inline], [static], [inherited]
```

12.298.2.35 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v) [inline], [static], [inherited]
```

12.298.2.36 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a) [inline], [static], [inherited]
```

12.298.2.37 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a) [inline], [static], [inherited]
```

12.298.2.38 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a) [inline], [static], [inherited]
```

12.298.2.39 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.40 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.41 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.42 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.43 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.44 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.45 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.46 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.47 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3)  [inline], [static], [inherited]
```

12.298.2.48 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.49 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.50 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.51 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.298.2.52 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.53 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.54 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.55 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.56 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.57 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.58 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.59 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.60 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.298.2.61 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static], [inherited]
```

12.298.2.62 mask_high()

```
static INLINE CONST vect_t mask_high ()  [inline], [static], [inherited]
```

12.298.2.63 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y)  [static], [inherited]
```

12.298.2.64 mod()

```
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], [inherited]
```

12.298.2.65 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected], [inherited]
```

12.298.2.66 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.298.3 Field Documentation

12.298.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr], [inherited]
```

12.298.3.2 alignment

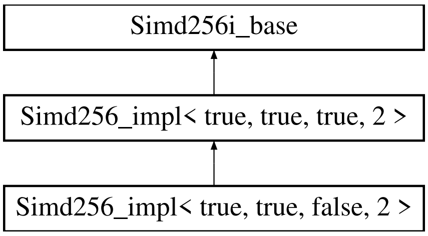
```
const constexpr size_t alignment = 32  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd256_int64.inl

12.299 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>
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)

- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7, vect_t &r8, vect_t &r9, vect_t &r10, vect_t &r11, vect_t &r12, vect_t &r13, vect_t &r14, vect_t &r15)
- template<uint16_t s, typename std::enable_if<(s &0x00ff)==(s >> 8)>::type * = nullptr>
  static INLINE vect_t blend (const vect_t a, const vect_t b)
- template<uint16_t s, typename std::enable_if<(s &0x00ff) !=(s >> 8)>::type * = nullptr>
  static INLINE 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 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 INLINE CONST vect_t zero ()

Static Public Attributes

- static const constexpr size_t vect_size = 16
- static const constexpr size_t alignment = 32

12.299.1 Member Typedef Documentation

12.299.1.1 vect_t

```
using vect_t = __m256i
```

12.299.1.2 half_t

```
using half_t = __m128i
```

12.299.1.3 scalar_t

```
using scalar_t = int16_t
```

12.299.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

12.299.1.5 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.299.1.6 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.299.1.7 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.299.2 Member Function Documentation

12.299.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.299.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.299.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.299.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.299.2.5 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]
```

12.299.2.6 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx) [inline], [static]
```

12.299.2.7 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p) [inline], [static]
```

12.299.2.8 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p) [inline], [static]
```

12.299.2.9 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.299.2.10 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v) [inline], [static]
```

12.299.2.11 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v) [inline], [static]
```

12.299.2.12 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a) [inline], [static]
```

12.299.2.13 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a) [inline], [static]
```

12.299.2.14 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a) [inline], [static]
```

12.299.2.15 shuffle()

```
template<uint64_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a) [inline], [static]
```

12.299.2.16 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.17 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.18 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.19 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.20 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.21 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.22 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.23 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.24 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
```

```
            vect_t & r5,
            vect_t & r6,
            vect_t & r7,
            vect_t & r8,
            vect_t & r9,
            vect_t & r10,
            vect_t & r11,
            vect_t & r12,
            vect_t & r13,
            vect_t & r14,
            vect_t & r15)  [inline], [static]
```

12.299.2.25 blend() [1/2]

```
template<uint16_t s, typename std::enable_if<(s &0x00ff)==(s > > 8)>::type * = nullptr>
static INLINE vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.26 blend() [2/2]

```
template<uint16_t s, typename std::enable_if<(s &0x00ff) !=(s > > 8)>::type * = nullptr>
static INLINE vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.27 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.28 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.299.2.29 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.30 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.299.2.31 mullo()

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.32 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.299.2.33 mulhi()

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.299.2.34 mulx()

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b) [inline], [static]
```

12.299.2.35 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.299.2.36 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.299.2.37 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.299.2.38 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.299.2.39 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.299.2.40 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.299.2.41 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.42 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.43 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.44 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.45 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.46 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.47 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.48 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.49 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.50 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.51 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.299.2.52 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.299.2.53 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.299.2.54 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]
```

12.299.2.55 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.299.3 Field Documentation

12.299.3.1 vect_size

```
const constexpr size_t vect_size = 16  [static], [constexpr]
```

12.299.3.2 alignment

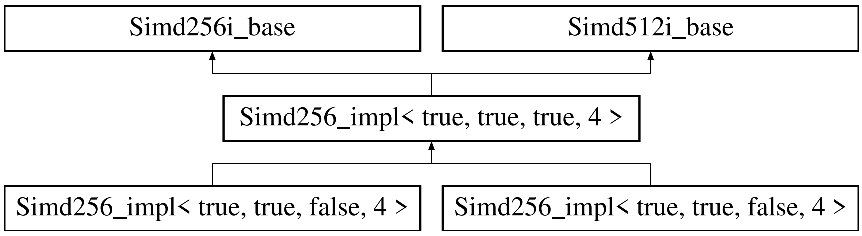
```
const constexpr size_t alignment = 32  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd256_int16.inl

12.300 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 aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using vect_t = __m512i
- using half_t = __m256i
- using scalar_t = int32_t
- using simdHalf = Simd256<scalar_t>
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- 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 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 ()
- 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 unpacklo_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7, vect_t &r8, vect_t &r9, vect_t &r10, vect_t &r11, vect_t &r12, vect_t &r13, vect_t &r14, vect_t &r15)
- template<uint16_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 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 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

12.300.1 Member Typedef Documentation

12.300.1.1 vect_t [1/2]

```
using vect_t = __m256i
```

12.300.1.2 half_t [1/2]

```
using half_t = __m128i
```

12.300.1.3 scalar_t [1/2]

```
using scalar_t = int32_t
```

12.300.1.4 simdHalf [1/2]

```
using simdHalf = Simd128<scalar_t>
```

12.300.1.5 aligned_allocator [1/2]

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.300.1.6 aligned_vector [1/2]

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.300.1.7 is_same_element [1/2]

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.300.1.8 vect_t [2/2]

```
using vect_t = __m512i
```

12.300.1.9 half_t [2/2]

```
using half_t = __m256i
```

12.300.1.10 scalar_t [2/2]

```
using scalar_t = int32_t
```

12.300.1.11 simdHalf [2/2]

```
using simdHalf = Simd256<scalar_t>
```

12.300.1.12 aligned_allocator [2/2]

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.300.1.13 aligned_vector [2/2]

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.300.1.14 is_same_element [2/2]

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.300.2 Member Function Documentation

12.300.2.1 type_string() [1/2]

```
static const std::string type_string ()  [inline], [static]
```

12.300.2.2 valid() [1/2]

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.300.2.3 compliant() [1/2]

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.300.2.4 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.300.2.5 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]
```

12.300.2.6 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.300.2.7 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.300.2.8 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.300.2.9 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.300.2.10 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.300.2.11 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.300.2.12 sll() [1/2]

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.300.2.13 srl() [1/2]

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.300.2.14 sra() [1/2]

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.300.2.15 shuffle_twice() [1/2]

```
template<uint8_t s>
static INLINE CONST vect_t shuffle_twice (
            const vect_t a)  [inline], [static]
```

12.300.2.16 shuffle() [1/2]

```
template<uint32_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.300.2.17 unpacklo_intrinsic() [1/2]

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.18 unpackhi_intrinsic() [1/2]

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.19 unpacklo() [1/2]

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.20 unpackhi() [1/2]

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.21 unpacklohi() [1/2]

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.22 pack_even() [1/2]

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.23 pack_odd() [1/2]

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.24 pack() [1/2]

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.25 transpose() [1/2]

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static]
```

12.300.2.26 blend() [1/2]

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.27 add() [1/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.28 addin() [1/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.300.2.29 sub() [1/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.30 subin() [1/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.300.2.31 mullo() [1/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.32 mul() [1/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.33 mulhi() [1/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.34 mulx() [1/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.300.2.35 fmadd() [1/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.36 fmaddin() [1/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.37 fmaddx() [1/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.38 fmaddxin() [1/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.39 fnmadd() [1/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.40 fnmaddin() [1/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.41 fnmaddx() [1/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.42 fnmaddxin() [1/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.43 fmsub() [1/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.44 fmsubin() [1/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.45 fmsubx() [1/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.46 fmsubxin() [1/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.47 eq() [1/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.48 greater() [1/2]

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.49 lesser() [1/2]

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.50 greater_eq() [1/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.51 lesser_eq() [1/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.52 hadd_to_scal() [1/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.300.2.53 round() [1/2]

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.300.2.54 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]
```

12.300.2.55 type_string() [2/2]

```
static const std::string type_string ()  [inline], [static]
```

12.300.2.56 valid() [2/2]

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.300.2.57 compliant() [2/2]

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.300.2.58 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.300.2.59 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]
```

12.300.2.60 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.300.2.61 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.300.2.62 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.300.2.63 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.300.2.64 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.300.2.65 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.300.2.66 sll() [2/2]

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.300.2.67 srl() [2/2]

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.300.2.68 sra() [2/2]

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.300.2.69 shuffle_twice() [2/2]

```
template<uint8_t s>
static INLINE CONST vect_t shuffle_twice (
            const vect_t a)  [inline], [static]
```

12.300.2.70 shuffle() [2/2]

```
template<uint64_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.300.2.71 unpacklo_intrinsic() [2/2]

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.300.2.72 unpackhi_intrinsic() [2/2]

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.300.2.73 unpacklo() [2/2]

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.300.2.74 unpackhi() [2/2]

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.300.2.75 unpacklohi() [2/2]

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.300.2.76 pack_even() [2/2]

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.300.2.77 pack_odd() [2/2]

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.300.2.78 pack() [2/2]

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.300.2.79 transpose() [2/2]

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
```

```
            vect_t & r5,
            vect_t & r6,
            vect_t & r7,
            vect_t & r8,
            vect_t & r9,
            vect_t & r10,
            vect_t & r11,
            vect_t & r12,
            vect_t & r13,
            vect_t & r14,
            vect_t & r15)  [inline], [static]
```

12.300.2.80 blend() [2/2]

```
template<uint16_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.81 add() [2/2]

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.82 addin() [2/2]

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.300.2.83 sub() [2/2]

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.84 subin() [2/2]

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.300.2.85 mullo() [2/2]

```
static INLINE CONST vect_t mullo (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.86 mul() [2/2]

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.87 mulhi() [2/2]

```
static INLINE CONST vect_t mulhi (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.88 mulx() [2/2]

```
static INLINE CONST vect_t mulx (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.300.2.89 fmadd() [2/2]

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.90 fmaddin() [2/2]

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.91 fmaddx() [2/2]

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.92 fmaddxin() [2/2]

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.93 fnmadd() [2/2]

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.94 fnmaddin() [2/2]

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.95 fnmaddx() [2/2]

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.96 fnmaddxin() [2/2]

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.97 fmsub() [2/2]

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.98 fmsubin() [2/2]

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.99 fmsubx() [2/2]

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.100 fmsubxin() [2/2]

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.101 eq() [2/2]

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.102 greater() [2/2]

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.103 lesser() [2/2]

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.104 greater_eq() [2/2]

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.105 lesser_eq() [2/2]

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.300.2.106 hadd_to_scal() [2/2]

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.300.2.107 round() [2/2]

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.300.2.108 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]
```

12.300.2.109 zero() [1/2]

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.300.2.110 zero() [2/2]

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.300.2.111 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.300.2.112 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.300.2.113 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.300.2.114 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.300.3 Field Documentation

12.300.3.1 vect_size

```
static const constexpr size_t vect_size = 8  [static], [constexpr]
```

12.300.3.2 alignment

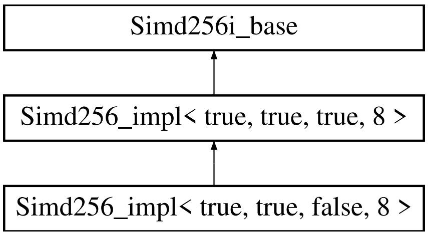
`static const constexpr size_t alignment = 32 [static], [constexpr]`

The documentation for this struct was generated from the following files:

- simd256_int32.inl
- simd512_int32.inl

12.301 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>
- using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3)
- 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 mulhi (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 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 __m256d &P, const __m256d &INVP, const __m256d &NEGP, const vect_t &POW50REM, const __m256d &MIN, const __m256d &MAX, __m256d &Q, __m256d &T)
- 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)

12.301.1 Member Typedef Documentation

12.301.1.1 vect_t

```
using vect_t = __m256i
```

12.301.1.2 half_t

```
using half_t = __m128i
```

12.301.1.3 scalar_t

```
using scalar_t = int64_t
```

12.301.1.4 simdHalf

```
using simdHalf = Simd128<scalar_t>
```

12.301.1.5 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.301.1.6 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.301.1.7 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.301.2 Member Function Documentation

12.301.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.301.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.301.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.301.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.301.2.5 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]
```

12.301.2.6 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.301.2.7 get()

```
template<int idx>
static INLINE CONST scalar_t get (
            vect_t v)  [inline], [static]
```

12.301.2.8 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.301.2.9 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.301.2.10 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.301.2.11 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.301.2.12 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.301.2.13 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.301.2.14 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.301.2.15 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a) [inline], [static]
```

12.301.2.16 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a) [inline], [static]
```

12.301.2.17 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.301.2.18 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.301.2.19 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.301.2.20 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.301.2.21 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.301.2.22 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.301.2.23 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.301.2.24 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.25 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3)  [inline], [static]
```

12.301.2.26 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.27 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.28 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.301.2.29 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.30 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.301.2.31 mullo()

```
static INLINE CONST vect_t mullo (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.301.2.32 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.33 mulhi()

```
static INLINE CONST vect_t mulhi (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.301.2.34 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.35 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.36 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.37 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.38 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.39 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.40 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.41 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.42 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.43 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.44 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.45 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.46 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.47 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.48 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.49 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.50 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.51 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.301.2.52 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.301.2.53 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.301.2.54 mask_high()

```
static INLINE CONST vect_t mask_high ()  [inline], [static]
```

12.301.2.55 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y)  [static]
```

12.301.2.56 mod()

```
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]
```

12.301.2.57 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected]
```

12.301.2.58 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.301.3 Field Documentation

12.301.3.1 vect_size

```
const constexpr size_t vect_size = 4  [static], [constexpr]
```

12.301.3.2 alignment

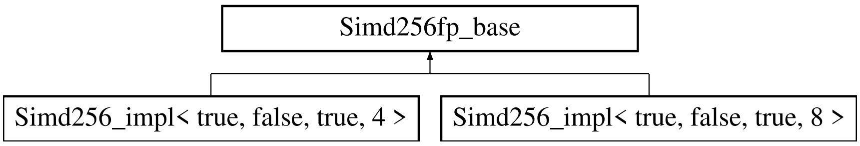
```
const constexpr size_t alignment = 32  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd256_int64.inl

12.302 Simd256fp_base Struct Reference

Inheritance diagram for Simd256fp_base:

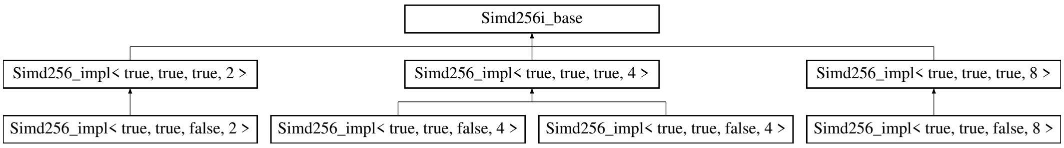


The documentation for this struct was generated from the following file:

- simd256.inl

12.303 Simd256i_base Struct Reference

Inheritance diagram for Simd256i_base:



Public Types

- using vect_t = __m256i

Static Public Member Functions

- static INLINE CONST vect_t zero ()

12.303.1 Member Typedef Documentation

12.303.1.1 vect_t

```
using vect_t = __m256i
```

12.303.2 Member Function Documentation

12.303.2.1 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static]
```

The documentation for this struct was generated from the following file:

- simd256.inl

12.304 Simd512_impl< ArithType, Int, Signed, Size > Struct Template Reference

The documentation for this struct was generated from the following file:

- simd512.inl

12.305 Simd512_impl< true, false, true, 4 > Struct Reference

The documentation for this struct was generated from the following file:

- simd512_float.inl

12.306 Simd512_impl< true, false, true, 8 > Struct Reference

Public Types

- using vect_t = __m512d
- using scalar_t = double
- using aligned_allocator = AlignedAllocator< scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector< scalar_t, aligned_allocator >
- template< class Field >
  using is_same_element = std::is_same< typename Field::Element, scalar_t >

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- 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 Public Attributes

- static const constexpr size_t vect_size = 8
- static const constexpr size_t alignment = 64

12.306.1 Member Typedef Documentation

12.306.1.1 vect_t

```
using vect_t = __m512d
```

12.306.1.2 scalar_t

```
using scalar_t = double
```

12.306.1.3 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.306.1.4 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.306.1.5 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.306.2 Member Function Documentation

12.306.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.306.2.2 valid()

```
template<class T>
static constexpr bool valid (
             T * p)  [inline], [static], [constexpr]
```

12.306.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
             T n)  [inline], [static], [constexpr]
```

12.306.2.4 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static]
```

12.306.2.5 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.306.2.6 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]
```

12.306.2.7 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.306.2.8 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.306.2.9 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.306.2.10 store()

```
static INLINE void store (
            const scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.306.2.11 storeu()

```
static INLINE void storeu (
            const scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.306.2.12 stream()

```
static INLINE void stream (
            const scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.306.2.13 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static]
```

12.306.2.14 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.15 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.16 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.17 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.18 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.19 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.20 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.21 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.22 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
```

```
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static]
```

12.306.2.23 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.24 blendv()

```
static INLINE CONST vect_t blendv (
            const vect_t a,
            const vect_t b,
            const vect_t mask)  [inline], [static]
```

12.306.2.25 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.26 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.306.2.27 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.28 subin()

```
static INLINE CONST vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.306.2.29 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.30 mulin()

```
static INLINE CONST vect_t mulin (
            vect_t & a,
            const vect_t b)  [inline], [static]
```

12.306.2.31 div()

```
static INLINE CONST vect_t div (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.306.2.32 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.33 fmaddin()

```
static INLINE CONST vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.34 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.35 fnmaddin()

```
static INLINE CONST vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.36 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.37 fmsubin()

```
static INLINE CONST vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.38 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.39 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.40 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.41 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.42 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.43 floor()

```
static INLINE CONST vect_t floor (
            const vect_t a) [inline], [static]
```

12.306.2.44 ceil()

```
static INLINE CONST vect_t ceil (
            const vect_t a) [inline], [static]
```

12.306.2.45 round()

```
static INLINE CONST vect_t round (
            const vect_t a) [inline], [static]
```

12.306.2.46 hadd()

```
static INLINE CONST vect_t hadd (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.306.2.47 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a) [inline], [static]
```

12.306.3 Field Documentation

12.306.3.1 vect_size

```
const constexpr size_t vect_size = 8 [static], [constexpr]
```

12.306.3.2 alignment

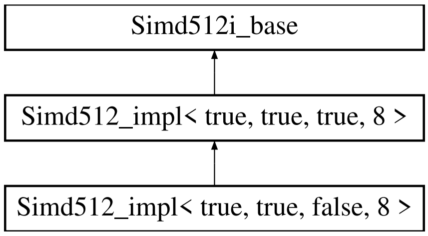
```
const constexpr size_t alignment = 64 [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd512_double.inl

12.307 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 aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector<scalar_t, aligned_allocator>
- template<class Field>
  using is_same_element = std::is_same<typename Field::Element, scalar_t>
- using simdHalf = Simd256<scalar_t>
- using vect_t = __m512i
- using half_t = __m256i

Static Public Member Functions

- static const std::string type_string ()
- 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 mulhi (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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- 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 __m512d &P, const __m512d &INVP, const __m512d &NEGP, const vect_t &POW50REM, const __m512d &MIN, const __m512d &MAX, __m512d &Q, __m512d &T)
- 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)

12.307.1 Member Typedef Documentation

12.307.1.1 scalar_t

```
using scalar_t = uint64_t
```

12.307.1.2 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.307.1.3 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.307.1.4 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.307.1.5 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

12.307.1.6 vect_t

```
using vect_t = __m512i  [inherited]
```

12.307.1.7 half_t

```
using half_t = __m256i  [inherited]
```

12.307.2 Member Function Documentation

12.307.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.307.2.2 set1() [1/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.307.2.3 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]
```

12.307.2.4 gather() [1/2]

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.307.2.5 load() [1/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.307.2.6 loadu() [1/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.307.2.7 store() [1/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.307.2.8 maskstore() [1/2]

```
template<uint8_t k>
static INLINE void maskstore (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.307.2.9 storeu() [1/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.307.2.10 stream() [1/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.307.2.11 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.307.2.12 greater()

```
static INLINE CONST vect_t greater (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.307.2.13 lesser()

```
static INLINE CONST vect_t lesser (
            vect_t a,
            vect_t b)  [inline], [static]
```

12.307.2.14 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.307.2.15 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.307.2.16 mullo()

```
static INLINE CONST vect_t mullo (
            vect_t a,
            vect_t b) [inline], [static]
```

12.307.2.17 mulhi()

```
static INLINE CONST vect_t mulhi (
            vect_t a,
            vect_t b) [inline], [static]
```

12.307.2.18 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.307.2.19 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.307.2.20 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.307.2.21 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.307.2.22 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.307.2.23 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.307.2.24 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.307.2.25 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a) [inline], [static]
```

12.307.2.26 valid()

```
template<class T>
static constexpr bool valid (
            T * p) [inline], [static], [constexpr], [inherited]
```

12.307.2.27 compliant()

```
template<class T>
static constexpr bool compliant (
            T n) [inline], [static], [constexpr], [inherited]
```

12.307.2.28 set1() [2/2]

```
static INLINE CONST vect_t set1 (
            const scalar_t x) [inline], [static], [inherited]
```

12.307.2.29 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]
```

12.307.2.30 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]
```

12.307.2.31 gather() [2/2]

```
template<class T>
static INLINE PURE vect_t gather (
```

```
            const scalar_t *const p,
            const T *const idx)  [inline], [static], [inherited]
```

12.307.2.32 load() [2/2]

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.307.2.33 loadu() [2/2]

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static], [inherited]
```

12.307.2.34 store() [2/2]

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.307.2.35 maskstore() [2/2]

```
template<uint8_t k>
static INLINE void maskstore (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.307.2.36 storeu() [2/2]

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static], [inherited]
```

12.307.2.37 stream() [2/2]

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static], [inherited]
```

12.307.2.38 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static], [inherited]
```

12.307.2.39 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static], [inherited]
```

12.307.2.40 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a)  [inline], [static], [inherited]
```

12.307.2.41 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.42 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.43 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.44 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.45 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.46 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.47 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.48 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.49 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
```

```
            vect_t & r5,
            vect_t & r6,
            vect_t & r7)  [inline], [static], [inherited]
```

12.307.2.50 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.51 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.52 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.53 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.54 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.55 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.56 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.57 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.58 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
```

```
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.307.2.59 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.307.2.60 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.307.2.61 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.307.2.62 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b) [inline], [static], [inherited]
```

12.307.2.63 round()

```
static INLINE CONST vect_t round (
            const vect_t a) [inline], [static], [inherited]
```

12.307.2.64 mask_high()

```
static INLINE CONST vect_t mask_high () [inline], [static], [inherited]
```

12.307.2.65 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y) [static], [inherited]
```

12.307.2.66 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], [inherited]
```

12.307.2.67 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected], [inherited]
```

12.307.2.68 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.307.2.69 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.70 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.71 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.2.72 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.307.3 Field Documentation

12.307.3.1 vect_size

```
const constexpr size_t vect_size = 8  [static], [constexpr], [inherited]
```

12.307.3.2 alignment

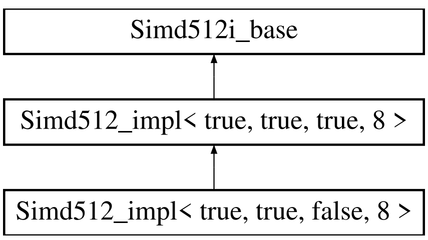
```
const constexpr size_t alignment = 64  [static], [constexpr], [inherited]
```

The documentation for this struct was generated from the following file:

- simd512_int64.inl

12.308 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 >
- using aligned_allocator = AlignedAllocator< scalar_t, Alignment(alignment)>
- using aligned_vector = std::vector< scalar_t, aligned_allocator >
- template<class Field >
  using is_same_element = std::is_same< typename Field::Element, scalar_t >

Static Public Member Functions

- static const std::string type_string ()
- 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_intrinsic (const vect_t a, const vect_t b)
- static INLINE CONST vect_t unpackhi_intrinsic (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 &lo, vect_t &hi, const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_even (const vect_t a, const vect_t b)
- static INLINE CONST vect_t pack_odd (const vect_t a, const vect_t b)
- static INLINE void pack (vect_t &even, vect_t &odd, const vect_t a, const vect_t b)
- static INLINE void transpose (vect_t &r0, vect_t &r1, vect_t &r2, vect_t &r3, vect_t &r4, vect_t &r5, vect_t &r6, vect_t &r7)
- 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 mulhi (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 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 __m512d &P, const __m512d &INVP, const __m512d &NEGP, const vect_t &POW50REM, const __m512d &MIN, const __m512d &MAX, __m512d &Q, __m512d &T)
- 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)

12.308.1 Member Typedef Documentation

12.308.1.1 vect_t

```
using vect_t = __m512i
```

12.308.1.2 half_t

```
using half_t = __m256i
```

12.308.1.3 scalar_t

```
using scalar_t = int64_t
```

12.308.1.4 simdHalf

```
using simdHalf = Simd256<scalar_t>
```

12.308.1.5 aligned_allocator

```
using aligned_allocator = AlignedAllocator<scalar_t, Alignment(alignment)>
```

12.308.1.6 aligned_vector

```
using aligned_vector = std::vector<scalar_t, aligned_allocator>
```

12.308.1.7 is_same_element

```
template<class Field>
using is_same_element = std::is_same<typename Field::Element, scalar_t>
```

12.308.2 Member Function Documentation

12.308.2.1 type_string()

```
static const std::string type_string ()  [inline], [static]
```

12.308.2.2 valid()

```
template<class T>
static constexpr bool valid (
            T * p)  [inline], [static], [constexpr]
```

12.308.2.3 compliant()

```
template<class T>
static constexpr bool compliant (
            T n)  [inline], [static], [constexpr]
```

12.308.2.4 set1()

```
static INLINE CONST vect_t set1 (
            const scalar_t x)  [inline], [static]
```

12.308.2.5 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]
```

12.308.2.6 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]
```

12.308.2.7 gather()

```
template<class T>
static INLINE PURE vect_t gather (
            const scalar_t *const p,
            const T *const idx)  [inline], [static]
```

12.308.2.8 load()

```
static INLINE PURE vect_t load (
            const scalar_t *const p)  [inline], [static]
```

12.308.2.9 loadu()

```
static INLINE PURE vect_t loadu (
            const scalar_t *const p)  [inline], [static]
```

12.308.2.10 store()

```
static INLINE void store (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.308.2.11 maskstore()

```
template<uint8_t k>
static INLINE void maskstore (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.308.2.12 storeu()

```
static INLINE void storeu (
            scalar_t * p,
            vect_t v)  [inline], [static]
```

12.308.2.13 stream()

```
static INLINE void stream (
            scalar_t * p,
            const vect_t v)  [inline], [static]
```

12.308.2.14 sll()

```
template<int s>
static INLINE CONST vect_t sll (
            const vect_t a)  [inline], [static]
```

12.308.2.15 srl()

```
template<int s>
static INLINE CONST vect_t srl (
            const vect_t a)  [inline], [static]
```

12.308.2.16 sra()

```
template<int s>
static INLINE CONST vect_t sra (
            const vect_t a)  [inline], [static]
```

12.308.2.17 shuffle()

```
template<uint8_t s>
static INLINE CONST vect_t shuffle (
            const vect_t a) [inline], [static]
```

12.308.2.18 unpacklo_intrinsic()

```
static INLINE CONST vect_t unpacklo_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.19 unpackhi_intrinsic()

```
static INLINE CONST vect_t unpackhi_intrinsic (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.20 unpacklo()

```
static INLINE CONST vect_t unpacklo (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.21 unpackhi()

```
static INLINE CONST vect_t unpackhi (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.22 unpacklohi()

```
static INLINE void unpacklohi (
            vect_t & lo,
            vect_t & hi,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.23 pack_even()

```
static INLINE CONST vect_t pack_even (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.24 pack_odd()

```
static INLINE CONST vect_t pack_odd (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.25 pack()

```
static INLINE void pack (
            vect_t & even,
            vect_t & odd,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.26 transpose()

```
static INLINE void transpose (
            vect_t & r0,
            vect_t & r1,
            vect_t & r2,
            vect_t & r3,
            vect_t & r4,
            vect_t & r5,
            vect_t & r6,
            vect_t & r7) [inline], [static]
```

12.308.2.27 blend()

```
template<uint8_t s>
static INLINE CONST vect_t blend (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.28 add()

```
static INLINE CONST vect_t add (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.29 addin()

```
static INLINE vect_t addin (
            vect_t & a,
            const vect_t b) [inline], [static]
```

12.308.2.30 sub()

```
static INLINE CONST vect_t sub (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.31 subin()

```
static INLINE vect_t subin (
            vect_t & a,
            const vect_t b) [inline], [static]
```

12.308.2.32 mullo()

```
static INLINE CONST vect_t mullo (
            vect_t a,
            vect_t b) [inline], [static]
```

12.308.2.33 mul()

```
static INLINE CONST vect_t mul (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.34 mulhi()

```
static INLINE CONST vect_t mulhi (
            vect_t a,
            vect_t b) [inline], [static]
```

12.308.2.35 mulx()

```
static INLINE CONST vect_t mulx (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.36 fmadd()

```
static INLINE CONST vect_t fmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.37 fmaddin()

```
static INLINE vect_t fmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.38 fmaddx()

```
static INLINE CONST vect_t fmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.39 fmaddxin()

```
static INLINE vect_t fmaddxin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.40 fnmadd()

```
static INLINE CONST vect_t fnmadd (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.41 fnmaddin()

```
static INLINE vect_t fnmaddin (
            vect_t & c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.42 fnmaddx()

```
static INLINE CONST vect_t fnmaddx (
            const vect_t c,
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.308.2.43 fnmaddxin()

```
static INLINE vect_t fnmaddxin (
            vect_t & c,
```

```
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.44 fmsub()

```
static INLINE CONST vect_t fmsub (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.45 fmsubin()

```
static INLINE vect_t fmsubin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.46 fmsubx()

```
static INLINE CONST vect_t fmsubx (
            const vect_t c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.47 fmsubxin()

```
static INLINE vect_t fmsubxin (
            vect_t & c,
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.48 eq()

```
static INLINE CONST vect_t eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.49 greater()

```
static INLINE CONST vect_t greater (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.50 lesser()

```
static INLINE CONST vect_t lesser (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.51 greater_eq()

```
static INLINE CONST vect_t greater_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.52 lesser_eq()

```
static INLINE CONST vect_t lesser_eq (
            const vect_t a,
            const vect_t b) [inline], [static]
```

12.308.2.53 hadd_to_scal()

```
static INLINE CONST scalar_t hadd_to_scal (
            const vect_t a)  [inline], [static]
```

12.308.2.54 round()

```
static INLINE CONST vect_t round (
            const vect_t a)  [inline], [static]
```

12.308.2.55 mask_high()

```
static INLINE CONST vect_t mask_high ()  [inline], [static]
```

12.308.2.56 mulhi_fast()

```
INLINE CONST vect_t mulhi_fast (
            vect_t x,
            vect_t y)  [static]
```

12.308.2.57 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]
```

12.308.2.58 signbits()

```
static INLINE CONST vect_t signbits (
            const vect_t x)  [inline], [static], [protected]
```

12.308.2.59 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static], [inherited]
```

12.308.2.60 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.308.2.61 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.308.2.62 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.308.2.63 vandnot()

```
static INLINE CONST vect_t vandnot (
            const vect_t a,
            const vect_t b)  [inline], [static], [inherited]
```

12.308.3 Field Documentation

12.308.3.1 vect_size

```
const constexpr size_t vect_size = 8  [static], [constexpr]
```

12.308.3.2 alignment

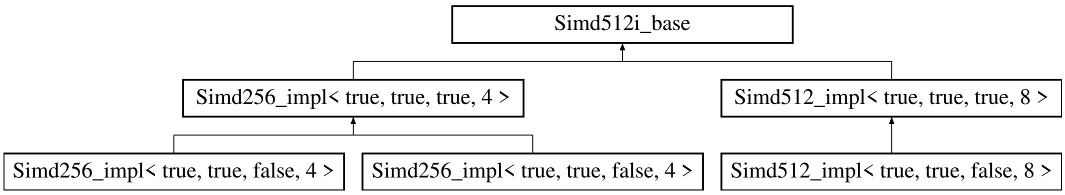
```
const constexpr size_t alignment = 64  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- simd512_int64.inl

12.309 Simd512i_base Struct Reference

Inheritance diagram for Simd512i_base:



Public Types

- using vect_t = __m512i

Static Public Member Functions

- 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)

12.309.1 Member Typedef Documentation

12.309.1.1 vect_t

```
using vect_t = __m512i
```

12.309.2 Member Function Documentation

12.309.2.1 zero()

```
static INLINE CONST vect_t zero ()  [inline], [static]
```

12.309.2.2 vor()

```
static INLINE CONST vect_t vor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.309.2.3 vxor()

```
static INLINE CONST vect_t vxor (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.309.2.4 vand()

```
static INLINE CONST vect_t vand (
            const vect_t a,
            const vect_t b)  [inline], [static]
```

12.309.2.5 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

12.310 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

12.311 SimdChooser< T, false, b > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using value = NoSimd<T>

12.311.1 Member Typedef Documentation

12.311.1.1 value

```
template<class T, bool b>
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

12.312 SimdChooser< T, true, false > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using value = NoSimd<T>

12.312.1 Member Typedef Documentation

12.312.1.1 value

```
template<class T>
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

12.313 SimdChooser< T, true, true > Struct Template Reference

```
#include <fflas_simd.h>
```

Public Types

- using value = NoSimd<T>

12.313.1 Member Typedef Documentation

12.313.1.1 value

```
template<class T>
using value = NoSimd<T>
```

The documentation for this struct was generated from the following file:

- fflas_simd.h

12.314 simdToType< T > Struct Template Reference

The documentation for this struct was generated from the following file:

- fflas_simd.h

12.315 Single Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.316 Sparse< Field, SparseMatrix_t, IdxT, PtrT > Struct Template Reference

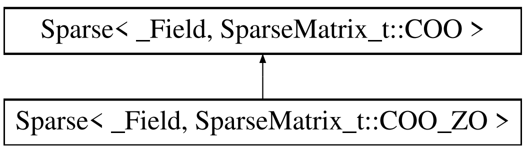
The documentation for this struct was generated from the following file:

- fflas_sparse.h

12.317 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

12.317.1 Member Typedef Documentation

12.317.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.317.2 Field Documentation

12.317.2.1 col

```
template<class _Field>
index_t* col = nullptr
```

12.317.2.2 row

```
template<class _Field>
index_t* row = nullptr
```

12.317.2.3 dat

```
template<class _Field>
_Field::Element_ptr dat
```

12.317.2.4 delayed

```
template<class _Field>
bool delayed = false
```

12.317.2.5 kmax

```
template<class _Field>
uint64_t kmax = 0
```

12.317.2.6 m

```
template<class _Field>
index_t m = 0
```

12.317.2.7 n

```
template<class _Field>
index_t n = 0
```

12.317.2.8 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.317.2.9 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.317.2.10 maxrow

```
template<class _Field>
uint64_t maxrow = 0
```

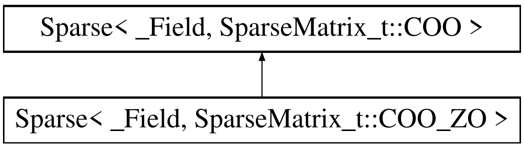
The documentation for this struct was generated from the following file:

- coo.h

12.318 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

12.318.1 Member Typedef Documentation

12.318.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.318.2 Field Documentation

12.318.2.1 cst

```
template<class _Field>
_Field::Element cst = 1
```

12.318.2.2 col

```
template<class _Field>
index_t* col = nullptr  [inherited]
```

12.318.2.3 row

```
template<class _Field>
index_t* row = nullptr  [inherited]
```

12.318.2.4 dat

```
template<class _Field>
_Field::Element_ptr dat  [inherited]
```

12.318.2.5 delayed

```
template<class _Field>
bool delayed = false  [inherited]
```

12.318.2.6 kmax

```
template<class _Field>
uint64_t kmax = 0  [inherited]
```

12.318.2.7 m

```
template<class _Field>
index_t m = 0  [inherited]
```

12.318.2.8 n

```
template<class _Field>
index_t n = 0  [inherited]
```

12.318.2.9 nnz

```
template<class _Field>
uint64_t nnz = 0  [inherited]
```

12.318.2.10 nElements

```
template<class _Field>
uint64_t nElements = 0  [inherited]
```

12.318.2.11 maxrow

```
template<class _Field>
uint64_t maxrow = 0  [inherited]
```

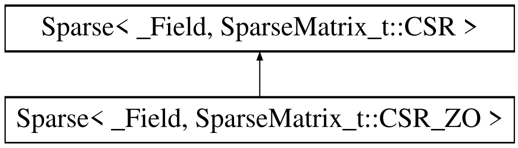
The documentation for this struct was generated from the following file:

- coo.h

12.319 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

12.319.1 Member Typedef Documentation

12.319.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.319.2 Field Documentation

12.319.2.1 delayed

```
template<class _Field>
bool delayed = false
```

12.319.2.2 kmax

```
template<class _Field>
uint64_t kmax = 0
```

12.319.2.3 m

```
template<class _Field>
index_t m = 0
```

12.319.2.4 n

```
template<class _Field>
index_t n = 0
```

12.319.2.5 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.319.2.6 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.319.2.7 maxrow

```
template<class _Field>
uint64_t maxrow = 0
```

12.319.2.8 col

```
template<class _Field>
index_t* col = nullptr
```

12.319.2.9 st

```
template<class _Field>
index_t* st = nullptr
```

12.319.2.10 stend

```
template<class _Field>
index_t* stend = nullptr
```

12.319.2.11 dat

```
template<class _Field>
_Field::Element_ptr dat
```

The documentation for this struct was generated from the following file:

- csr.h

12.320 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

12.320.1 Member Typedef Documentation

12.320.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.320.2 Field Documentation

12.320.2.1 delayed

```
template<class _Field>
bool delayed = false
```

12.320.2.2 col

```
template<class _Field>
index_t* col = nullptr
```

12.320.2.3 st

```
template<class _Field>
index_t* st = nullptr
```

12.320.2.4 dat

```
template<class _Field>
_Field::Element_ptr dat
```

12.320.2.5 kmax

```
template<class _Field>
uint64_t kmax = 0
```

12.320.2.6 m

```
template<class _Field>
index_t m = 0
```

12.320.2.7 n

```
template<class _Field>
index_t n = 0
```

12.320.2.8 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.320.2.9 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.320.2.10 maxrow

```
template<class _Field>
uint64_t maxrow = 0
```

12.320.2.11 nOnes

```
template<class _Field>
uint64_t nOnes = 0
```

12.320.2.12 nMOnes

```
template<class _Field>
uint64_t nMOnes = 0
```

12.320.2.13 nOthers

```
template<class _Field>
uint64_t nOthers = 0
```

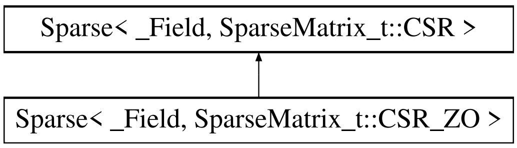
The documentation for this struct was generated from the following file:

- csr_hyb.h

12.321 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

12.321.1 Member Typedef Documentation

12.321.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.321.2 Field Documentation

12.321.2.1 cst

```
template<class _Field>
int64_t cst = 1
```

12.321.2.2 delayed

```
template<class _Field>
bool delayed = false
```

12.321.2.3 kmax

```
template<class _Field>
uint64_t kmax = 0 [inherited]
```

12.321.2.4 m

```
template<class _Field>
index_t m = 0 [inherited]
```

12.321.2.5 n

```
template<class _Field>
index_t n = 0 [inherited]
```

12.321.2.6 nnz

```
template<class _Field>
uint64_t nnz = 0 [inherited]
```

12.321.2.7 nElements

```
template<class _Field>
uint64_t nElements = 0 [inherited]
```

12.321.2.8 maxrow

```
template<class _Field>
uint64_t maxrow = 0 [inherited]
```

12.321.2.9 col

```
template<class _Field>
index_t* col = nullptr [inherited]
```

12.321.2.10 st

```
template<class _Field>
index_t* st = nullptr [inherited]
```

12.321.2.11 stend

```
template<class _Field>
index_t* stend = nullptr  [inherited]
```

12.321.2.12 dat

```
template<class _Field>
_Field::Element_ptr dat  [inherited]
```

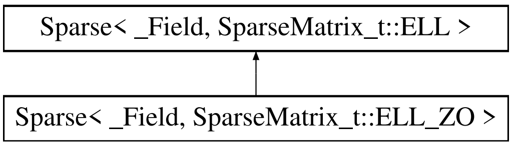
The documentation for this struct was generated from the following file:

- csr.h

12.322 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

12.322.1 Member Typedef Documentation

12.322.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.322.2 Field Documentation

12.322.2.1 delayed

```
template<class _Field>
bool delayed = false
```

12.322.2.2 kmax

```
template<class _Field>
uint64_t kmax = 0
```

12.322.2.3 m

```
template<class _Field>
index_t m = 0
```

12.322.2.4 n

```
template<class _Field>
index_t n = 0
```

12.322.2.5 ld

```
template<class _Field>
index_t ld = 0
```

12.322.2.6 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.322.2.7 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.322.2.8 maxrow

```
template<class _Field>
uint64_t maxrow = 0
```

12.322.2.9 col

```
template<class _Field>
index_t* col = nullptr
```

12.322.2.10 dat

```
template<class _Field>
_Field::Element_ptr dat
```

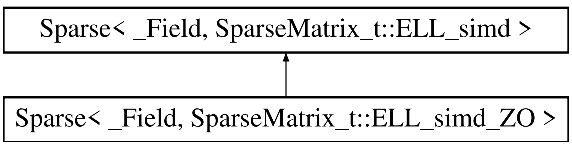
The documentation for this struct was generated from the following file:

- ell.h

12.323 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

12.323.1 Field Documentation

12.323.1.1 delayed

```
template<class _Field>
bool delayed = false
```

12.323.1.2 chunk

```
template<class _Field>
int chunk = 0
```

12.323.1.3 m

```
template<class _Field>
index_t m = 0
```

12.323.1.4 n

```
template<class _Field>
index_t n = 0
```

12.323.1.5 ld

```
template<class _Field>
index_t ld = 0
```

12.323.1.6 kmax

```
template<class _Field>
uint64_t kmax = 0
```

12.323.1.7 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.323.1.8 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.323.1.9 maxrow

```
template<class _Field>
uint64_t maxrow = 0
```

12.323.1.10 nChunks

```
template<class _Field>
uint64_t nChunks = 0
```

12.323.1.11 col

```
template<class _Field>
index_t* col = nullptr
```

12.323.1.12 dat

```
template<class _Field>
_Field::Element_ptr dat
```

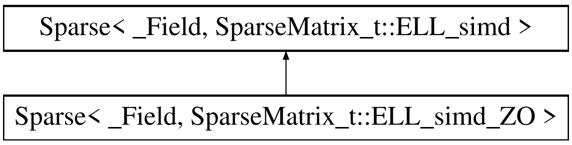
The documentation for this struct was generated from the following file:

- ell_simd.h

12.324 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

12.324.1 Field Documentation

12.324.1.1 cst

```
template<class _Field>
_Field::Element cst = 1
```

12.324.1.2 delayed

```
template<class _Field>
bool delayed = false [inherited]
```

12.324.1.3 chunk

```
template<class _Field>
int chunk = 0 [inherited]
```

12.324.1.4 m

```
template<class _Field>
index_t m = 0 [inherited]
```

12.324.1.5 n

```
template<class _Field>
index_t n = 0 [inherited]
```

12.324.1.6 ld

```
template<class _Field>
index_t ld = 0 [inherited]
```

12.324.1.7 kmax

```
template<class _Field>
uint64_t kmax = 0 [inherited]
```

12.324.1.8 nnz

```
template<class _Field>
uint64_t nnz = 0 [inherited]
```

12.324.1.9 nElements

```
template<class _Field>
uint64_t nElements = 0 [inherited]
```

12.324.1.10 maxrow

```
template<class _Field>
uint64_t maxrow = 0 [inherited]
```

12.324.1.11 nChunks

```
template<class _Field>
uint64_t nChunks = 0 [inherited]
```

12.324.1.12 col

```
template<class _Field>
index_t* col = nullptr [inherited]
```

12.324.1.13 dat

```
template<class _Field>
_Field::Element_ptr dat [inherited]
```

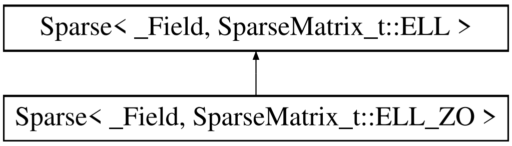
The documentation for this struct was generated from the following file:

- ell_simd.h

12.325 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

12.325.1 Member Typedef Documentation

12.325.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.325.2 Field Documentation

12.325.2.1 cst

```
template<class _Field>
_Field::Element cst = 1
```

12.325.2.2 delayed

```
template<class _Field>
bool delayed = false  [inherited]
```

12.325.2.3 kmax

```
template<class _Field>
uint64_t kmax = 0  [inherited]
```

12.325.2.4 m

```
template<class _Field>
index_t m = 0  [inherited]
```

12.325.2.5 n

```
template<class _Field>
index_t n = 0  [inherited]
```

12.325.2.6 ld

```
template<class _Field>
index_t ld = 0  [inherited]
```

12.325.2.7 nnz

```
template<class _Field>
uint64_t nnz = 0  [inherited]
```

12.325.2.8 nElements

```
template<class _Field>
uint64_t nElements = 0  [inherited]
```

12.325.2.9 maxrow

```
template<class _Field>
uint64_t maxrow = 0  [inherited]
```

12.325.2.10 col

```
template<class _Field>
index_t* col = nullptr  [inherited]
```

12.325.2.11 dat

```
template<class _Field>
_Field::Element_ptr dat  [inherited]
```

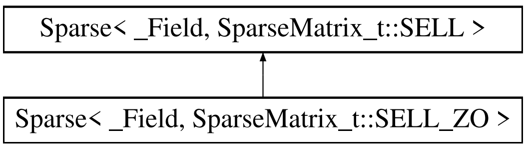
The documentation for this struct was generated from the following file:

- ell.h

12.326 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

12.326.1 Member Typedef Documentation

12.326.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.326.2 Field Documentation

12.326.2.1 delayed

```
template<class _Field>
bool delayed = false
```

12.326.2.2 chunk

```
template<class _Field>
int chunk = 0
```

12.326.2.3 kmax

```
template<class _Field>
index_t kmax = 0
```

12.326.2.4 m

```
template<class _Field>
index_t m = 0
```

12.326.2.5 n

```
template<class _Field>
index_t n = 0
```

12.326.2.6 maxrow

```
template<class _Field>
index_t maxrow = 0
```

12.326.2.7 sigma

```
template<class _Field>
index_t sigma = 0
```

12.326.2.8 nChunks

```
template<class _Field>
index_t nChunks = 0
```

12.326.2.9 nnz

```
template<class _Field>
uint64_t nnz = 0
```

12.326.2.10 nElements

```
template<class _Field>
uint64_t nElements = 0
```

12.326.2.11 perm

```
template<class _Field>
index_t* perm = nullptr
```

12.326.2.12 st

```
template<class _Field>
uint64_t* st = nullptr
```

12.326.2.13 chunkSize

```
template<class _Field>
index_t* chunkSize = nullptr
```

12.326.2.14 col

```
template<class _Field>
index_t* col = nullptr
```

12.326.2.15 dat

```
template<class _Field>
_Field::Element_ptr dat
```

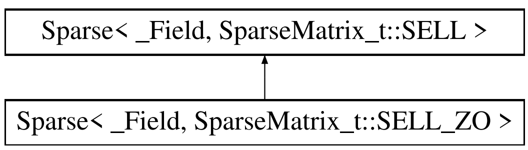
The documentation for this struct was generated from the following file:

- sell.h

12.327 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

12.327.1 Member Typedef Documentation

12.327.1.1 Field

```
template<class _Field>
using Field = _Field
```

12.327.2 Field Documentation

12.327.2.1 cst

```
template<class _Field>
_Field::Element cst = 1
```

12.327.2.2 delayed

```
template<class _Field>
bool delayed = false  [inherited]
```

12.327.2.3 chunk

```
template<class _Field>
int chunk = 0  [inherited]
```

12.327.2.4 kmax

```
template<class _Field>
index_t kmax = 0  [inherited]
```

12.327.2.5 m

```
template<class _Field>
index_t m = 0  [inherited]
```

12.327.2.6 n

```
template<class _Field>
index_t n = 0  [inherited]
```

12.327.2.7 maxrow

```
template<class _Field>
index_t maxrow = 0  [inherited]
```

12.327.2.8 sigma

```
template<class _Field>
index_t sigma = 0  [inherited]
```

12.327.2.9 nChunks

```
template<class _Field>
index_t nChunks = 0  [inherited]
```

12.327.2.10 nnz

```
template<class _Field>
uint64_t nnz = 0  [inherited]
```

12.327.2.11 nElements

```
template<class _Field>
uint64_t nElements = 0  [inherited]
```

12.327.2.12 perm

```
template<class _Field>
index_t* perm = nullptr  [inherited]
```

12.327.2.13 st

```
template<class _Field>
uint64_t* st = nullptr  [inherited]
```

12.327.2.14 chunkSize

```
template<class _Field>
index_t* chunkSize = nullptr  [inherited]
```

12.327.2.15 col

```
template<class _Field>
index_t* col = nullptr  [inherited]
```

12.327.2.16 dat

```
template<class _Field>
_Field::Element_ptr dat  [inherited]
```

The documentation for this struct was generated from the following file:

- sell.h

12.328 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

12.328.1 Field Documentation

12.328.1.1 _coo

```
template<class Field, int flag = HelperFlag::none>
FFLAS::CooMat<Field>* _coo = nullptr
```

12.328.1.2 _csr

```
template<class Field, int flag = HelperFlag::none>
FFLAS::CsrMat<Field>* _csr = nullptr
```

12.328.1.3 _ell

```
template<class Field, int flag = HelperFlag::none>
FFLAS::EllMat<Field>* _ell = nullptr
```

The documentation for this struct was generated from the following file:

- fflas_sparse.h

12.329 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

12.329.1 Field Documentation

12.329.1.1 rowdim

```
uint64_t rowdim = 0
```

12.329.1.2 coldim

```
uint64_t coldim = 0
```

12.329.1.3 nOnes

```
uint64_t nOnes = 0
```

12.329.1.4 nMOnes

```
uint64_t nMOnes = 0
```

12.329.1.5 nOthers

```
uint64_t nOthers = 0
```

12.329.1.6 nnz

```
uint64_t nnz = 0
```

12.329.1.7 maxRow

```
uint64_t maxRow = 0
```

12.329.1.8 minRow

```
uint64_t minRow = 0
```

12.329.1.9 averageRow

```
uint64_t averageRow = 0
```

12.329.1.10 deviationRow

```
uint64_t deviationRow = 0
```

12.329.1.11 maxCol

```
uint64_t maxCol = 0
```

12.329.1.12 minCol

```
uint64_t minCol = 0
```

12.329.1.13 averageCol

```
uint64_t averageCol = 0
```

12.329.1.14 deviationCol

```
uint64_t deviationCol = 0
```

12.329.1.15 minColDifference

```
uint64_t minColDifference = 0
```

12.329.1.16 maxColDifference

```
uint64_t maxColDifference = 0
```

12.329.1.17 averageColDifference

```
uint64_t averageColDifference = 0
```

12.329.1.18 deviationColDifference

```
uint64_t deviationColDifference = 0
```

12.329.1.19 minRowDifference

```
uint64_t minRowDifference = 0
```

12.329.1.20 maxRowDifference

```
uint64_t maxRowDifference = 0
```

12.329.1.21 averageRowDifference

```
uint64_t averageRowDifference = 0
```

12.329.1.22 deviationRowDifference

```
uint64_t deviationRowDifference = 0
```

12.329.1.23 nDenseRows

```
uint64_t nDenseRows = 0
```

12.329.1.24 nDenseCols

```
uint64_t nDenseCols = 0
```

12.329.1.25 nEmptyRows

```
uint64_t nEmptyRows = 0
```

12.329.1.26 nEmptyCols

```
uint64_t nEmptyCols = 0
```

12.329.1.27 nEmptyColsEnd

```
uint64_t nEmptyColsEnd = 0
```

12.329.1.28 denseRows

```
std::vector<uint64_t> denseRows
```

12.329.1.29 denseCols

```
std::vector<uint64_t> denseCols
```

The documentation for this struct was generated from the following file:

- utils.h

12.330 string Class Reference

STL class.

Data Structures

- class const_iterator
  STL iterator class.
- class const_reverse_iterator
  STL iterator class.
- class iterator
  STL iterator class.
- class reverse_iterator
  STL iterator class.

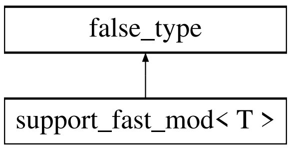
12.330.1 Detailed Description

STL class.
The documentation for this class was generated from the following files:

12.331 support_fast_mod< T > Struct Template Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support_fast_mod< T >:



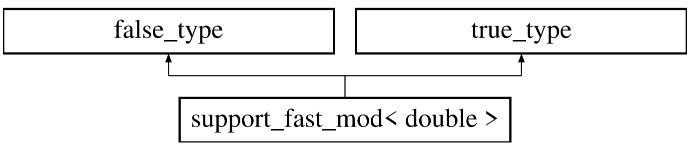
The documentation for this struct was generated from the following file:

- fflas_freduce.h

12.332 support_fast_mod< double > Struct Reference

```
#include <fflas_freduce.h>
```

Inheritance diagram for support_fast_mod< double >:



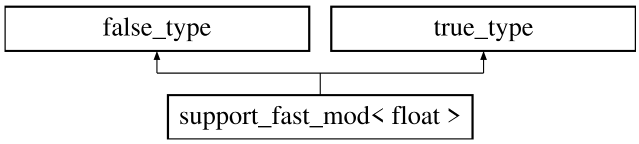
The documentation for this struct was generated from the following file:

- fflas_freduce.h

12.333 support_fast_mod< float > Struct Reference

`#include <fflas_freduce.h>`

Inheritance diagram for support_fast_mod< float >:



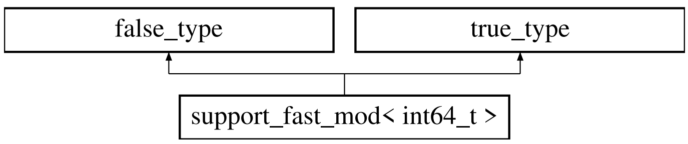
The documentation for this struct was generated from the following file:

- fflas_freduce.h

12.334 support_fast_mod< int64_t > Struct Reference

`#include <fflas_freduce.h>`

Inheritance diagram for support_fast_mod< int64_t >:



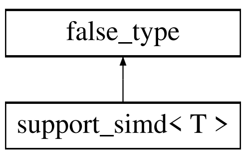
The documentation for this struct was generated from the following file:

- fflas_freduce.h

12.335 support_simd< T > Struct Template Reference

`#include <fflas_simd.h>`

Inheritance diagram for support_simd< T >:



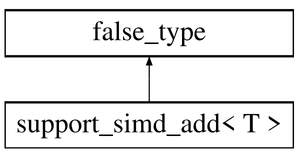
The documentation for this struct was generated from the following file:

- fflas_simd.h

12.336 support_simd_add< T > Struct Template Reference

`#include <fflas_fadd.h>`

Inheritance diagram for support_simd_add< T >:



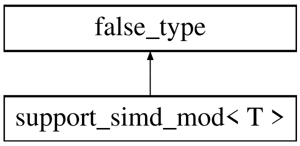
The documentation for this struct was generated from the following file:

- fflas_fadd.h

12.337 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

12.338 Test< Elt > Class Template Reference

Public Types

- using Field = Modular<Elt>
- using Elt_ptr = typename Field::Element_ptr
- using Residu = typename Field::Residu_t
- template<bool B, class T = void>
  using enable_if_t = typename std::enable_if<B, T>::type
- template<typename Simd>
  using is_same_element = typename Simd::template is_same_element<Field>
- template<typename E>
  using enable_if_no_simd_t = enable_if_t<Simd<E>::vect_size == 1>
- template<typename E>
  using enable_if_simd128_t = enable_if_t<sizeof(E)∗Simd<E>::vect_size == 16>
- template<typename E>
  using enable_if_simd256_t = enable_if_t<sizeof(E)∗Simd<E>::vect_size == 32>
- template<typename E>
  using enable_if_simd512_t = enable_if_t<sizeof(E)∗Simd<E>::vect_size == 64>

Public Member Functions

- Test (size_t mm, size_t nn)
- template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > ∗ = nullptr>
  bool test_ftranspose (size_t m, size_t n, Elt_ptr A, size_t lda, Elt_ptr B, size_t ldb)
- template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > ∗ = nullptr>
  bool doTests ()
- template<typename _E = Elt, enable_if_t< is_same< _E, Elt >::value > ∗ = nullptr, enable_if_no_simd_t< _E > ∗ = nullptr>
  bool run ()

Static Public Member Functions

- template<typename _E = Elt, enable_if_t<!is_same< _E, Givaro::Integer >::value > ∗ = nullptr>
  static Residu cardinality ()
- template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > ∗ = nullptr>
  static Residu cardinality ()

Protected Attributes

- Field F
- size_t _mm
- size_t _nn

12.338.1 Member Typedef Documentation

12.338.1.1 Field

```
template<typename Elt>
using Field = Modular<Elt>
```

12.338.1.2 Elt_ptr

```
template<typename Elt>
using Elt_ptr = typename Field::Element_ptr
```

12.338.1.3 Residu

```
template<typename Elt>
using Residu = typename Field::Residu_t
```

12.338.1.4 enable_if_t

```
template<typename Elt>
template<bool B, class T = void>
using enable_if_t = typename std::enable_if<B, T>::type
```

12.338.1.5 is_same_element

```
template<typename Elt>
template<typename Simd>
using is_same_element = typename Simd::template is_same_element<Field>
```

12.338.1.6 enable_if_no_simd_t

```
template<typename Elt>
template<typename E>
using enable_if_no_simd_t = enable_if_t<Simd<E>::vect_size == 1>
```

12.338.1.7 enable_if_simd128_t

```
template<typename Elt>
template<typename E>
using enable_if_simd128_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 16>
```

12.338.1.8 enable_if_simd256_t

```
template<typename Elt>
template<typename E>
using enable_if_simd256_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 32>
```

12.338.1.9 enable_if_simd512_t

```
template<typename Elt>
template<typename E>
using enable_if_simd512_t = enable_if_t<sizeof(E)*Simd<E>::vect_size == 64>
```

12.338.2 Constructor & Destructor Documentation

12.338.2.1 Test()

```
template<typename Elt>
Test (
            size_t mm,
            size_t nn)  [inline]
```

12.338.3 Member Function Documentation

12.338.3.1 cardinality() [1/2]

```
template<typename Elt>
template<typename _E = Elt, enable_if_t<!is_same< _E, Givaro::Integer >::value > * = nullptr>
static Residu cardinality ()  [inline], [static]
```

12.338.3.2 cardinality() [2/2]

```
template<typename Elt>
template<typename _E = Elt, enable_if_t< is_same< _E, Givaro::Integer >::value > * = nullptr>
static Residu cardinality ()  [inline], [static]
```

12.338.3.3 test_ftranspose()

```
template<typename Elt>
template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > * =
nullptr>
bool test_ftranspose (
            size_t m,
            size_t n,
            Elt_ptr A,
            size_t lda,
            Elt_ptr B,
            size_t ldb)  [inline]
```

12.338.3.4 doTests()

```
template<typename Elt>
template<typename Simd = NoSimd<Elt>, enable_if_t< is_same_element< Simd >::value > * =
nullptr>
bool doTests ()  [inline]
```

12.338.3.5 run()

```
template<typename Elt>
template<typename _E = Elt, enable_if_t< is_same< _E, Elt >::value > * = nullptr, enable_if_no_simd_t<
_E > * = nullptr>
bool run ()  [inline]
```

12.338.4 Field Documentation

12.338.4.1 F

```
template<typename Elt>
Field F  [protected]
```

12.338.4.2 _mm

```
template<typename Elt>
size_t _mm  [protected]
```

12.338.4.3 _nn

```
template<typename Elt>
size_t _nn [protected]
```

The documentation for this class was generated from the following file:

- test-storage-transpose.C

12.339 TestOneMethod< Simd > Class Template Reference

Public Types

- using Element = typename Simd::scalar_t
- using vect_t = typename Simd::vect_t
- using vectElt = vector<Element>
- template<bool B, typename T = void>
  using enable_if_t = typename enable_if<B, T>::type

Public Member Functions

- template<typename... AScal, typename RScal, typename... ASimd, typename RSimd, enable_if_t< sizeof...(AScal)==sizeof...(ASimd)> * = nullptr, enable_if_t< count_nonconst_lvalue_reference< AScal... >::n==count_nonconst_lvalue_reference< ASimd... >::n > * = nullptr, enable_if_t< is_all_same< AScal... >::value > * = nullptr, enable_if_t< is_all_same< vect_t, ASimd... >::value > * = nullptr>
  TestOneMethod (function< RSimd(ASimd...)> fsimd, function< RScal(AScal...)> fscal, function< void(vector< vectElt > &)> genInputs, string fname)
- template<typename Ret, typename... AScal>
  enable_if_t< is_all_same< Element, AScal... >::value &&std::is_convertible< Ret, Element >::value, void > evaluate_scalar_method (function< Ret(AScal...)> fscal)
- template<typename... AScal>
  enable_if_t< is_all_same< vectElt, AScal... >::value, void > evaluate_scalar_method (function< vectElt(AScal...)> fscal)
- template<typename... AScal>
  enable_if_t< is_all_same< vectElt, AScal... >::value, void > evaluate_scalar_method (function< void(AScal...)> fscal)
- template<typename Ret, typename... ASimd>
  enable_if_t< is_all_same< vect_t, ASimd... >::value &&std::is_convertible< Ret, vect_t >::value, void > evaluate_simd_method (function< Ret(ASimd...)> fsimd, array< vect_t, sizeof...(ASimd)> &simd_in)
- template<typename... ASimd>
  enable_if_t< is_all_same< vect_t, ASimd... >::value, void > evaluate_simd_method (function< void(ASimd...)> fsimd, array< vect_t, sizeof...(ASimd)> &simd_in)
- bool getStatus () const
- string getTestName () const
- bool writeResultLine () const
- void writeDebugData () const

Static Public Attributes

- static constexpr size_t vect_size = Simd::vect_size

Protected Attributes

- size_t nb_lref
- string name
- vector< vectElt > inputs
- vector< vectElt > outputs_simd
- vector< vectElt > outputs_scalar

12.339.1 Member Typedef Documentation

12.339.1.1 Element

```
template<typename Simd>
using Element = typename Simd::scalar_t
```

12.339.1.2 vect_t

```
template<typename Simd>
using vect_t = typename Simd::vect_t
```

12.339.1.3 vectElt

```
template<typename Simd>
using vectElt = vector<Element>
```

12.339.1.4 enable_if_t

```
template<typename Simd>
template<bool B, typename T = void>
using enable_if_t = typename enable_if<B, T>::type
```

12.339.2 Constructor & Destructor Documentation

12.339.2.1 TestOneMethod()

```
template<typename Simd>
template<typename... AScal, typename RScal, typename... ASimd, typename RSimd, enable_if_t<
sizeof...(AScal)==sizeof...(ASimd)> * = nullptr, enable_if_t< count_nonconst_lvalue_reference<
AScal... >::n==count_nonconst_lvalue_reference< ASimd... >::n > * = nullptr, enable_if_t<
is_all_same< AScal... >::value > * = nullptr, enable_if_t< is_all_same< vect_t, ASimd...
>::value > * = nullptr>
TestOneMethod (
            function< RSimd(ASimd...)> fsimd,
            function< RScal(AScal...)> fscal,
            function< void(vector< vectElt > &)> genInputs,
            string fname) [inline]
```

12.339.3 Member Function Documentation

12.339.3.1 evaluate_scalar_method() [1/3]

```
template<typename Simd>
template<typename Ret, typename... AScal>
enable_if_t< is_all_same< Element, AScal... >::value &&std::is_convertible< Ret, Element >↩
::value, void > evaluate_scalar_method (
            function< Ret(AScal...)> fscal) [inline]
```

12.339.3.2 evaluate_scalar_method() [2/3]

```
template<typename Simd>
template<typename... AScal>
enable_if_t< is_all_same< vectElt, AScal... >::value, void > evaluate_scalar_method (
            function< vectElt(AScal...)> fscal) [inline]
```

12.339.3.3 evaluate_scalar_method() [3/3]

```
template<typename Simd>
template<typename... AScal>
```

```
enable_if_t< is_all_same< vectElt, AScal...  >::value, void > evaluate_scalar_method (
            function< void(AScal...)> fscal)  [inline]
```

12.339.3.4 evaluate_simd_method() [1/2]

```
template<typename Simd>
template<typename Ret, typename...  ASimd>
enable_if_t< is_all_same< vect_t, ASimd...  >::value &&std::is_convertible< Ret, vect_t >↩
::value, void > evaluate_simd_method (
            function< Ret(ASimd...)> fsimd,
            array< vect_t, sizeof...(ASimd)> & simd_in)  [inline]
```

12.339.3.5 evaluate_simd_method() [2/2]

```
template<typename Simd>
template<typename...  ASimd>
enable_if_t< is_all_same< vect_t, ASimd...  >::value, void > evaluate_simd_method (
            function< void(ASimd...)> fsimd,
            array< vect_t, sizeof...(ASimd)> & simd_in)  [inline]
```

12.339.3.6 getStatus()

```
template<typename Simd>
bool getStatus () const  [inline]
```

12.339.3.7 getTestName()

```
template<typename Simd>
string getTestName () const  [inline]
```

12.339.3.8 writeResultLine()

```
template<typename Simd>
bool writeResultLine () const  [inline]
```

12.339.3.9 writeDebugData()

```
template<typename Simd>
void writeDebugData () const  [inline]
```

12.339.4 Field Documentation

12.339.4.1 vect_size

```
template<typename Simd>
size_t vect_size = Simd::vect_size  [static], [constexpr]
```

12.339.4.2 nb_lref

```
template<typename Simd>
size_t nb_lref  [protected]
```

12.339.4.3 name

```
template<typename Simd>
string name  [protected]
```

12.339.4.4 inputs

```
template<typename Simd>
vector<vectElt> inputs  [protected]
```

12.339.4.5 outputs_simd

```
template<typename Simd>
vector<vectElt> outputs_simd [protected]
```

12.339.4.6 outputs_scalar

```
template<typename Simd>
vector<vectElt> outputs_scalar [protected]
```

The documentation for this class was generated from the following file:

- test-simd.C

12.340 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)...))

12.340.1 Member Function Documentation

12.340.1.1 operator()()

```
template<typename... Args>
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

12.341 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)...))

12.341.1 Member Function Documentation

12.341.1.1 operator()()

```
template<typename... Args>
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

12.342 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)...))

12.342.1 Member Function Documentation

12.342.1.1 operator()()

```
template<typename... Args>
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

12.343 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)...))

12.343.1 Member Function Documentation

12.343.1.1 operator()()

```
template<typename... Args>
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

12.344 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)...))

12.344.1 Member Function Documentation

12.344.1.1 operator()()

```
template<typename... Args>
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

12.345 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)...))

12.345.1 Member Function Documentation

12.345.1.1 operator()()

```
template<typename...  Args>
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

12.346 Threads Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.347 ThreeD Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.348 ThreeDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.349 ThreeDInPlace Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.350 TRSMHelper< RecIterTrait, 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

12.350.1 Detailed Description

template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
struct FFLAS::TRSMHelper< RecIterTrait, ParSeqTrait >

TRSM Helper.

12.350.2 Constructor & Destructor Documentation

12.350.2.1 TRSMHelper() [1/3]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
template<class Cut, class Param>
TRSMHelper (
            ParSeqHelper::Parallel< Cut, Param > _PS)  [inline]
```

12.350.2.2 TRSMHelper() [2/3]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
TRSMHelper (
            ParSeqHelper::Sequential _PS)  [inline]
```

12.350.2.3 TRSMHelper() [3/3]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
template<typename RIT, typename PST>
TRSMHelper (
            TRSMHelper< RIT, PST > & _TH)  [inline]
```

12.350.3 Member Function Documentation

12.350.3.1 pMMH() [1/2]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeqHelper::Sequential>
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  [inline]
```

12.350.3.2 pMMH() [2/2]

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeq←
Helper::Sequential>
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 [inline]
```

12.350.4 Field Documentation

12.350.4.1 parseq

```
template<typename RecIterTrait = StructureHelper::Recursive, typename ParSeqTrait = ParSeq←
Helper::Sequential>
ParSeqTrait parseq
```

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.351 TwoD Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.352 TwoDAdaptive Struct Reference

The documentation for this struct was generated from the following file:

- blockcuts.inl

12.353 UnparametricTag Struct Reference

If the field uses a representation with infix operators.

`#include <field-traits.h>`

12.353.1 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

12.354 vector< T > Class Template Reference

STL class.

Data Structures

- class const_iterator

  STL iterator class.
- class const_reverse_iterator

  STL iterator class.

- class iterator

  STL iterator class.
- class reverse_iterator

  STL iterator class.

Data Fields

- T elements

  STL member.

12.354.1 Detailed Description

STL class.

12.354.2 Field Documentation

12.354.2.1 elements

```
T elements
```
STL member.

The documentation for this class was generated from the following files:

12.355 width< T > Struct Template Reference

Static Public Attributes

- static constexpr size_t value = 2+2∗sizeof(T)

12.355.1 Field Documentation

12.355.1.1 value

```
template<typename T>
size_t value = 2+2*sizeof(T)  [static], [constexpr]
```
The documentation for this struct was generated from the following file:

- test-simd.C

12.356 width< double > Struct Reference

Static Public Attributes

- static constexpr size_t value = 24
- static constexpr size_t value

12.356.1 Field Documentation

12.356.1.1 value [1/2]

```
size_t value = 24  [static], [constexpr]
```

12.356.1.2 value [2/2]

```
size_t value  [static], [constexpr]
```
The documentation for this struct was generated from the following file:

- test-simd.C

12.357 width< float > Struct Reference

Static Public Attributes

- static constexpr size_t value = 16
- static constexpr size_t value

12.357.1 Field Documentation

12.357.1.1 value [1/2]

```
size_t value = 16  [static], [constexpr]
```

12.357.1.2 value [2/2]

```
size_t value  [static], [constexpr]
```

The documentation for this struct was generated from the following file:

- test-simd.C

12.358 Winograd Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

12.359 WinogradPar Struct Reference

The documentation for this struct was generated from the following file:

- fflas_helpers.inl

Chapter 13

File Documentation

13.1 arithprog.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
```

Typedefs

- typedef Givaro::OMPTimer TTimer

Functions

- int main (int argc, char **argv)

13.1.1 Typedef Documentation

13.1.1.1 TTimer

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

13.1.2 Function Documentation

13.1.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.2 autotune/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)
- #define GFOPS(m, n, r, t)

Typedefs

- typedef Givaro::Timer TTimer

Functions

- int main ()

13.2.1 Macro Definition Documentation

13.2.1.1 CUBE

```
#define CUBE(
              x)
```

Value:

```
((x)*(x)*(x))
```

13.2.1.2 GFOPS

```
#define GFOPS(
              m,
              n,
              r,
              t)
```

Value:

```
(2.7*CUBE(double(n)/1000.0))/t
```

13.2.2 Typedef Documentation

13.2.2.1 TTimer

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

13.2.3 Function Documentation

13.2.3.1 main()

```
int main (
             void )
```

13.3 examples/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.

13.3.1 Function Documentation

13.3.1.1 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.

13.4 fsyrk.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)
- #define GFOPS(n, t)

Functions

- int main ()

13.4.1 Macro Definition Documentation

13.4.1.1 CUBE

```
#define CUBE(
              x)
```

Value:

```
((x)*(x)*(x))
```

13.4.1.2 GFOPS

```
#define GFOPS(
               n,
               t)
```

Value:

```
(CUBE(double(n)/1000.0)/(3.0*t))
```

13.4.2 Function Documentation

13.4.2.1 main()

```
int main (
           void )
```

13.5 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)
- #define GFOPS(n, t)

Functions

- int main ()

13.5.1 Macro Definition Documentation

13.5.1.1 CUBE

```
#define CUBE(
                x)
```

Value:

```
((x)*(x)*(x))
```

13.5.1.2 GFOPS

```
#define GFOPS(
                n,
                t)
```

Value:

```
(CUBE(double(n)/1000.0)/(3.0*t))
```

13.5.2 Function Documentation

13.5.2.1 main()

```
int main (
            void )
```

13.6 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)
- #define GFOPS(n, t)

Functions

- int main ()

13.6.1 Macro Definition Documentation

13.6.1.1 CUBE

```
#define CUBE(
                x)
```

Value:

```
((x)*(x)*(x))
```

13.6.1.2 GFOPS

```
#define GFOPS(
              n,
              t)
```

Value:

```
(CUBE(double(n)/1000.0)/(3.0*t))
```

13.6.2 Function Documentation

13.6.2.1 main()

```
int main (
            void )
```

13.7 autotune/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)
- #define GFOPS(m, n, r, t)

Functions

- int main ()

13.7.1 Macro Definition Documentation

13.7.1.1 CUBE

```
#define CUBE(
              x)
```

Value:

```
((x)*(x)*(x))
```

13.7.1.2 GFOPS

```
#define GFOPS(
              m,
              n,
              r,
              t)
```

Value:

```
(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
```

13.7.2 Function Documentation

13.7.2.1 main()

```
int main (
            void )
```

13.8 examples/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)

13.8.1 Function Documentation

13.8.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.9 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)

Functions

- template<class Field>
  bool balanced (const Field &)
- template<class T>
  bool balanced (const Givaro::ModularBalanced< T > &)
- int main ()

13.9.1 Macro Definition Documentation

13.9.1.1 DOUBLE_TO_FLOAT_CROSSOVER

```
#define DOUBLE_TO_FLOAT_CROSSOVER 0
```

13.9.1.2 GFOPS

```
#define GFOPS(
             n,
             t)
```

Value:

```
(2.0/t*(double)n/1000.0*(double)n/1000.0*(double)n/1000.0)
```

13.9.2 Function Documentation

13.9.2.1 balanced() [1/2]

```
template<class Field>
bool balanced (
            const Field & )
```

13.9.2.2 balanced() [2/2]

```
template<class T>
bool balanced (
            const Givaro::ModularBalanced< T > & )
```

13.9.2.3 main()

```
int main (
            void )
```

13.10 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/test-utils.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)

13.10.1 Macro Definition Documentation

13.10.1.1 __FFLASFFPACK_FORCE_SEQ

```
#define __FFLASFFPACK_FORCE_SEQ
```

13.10.2 Function Documentation

13.10.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.11 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/test-utils.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, uint64_t bits, size_t n, size_t d, size_t iter, std::string file, int variant, uint64_t seed)
- int main (int argc, char **argv)

13.11.1 Macro Definition Documentation

13.11.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.11.2 Function Documentation

13.11.2.1 run_with_field()

```
template<class Field>
void run_with_field (
            int q,
            uint64_t bits,
            size_t n,
            size_t d,
            size_t iter,
            std::string file,
            int variant,
            uint64_t seed)
```

13.11.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.12 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)

Functions

- int main (int argc, char **argv)

13.12.1 Macro Definition Documentation

13.12.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.12.1.2 _NR_TESTS

```
#define _NR_TESTS 5
```

13.12.1.3 _MAX_SIZE_MATRICES

```
#define _MAX_SIZE_MATRICES 1000
```

13.12.1.4 CUBE

```
#define CUBE(
              x)
```

Value:

```
((x)*(x)*(x))
```

13.12.2 Function Documentation

13.12.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.13 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)

13.13.1 Macro Definition Documentation

13.13.1.1 CBLAS_GEMM

```
#define CBLAS_GEMM cblas_dgemm
```

13.13.2 Typedef Documentation

13.13.2.1 TTimer

```
typedef FFLAS::Timer TTimer
```

13.13.2.2 Floats

```
typedef double Floats
```

13.13.3 Function Documentation

13.13.3.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.14 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

Functions

- int main (int argc, char **argv)

13.14.1 Macro Definition Documentation

13.14.1.1 __FFLASFFPACK_HAVE_DGETRF

```
#define __FFLASFFPACK_HAVE_DGETRF 1
```

13.14.2 Function Documentation

13.14.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.15 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"
```

Functions

- int main (int argc, char **argv)

13.15.1 Function Documentation

13.15.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.16 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)

Functions

- int main (int argc, char **argv)

13.16.1 Macro Definition Documentation

13.16.1.1 EFFGFF

```
#define EFFGFF(
            n,
            t,
            i)
```

Value:

```
( (double(n)/1000.*double(n)/1000.*double(n)/1000.0) / double(t) * double(i) / 3.)
```

13.16.2 Function Documentation

13.16.2.1 main()

```
int main (
          int argc,
          char ** argv)
```

13.17 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"
```

Functions

- int main (int argc, char **argv)

13.17.1 Function Documentation

13.17.1.1 main()

```
int main (
          int argc,
          char ** argv)
```

13.18 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

Functions

- int main (int argc, char **argv)

13.18.1 Macro Definition Documentation

13.18.1.1 __FFLASFFPACK_HAVE_DTRTRI

```
#define __FFLASFFPACK_HAVE_DTRTRI 1
```

13.18.2 Function Documentation

13.18.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.19 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)

13.19.1 Macro Definition Documentation

13.19.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.19.2 Function Documentation

13.19.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.20 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/test-utils.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 uint64_t BS, const size_t p, const size_↩ t threads, uint64_t seed)
- int main (int argc, char **argv)

13.20.1 Macro Definition Documentation

13.20.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.20.2 Function Documentation

13.20.2.1 run_with_field()

```
template<class Field>
Field::Element run_with_field (
            int q,
            size_t iter,
            size_t N,
            const uint64_t BS,
            const size_t p,
            const size_t threads,
            uint64_t seed)
```

13.20.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.21 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)

13.21.1 Macro Definition Documentation

13.21.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.21.1.2 MG_DEFAULT

```
#define MG_DEFAULT MG_ACTIVE
```

13.21.1.3 STD_RECINT_SIZE

```
#define STD_RECINT_SIZE 8
```

13.21.2 Function Documentation

13.21.2.1 tmain()

```
template<typename Ints>
int tmain ()
```

13.21.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.22 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[ ])

13.22.1 Macro Definition Documentation

13.22.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.22.2 Typedef Documentation

13.22.2.1 RNS

```
typedef FFPACK::rns_double RNS
```

13.22.2.2 Field

```
typedef FFPACK::RNSInteger<RNS> Field
```

13.22.2.3 Element_ptr

```
typedef Field::Element_ptr Element_ptr
```

13.22.2.4 ConstElement_ptr

```
typedef Field::ConstElement_ptr ConstElement_ptr
```

13.22.2.5 THREADS

```
typedef StrategyParameter::Threads THREADS
```

13.22.2.6 GRAIN

```
typedef StrategyParameter::Grain GRAIN
```

13.22.2.7 TWOD

```
typedef StrategyParameter::TwoD TWOD
```

13.22.2.8 TWODA

```
typedef StrategyParameter::TwoDAdaptive TWODA
```

13.22.2.9 THREED

```
typedef StrategyParameter::ThreeD THREED
```

13.22.2.10 THREEDA

```
typedef StrategyParameter::ThreeDAdaptive THREEDA
```

13.22.2.11 THREEDIP

```
typedef StrategyParameter::ThreeDInPlace THREEDIP
```

13.22.2.12 PSeq

```
typedef ParSeqHelper::Sequential PSeq
```

13.22.3 Function Documentation

13.22.3.1 main()

```
int main (
            int argc,
            char * argv[])
```

13.23 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)

13.23.1 Macro Definition Documentation

13.23.1.1 CLASSIC_HYBRID

```
#define CLASSIC_HYBRID
```

13.23.2 Function Documentation

13.23.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.24 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)

13.24.1 Macro Definition Documentation

13.24.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.24.1.2 MG_DEFAULT

```
#define MG_DEFAULT MG_ACTIVE
```

13.24.1.3 STD_RECINT_SIZE

```
#define STD_RECINT_SIZE 8
```

13.24.2 Function Documentation

13.24.2.1 write_matrix()

```
template<typename T>
std::ostream & write_matrix (
            std::ostream & out,
            Givaro::Integer p,
            size_t m,
            size_t n,
            T * C,
            size_t ldc)
```

13.24.2.2 tmain()

```
template<typename Ints>
int tmain ()
```

13.24.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.25 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, uint64_t 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, uint64_t 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, uint64_t 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, uint64_t bitsize, uint64_t seed, size_t iters, int t, arg &as, size_t GrainSize)
- int main (int argc, char **argv)

13.25.1 Macro Definition Documentation

13.25.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.25.2 Function Documentation

13.25.2.1 fill_value()

```
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)
```

13.25.2.2 genData()

```
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,
            uint64_t bitsize,
            uint64_t seed)
```

13.25.2.3 check_result()

```
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)
```

13.25.2.4 benchmark_with_timer()

```
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)
```

13.25.2.5 benchmark_disp()

```
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)
```

13.25.2.6 benchmark_in_Field()

```
template<class Field, class arg>
void benchmark_in_Field (
            Field & F,
            int p,
            size_t m,
            size_t k,
            int NBK,
            uint64_t bitsize,
            uint64_t seed,
            size_t iters,
            int t,
            arg & as,
            size_t GrainSize)
```

13.25.2.7 benchmark_with_field() [1/2]

```
template<class Field, class arg>
void benchmark_with_field (
            int p,
            size_t m,
            size_t k,
            int NBK,
            uint64_t bitsize,
            uint64_t seed,
            size_t iters,
            int t,
            arg & as,
            size_t GrainSize)
```

13.25.2.8 benchmark_with_field() [2/2]

```
template<class Field, class arg>
void benchmark_with_field (
            const Givaro::Integer & q,
            int p,
            size_t m,
            size_t k,
```

```
            int NBK,
            uint64_t bitsize,
            uint64_t seed,
            size_t iters,
            int t,
            arg & as,
            size_t GrainSize)
```

13.25.2.9 main()

```
int main (
            int argc,
            char ** argv)
```

13.26 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)

13.26.1 Macro Definition Documentation

13.26.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.26.2 Function Documentation

13.26.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.27 benchmark-fsyr2k.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"
```

Functions

- int main (int argc, char **argv)

13.27.1 Function Documentation

13.27.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.28 benchmark-fsyrk.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)

13.28.1 Macro Definition Documentation

13.28.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.28.2 Function Documentation

13.28.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.29 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)

Functions

- int main (int argc, char **argv)

13.29.1 Macro Definition Documentation

13.29.1.1 __FFPACK_FSYTRF_BC_CROUT

```
#define __FFPACK_FSYTRF_BC_CROUT
```

13.29.1.2 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.29.1.3 CUBE

```
#define CUBE(
            x)
```

Value:

```
((x)*(x)*(x))
```

13.29.2 Function Documentation

13.29.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.30 benchmark-ftrsm-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)

13.30.1 Macro Definition Documentation

13.30.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.30.2 Function Documentation

13.30.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.31 benchmark-ftrsm.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)

13.31.1 Macro Definition Documentation

13.31.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.31.2 Function Documentation

13.31.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.32 benchmark-ftrsv.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)

13.32.1 Macro Definition Documentation

13.32.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.32.2 Function Documentation

13.32.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.33 benchmark-ftrtri.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)

Functions

- int main (int argc, char **argv)

13.33.1 Macro Definition Documentation

13.33.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.33.1.2 CUBE

```
#define CUBE(
             x)
```

Value:

```
((x)*(x)*(x))
```

13.33.2 Function Documentation

13.33.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.34 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)

Functions

- int main (int argc, char **argv)

13.34.1 Macro Definition Documentation

13.34.1.1 CUBE

```
#define CUBE(
               x)
```

Value:

```
((x)*(x)*(x))
```

13.34.2 Function Documentation

13.34.2.1 main()

```
int main (
             int argc,
             char ** argv)
```

13.35 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)

13.35.1 Function Documentation

13.35.1.1 main()

```
int main (
             int argc,
             char ** argv)
```

13.36 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)

Functions

- int main (int argc, char **argv)

13.36.1 Macro Definition Documentation

13.36.1.1 CUBE

```
#define CUBE(
                x)
```

Value:

```
((x)*(x)*(x))
```

13.36.2 Function Documentation

13.36.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.37 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)

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)

13.37.1 Macro Definition Documentation

13.37.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.37.1.2 CUBE

```
#define CUBE(
            x)
```

Value:

```
((x)*(x)*(x))
```

13.37.2 Typedef Documentation

13.37.2.1 Field

```
typedef Givaro::ModularBalanced<double> Field
```

13.37.3 Function Documentation

13.37.3.1 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)
```

13.37.3.2 Rec_Initialize()

```
void Rec_Initialize (
            Field & F,
            Field::Element * C,
            size_t m,
            size_t n,
            size_t ldc)
```

13.37.3.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.38 benchmark-quasisep.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/test-utils.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 n, size_t m, size_t t, size_t r, size_t iter, uint64_t seed)
- int main (int argc, char **argv)

13.38.1 Macro Definition Documentation

13.38.1.1 __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET

```
#define __FFLASFFPACK_OPENBLAS_NT_ALREADY_SET 1
```

13.38.2 Function Documentation

13.38.2.1 run_with_field()

```
template<class Field>
void run_with_field (
            int q,
            size_t n,
            size_t m,
            size_t t,
            size_t r,
            size_t iter,
            uint64_t seed)
```

13.38.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.39 benchmark-storage-transpose.C File Reference

```
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/fflas/fflas_transpose.h"
```

Data Structures

- class Bench< Elt >
- class ModularBalanced< T >

Functions

- int main (int argc, char **argv)

13.39.1 Function Documentation

13.39.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.40 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)

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)

13.40.1 Macro Definition Documentation

13.40.1.1 CUBE

```
#define CUBE(
            x)
```

Value:

```
((x)*(x)*(x))
```

13.40.2 Function Documentation

13.40.2.1 launch_wino()

```
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)
```

13.40.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.41 mainpage.doxy File Reference

13.42 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.

13.42.1 Function Documentation

13.42.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

This example computes the determinant of a matrix over a defined finite field.
Outputs the determinant.

13.43 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.

13.43.1 Function Documentation

13.43.1.1 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.

13.44 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.

13.44.1 Function Documentation

13.44.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

This example computes the rank of a matrix over a defined finite field.
Outputs the rank.

13.45 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.

13.45.1 Function Documentation

13.45.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

This example solve the quare system defined by the input over a defined finite field.

13.46 checker_charpoly.inl File Reference

```
#include "fflas-ffpack/ffpack/ffpack.h"
```

Data Structures

- class CheckerImplem_charpoly< Field, Polynomial >
- class CheckerImplem_charpoly< Givaro::ZRing< Givaro::Integer >, Polynomial >

Namespaces

- namespace FFPACK

  *Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_checker_charpoly_INL

13.46.1 Macro Definition Documentation

13.46.1.1 __FFLASFFPACK_checker_charpoly_INL

```
#define __FFLASFFPACK_checker_charpoly_INL
```

13.47 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

13.47.1 Macro Definition Documentation

13.47.1.1 __FFLASFFPACK_checker_det_INL

```
#define __FFLASFFPACK_checker_det_INL
```

13.48 checker_empty.h File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
```

Data Structures

- struct Checker_Empty< Field >

Namespaces

- namespace FFLAS

13.49 checker_fgemm.inl File Reference

Data Structures

- class CheckerImplem_fgemm< Field >

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_checker_fgemm_INL

13.49.1 Macro Definition Documentation

13.49.1.1 __FFLASFFPACK_checker_fgemm_INL

```
#define __FFLASFFPACK_checker_fgemm_INL
```

13.50 checker_ftrsm.inl File Reference

Data Structures

- class CheckerImplem_ftrsm< Field >

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_checker_ftrsm_INL

13.50.1 Macro Definition Documentation

13.50.1.1 __FFLASFFPACK_checker_ftrsm_INL

```
#define __FFLASFFPACK_checker_ftrsm_INL
```

13.51 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

13.51.1 Macro Definition Documentation

13.51.1.1 __FFLASFFPACK_checker_invert_INL

```
#define __FFLASFFPACK_checker_invert_INL
```

13.52 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

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_checker_pluq_INL

13.52.1 Macro Definition Documentation

13.52.1.1 __FFLASFFPACK_checker_pluq_INL

```
#define __FFLASFFPACK_checker_pluq_INL
```

13.53 checkers.doxy File Reference

13.54 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>

13.55 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>

13.55.1 Macro Definition Documentation

13.55.1.1 FFLASFFPACK_checkers_fflas_inl_H

```
#define FFLASFFPACK_checkers_fflas_inl_H
```

13.56 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>

13.57 checkers_ffpack.inl File Reference

```
#include "checker_pluq.inl"
#include "checker_det.inl"
#include "checker_invert.inl"
#include "checker_charpoly.inl"
```

Namespaces

- namespace FFPACK

    ***F**inite **F**ield **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>

13.57.1 Macro Definition Documentation

13.57.1.1 FFLASFFPACK_checkers_ffpack_inl_H

```
#define FFLASFFPACK_checkers_ffpack_inl_H
```

13.58 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 ∗)
- void cblas_dsyrk (const enum CBLAS_ORDER Order, const enum CBLAS_UPLO Uplo, const enum CBLAS_TRANSPOSE Trans, const int N, const int K, const double alpha, const double ∗A, const int lda, const double beta, double ∗C, const int ldc)

13.58.1 Macro Definition Documentation

13.58.1.1 CBLAS_INT

```
#define CBLAS_INT int
```

13.58.1.2 CBLAS_ENUM_DEFINED_H

```
#define CBLAS_ENUM_DEFINED_H
```

13.58.1.3 CBLAS_EXTERNALS

```
#define CBLAS_EXTERNALS
```

13.58.1.4 blas_enum

```
#define blas_enum enum
```

13.58.2 Enumeration Type Documentation

13.58.2.1 CBLAS_ORDER

```
enum CBLAS_ORDER
```

Enumerator

CblasRowMajor	
CblasColMajor	

13.58.2.2 CBLAS_TRANSPOSE

```
enum CBLAS_TRANSPOSE
```

Enumerator

CblasNoTrans	
CblasTrans	
CblasConjTrans	
AtlasConj	

13.58.2.3 CBLAS_UPLO

```
enum CBLAS_UPLO
```

Enumerator

CblasUpper	
CblasLower	

13.58.2.4 CBLAS_DIAG

```
enum CBLAS_DIAG
```

Enumerator

CblasNonUnit	
CblasUnit	

13.58.2.5 CBLAS_SIDE

```
enum CBLAS_SIDE
```

Enumerator

CblasLeft	
CblasRight	

13.58.3 Function Documentation

13.58.3.1 daxpy_()

```
void daxpy_ (
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

13.58.3.2 saxpy_()

```
void saxpy_ (
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

13.58.3.3 ddot_()

```
double ddot_ (
            const int * ,
            const double * ,
            const int * ,
            const double * ,
            const int * )
```

13.58.3.4 sdot_()

```
float sdot_ (
            const int * ,
            const float * ,
            const int * ,
            const float * ,
            const int * )
```

13.58.3.5 dasum_()

```
double dasum_ (
            const int * ,
            const double * ,
            const int * )
```

13.58.3.6 idamax_()

```
int idamax_ (
            const int * ,
            const double * ,
            const int * )
```

13.58.3.7 dnrm2_()

```
double dnrm2_ (
            const int * ,
            const double * ,
            const int * )
```

13.58.3.8 dgemv_()

```
void dgemv_ (
            const char * ,
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            const double * ,
            const int * ,
            const double * ,
            double * ,
            const int * )
```

13.58.3.9 sgemv_()

```
void sgemv_ (
            const char * ,
            const int * ,
            const int * ,
```

```
            const float * ,
            const float * ,
            const int * ,
            const float * ,
            const int * ,
            const float * ,
            float * ,
            const int * )
```

13.58.3.10 dger_()

```
void dger_ (
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

13.58.3.11 sger_()

```
void sger_ (
            const int * ,
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

13.58.3.12 dcopy_()

```
void dcopy_ (
            const int * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

13.58.3.13 scopy_()

```
void scopy_ (
            const int * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

13.58.3.14 dscal_()

```
void dscal_ (
            const int * ,
            const double * ,
```

```
            double * ,
            const int * )
```

13.58.3.15 sscal_()

```
void sscal_ (
            const int * ,
            const float * ,
            float * ,
            const int * )
```

13.58.3.16 dtrsm_()

```
void dtrsm_ (
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

13.58.3.17 strsm_()

```
void strsm_ (
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

13.58.3.18 dtrmm_()

```
void dtrmm_ (
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const double * ,
            const double * ,
            const int * ,
            double * ,
            const int * )
```

13.58.3.19 strmm_()

```
void strmm_ (
```

```
            const char * ,
            const char * ,
            const char * ,
            const char * ,
            const int * ,
            const int * ,
            const float * ,
            const float * ,
            const int * ,
            float * ,
            const int * )
```

13.58.3.20 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 * )
```

13.58.3.21 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 * )
```

13.58.3.22 cblas_dsyrk()

```
void cblas_dsyrk (
            const enum CBLAS_ORDER Order,
            const enum CBLAS_UPLO Uplo,
            const enum CBLAS_TRANSPOSE Trans,
            const int N,
            const int K,
            const double alpha,
            const double * A,
            const int lda,
            const double beta,
```

```
double * C,
const int ldc)
```

13.59 config.h File Reference

Macros

- #define HAVE_BIG_ENDIAN 1
- #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_INTTYPES_H 1
- #define HAVE_LAPACK 1
- #define HAVE_LIMITS_H 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 2.5.0"
- #define PACKAGE_TARNAME "fflas-ffpack"
- #define PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
- #define PACKAGE_VERSION "2.5.0"
- #define SIZEOF_CHAR 1
- #define SIZEOF_INT 4
- #define SIZEOF_LONG 8
- #define SIZEOF_LONG_LONG 8
- #define SIZEOF_SHORT 2
- #define SIZEOF___INT64_T 8
- #define STDC_HEADERS 1
- #define USE_OPENMP 1
- #define VERSION "2.5.0"

13.59.1 Macro Definition Documentation

13.59.1.1 HAVE_BIG_ENDIAN

```
#define HAVE_BIG_ENDIAN 1
```

13.59.1.2 HAVE_BLAS

```
#define HAVE_BLAS 1
```

13.59.1.3 HAVE_CBLAS

```
#define HAVE_CBLAS 1
```

13.59.1.4 HAVE_CXX11

```
#define HAVE_CXX11 1
```

13.59.1.5 HAVE_DLFCN_H

```
#define HAVE_DLFCN_H 1
```

13.59.1.6 HAVE_FLOAT_H

```
#define HAVE_FLOAT_H 1
```

13.59.1.7 HAVE_INT128

```
#define HAVE_INT128 1
```

13.59.1.8 HAVE_INTTYPES_H

```
#define HAVE_INTTYPES_H 1
```

13.59.1.9 HAVE_LAPACK

```
#define HAVE_LAPACK 1
```

13.59.1.10 HAVE_LIMITS_H

```
#define HAVE_LIMITS_H 1
```

13.59.1.11 HAVE_PTHREAD_H

```
#define HAVE_PTHREAD_H 1
```

13.59.1.12 HAVE_STDDEF_H

```
#define HAVE_STDDEF_H 1
```

13.59.1.13 HAVE_STDINT_H

```
#define HAVE_STDINT_H 1
```

13.59.1.14 HAVE_STDIO_H

```
#define HAVE_STDIO_H 1
```

13.59.1.15 HAVE_STDLIB_H

```
#define HAVE_STDLIB_H 1
```

13.59.1.16 HAVE_STRINGS_H

```
#define HAVE_STRINGS_H 1
```

13.59.1.17 HAVE_STRING_H

```
#define HAVE_STRING_H 1
```

13.59.1.18 HAVE_SYS_STAT_H

```
#define HAVE_SYS_STAT_H 1
```

13.59.1.19 HAVE_SYS_TIME_H

```
#define HAVE_SYS_TIME_H 1
```

13.59.1.20 HAVE_SYS_TYPES_H

```
#define HAVE_SYS_TYPES_H 1
```

13.59.1.21 HAVE_UNISTD_H

```
#define HAVE_UNISTD_H 1
```

13.59.1.22 LT_OBJDIR

```
#define LT_OBJDIR ".libs/"
```

13.59.1.23 OPENBLAS_NUM_THREADS

```
#define OPENBLAS_NUM_THREADS 1
```

13.59.1.24 PACKAGE

```
#define PACKAGE "fflas-ffpack"
```

13.59.1.25 PACKAGE_BUGREPORT

```
#define PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"
```

13.59.1.26 PACKAGE_NAME

```
#define PACKAGE_NAME "FFLAS-FFPACK"
```

13.59.1.27 PACKAGE_STRING

```
#define PACKAGE_STRING "FFLAS-FFPACK 2.5.0"
```

13.59.1.28 PACKAGE_TARNAME

```
#define PACKAGE_TARNAME "fflas-ffpack"
```

13.59.1.29 PACKAGE_URL

```
#define PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

13.59.1.30 PACKAGE_VERSION

```
#define PACKAGE_VERSION "2.5.0"
```

13.59.1.31 SIZEOF_CHAR

```
#define SIZEOF_CHAR 1
```

13.59.1.32 SIZEOF_INT

```
#define SIZEOF_INT 4
```

13.59.1.33 SIZEOF_LONG

```
#define SIZEOF_LONG 8
```

13.59.1.34 SIZEOF_LONG_LONG

```
#define SIZEOF_LONG_LONG 8
```

13.59.1.35 SIZEOF_SHORT

```
#define SIZEOF_SHORT 2
```

13.59.1.36 SIZEOF___INT64_T

```
#define SIZEOF___INT64_T 8
```

13.59.1.37 STDC_HEADERS

```
#define STDC_HEADERS 1
```

13.59.1.38 USE_OPENMP

```
#define USE_OPENMP 1
```

13.59.1.39 VERSION

```
#define VERSION "2.5.0"
```

13.60 fflas-ffpack/config.h File Reference

Macros

- #define __FFLASFFPACK_HAVE_BIG_ENDIAN 1
- #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_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 2.5.0"
- #define __FFLASFFPACK_PACKAGE_TARNAME "fflas-ffpack"
- #define __FFLASFFPACK_PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
- #define __FFLASFFPACK_PACKAGE_VERSION "2.5.0"
- #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_T 8
- #define __FFLASFFPACK_STDC_HEADERS 1
- #define __FFLASFFPACK_USE_OPENMP 1
- #define __FFLASFFPACK_VERSION "2.5.0"

13.60.1 Macro Definition Documentation

13.60.1.1 __FFLASFFPACK_HAVE_BIG_ENDIAN

```
#define __FFLASFFPACK_HAVE_BIG_ENDIAN 1
```

13.60.1.2 __FFLASFFPACK_HAVE_BLAS

```
#define __FFLASFFPACK_HAVE_BLAS 1
```

13.60.1.3 __FFLASFFPACK_HAVE_CBLAS

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

13.60.1.4 __FFLASFFPACK_HAVE_CXX11

```
#define __FFLASFFPACK_HAVE_CXX11 1
```

13.60.1.5 __FFLASFFPACK_HAVE_DLFCN_H

```
#define __FFLASFFPACK_HAVE_DLFCN_H 1
```

13.60.1.6 __FFLASFFPACK_HAVE_FLOAT_H

```
#define __FFLASFFPACK_HAVE_FLOAT_H 1
```

13.60.1.7 __FFLASFFPACK_HAVE_INT128

```
#define __FFLASFFPACK_HAVE_INT128 1
```

13.60.1.8 __FFLASFFPACK_HAVE_INTTYPES_H

```
#define __FFLASFFPACK_HAVE_INTTYPES_H 1
```

13.60.1.9 __FFLASFFPACK_HAVE_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

13.60.1.10 __FFLASFFPACK_HAVE_LIMITS_H

```
#define __FFLASFFPACK_HAVE_LIMITS_H 1
```

13.60.1.11 __FFLASFFPACK_HAVE_PTHREAD_H

```
#define __FFLASFFPACK_HAVE_PTHREAD_H 1
```

13.60.1.12 __FFLASFFPACK_HAVE_STDDEF_H

`#define __FFLASFFPACK_HAVE_STDDEF_H 1`

13.60.1.13 __FFLASFFPACK_HAVE_STDINT_H

`#define __FFLASFFPACK_HAVE_STDINT_H 1`

13.60.1.14 __FFLASFFPACK_HAVE_STDIO_H

`#define __FFLASFFPACK_HAVE_STDIO_H 1`

13.60.1.15 __FFLASFFPACK_HAVE_STDLIB_H

`#define __FFLASFFPACK_HAVE_STDLIB_H 1`

13.60.1.16 __FFLASFFPACK_HAVE_STRINGS_H

`#define __FFLASFFPACK_HAVE_STRINGS_H 1`

13.60.1.17 __FFLASFFPACK_HAVE_STRING_H

`#define __FFLASFFPACK_HAVE_STRING_H 1`

13.60.1.18 __FFLASFFPACK_HAVE_SYS_STAT_H

`#define __FFLASFFPACK_HAVE_SYS_STAT_H 1`

13.60.1.19 __FFLASFFPACK_HAVE_SYS_TIME_H

`#define __FFLASFFPACK_HAVE_SYS_TIME_H 1`

13.60.1.20 __FFLASFFPACK_HAVE_SYS_TYPES_H

`#define __FFLASFFPACK_HAVE_SYS_TYPES_H 1`

13.60.1.21 __FFLASFFPACK_HAVE_UNISTD_H

`#define __FFLASFFPACK_HAVE_UNISTD_H 1`

13.60.1.22 __FFLASFFPACK_LT_OBJDIR

`#define __FFLASFFPACK_LT_OBJDIR ".libs/"`

13.60.1.23 __FFLASFFPACK_OPENBLAS_NUM_THREADS

`#define __FFLASFFPACK_OPENBLAS_NUM_THREADS 1`

13.60.1.24 __FFLASFFPACK_PACKAGE

`#define __FFLASFFPACK_PACKAGE "fflas-ffpack"`

13.60.1.25 __FFLASFFPACK_PACKAGE_BUGREPORT

`#define __FFLASFFPACK_PACKAGE_BUGREPORT "ffpack-devel@googlegroups.com"`

13.60.1.26 __FFLASFFPACK_PACKAGE_NAME

`#define __FFLASFFPACK_PACKAGE_NAME "FFLAS-FFPACK"`

13.60.1.27 __FFLASFFPACK_PACKAGE_STRING

```
#define __FFLASFFPACK_PACKAGE_STRING "FFLAS-FFPACK 2.5.0"
```

13.60.1.28 __FFLASFFPACK_PACKAGE_TARNAME

```
#define __FFLASFFPACK_PACKAGE_TARNAME "fflas-ffpack"
```

13.60.1.29 __FFLASFFPACK_PACKAGE_URL

```
#define __FFLASFFPACK_PACKAGE_URL "https://github.com/linbox-team/fflas-ffpack"
```

13.60.1.30 __FFLASFFPACK_PACKAGE_VERSION

```
#define __FFLASFFPACK_PACKAGE_VERSION "2.5.0"
```

13.60.1.31 __FFLASFFPACK_SIZEOF_CHAR

```
#define __FFLASFFPACK_SIZEOF_CHAR 1
```

13.60.1.32 __FFLASFFPACK_SIZEOF_INT

```
#define __FFLASFFPACK_SIZEOF_INT 4
```

13.60.1.33 __FFLASFFPACK_SIZEOF_LONG

```
#define __FFLASFFPACK_SIZEOF_LONG 8
```

13.60.1.34 __FFLASFFPACK_SIZEOF_LONG_LONG

```
#define __FFLASFFPACK_SIZEOF_LONG_LONG 8
```

13.60.1.35 __FFLASFFPACK_SIZEOF_SHORT

```
#define __FFLASFFPACK_SIZEOF_SHORT 2
```

13.60.1.36 __FFLASFFPACK_SIZEOF___INT64_T

```
#define __FFLASFFPACK_SIZEOF___INT64_T 8
```

13.60.1.37 __FFLASFFPACK_STDC_HEADERS

```
#define __FFLASFFPACK_STDC_HEADERS 1
```

13.60.1.38 __FFLASFFPACK_USE_OPENMP

```
#define __FFLASFFPACK_USE_OPENMP 1
```

13.60.1.39 __FFLASFFPACK_VERSION

```
#define __FFLASFFPACK_VERSION "2.5.0"
```

13.61 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__)

13.61.1 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.

13.61.2 Macro Definition Documentation

13.61.2.1 GCC_VERSION

```
#define GCC_VERSION (__GNUC__ * 10000 + __GNUC_MINOR__ * 100 + __GNUC_PATCHLEVEL__)
```

13.62 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

13.62.1 Macro Definition Documentation

13.62.1.1 __FFLASFFPACK_WINOTHRESHOLD

```
#define __FFLASFFPACK_WINOTHRESHOLD 1000
```

13.62.1.2 __FFLASFFPACK_WINOTHRESHOLD_FLT

```
#define __FFLASFFPACK_WINOTHRESHOLD_FLT 2000
```

13.62.1.3 __FFLASFFPACK_WINOTHRESHOLD_BAL

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL 1000
```

13.62.1.4 __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT

```
#define __FFLASFFPACK_WINOTHRESHOLD_BAL_FLT 2000
```

13.62.1.5 __FFLASFFPACK_PLUQ_THRESHOLD

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 256
```

13.62.1.6 __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD

```
#define __FFLASFFPACK_CHARPOLY_LUKrylov_ArithProg_THRESHOLD 1000
```

13.62.1.7 __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD

```
#define __FFLASFFPACK_CHARPOLY_Danilevskii_LUKrylov_THRESHOLD 16
```

13.62.1.8 __FFLASFFPACK_ARITHPROG_THRESHOLD

```
#define __FFLASFFPACK_ARITHPROG_THRESHOLD 30
```

13.62.1.9 __FFLASFFPACK_FTRTRI_THRESHOLD

```
#define __FFLASFFPACK_FTRTRI_THRESHOLD 32
```

13.62.1.10 __FFLASFFPACK_FSYTRF_THRESHOLD

```
#define __FFLASFFPACK_FSYTRF_THRESHOLD 64
```

13.62.1.11 __FFLASFFPACK_FSYRK_THRESHOLD

```
#define __FFLASFFPACK_FSYRK_THRESHOLD 3000
```

13.63 fflas-ffpack-thresholds.h File Reference

13.64 fflas-ffpack.doxy File Reference

13.65 fflas-ffpack.h File Reference

Includes FFLAS and FFPACK.

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas/fflas.h"
#include "ffpack/ffpack.h"
```

13.65.1 Detailed Description

Includes FFLAS and FFPACK.

13.66 fflas.doxy File Reference

13.67 fflas.h File Reference

Finite **F**ield **L**inear **A**lgebra **S**ubroutines

```
#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_fsyrk_strassen.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.

13.67.1 Detailed Description

Finite **F**ield **L**inear **A**lgebra **S**ubroutines

Author

Clément Pernet.

13.67.2 Macro Definition Documentation

13.67.2.1 WINOTHRESHOLD

```
#define WINOTHRESHOLD __FFLASFFPACK_WINOTHRESHOLD
```

13.67.2.2 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.

13.68 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 double Computes nmax s.t.
- template<class Element>
  size_t TRSMBound (const Givaro::ModularBalanced< Element > &F)

  Specialization for balanced modular representation over double.

13.68.1 Macro Definition Documentation

13.68.1.1 __FFLASFFPACK_fflas_bounds_INL

```
#define __FFLASFFPACK_fflas_bounds_INL
```

13.68.1.2 FFLAS_INT_TYPE

```
#define FFLAS_INT_TYPE uint64_t
```

13.69 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 , FflasLeftTri = 123 , FflasRightTri = 124 }
  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)

13.70 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 faddin (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

  fadding for symmetric matrices
- 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.

13.71 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< 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)
- template< class Field, bool ADD>
  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)
- template< class Field, bool ADD>
  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)
- template< class Field, bool ADD>
  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)
- template< class Field, bool ADD>
  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)

13.71.1 Macro Definition Documentation

13.71.1.1 __FFLASFFPACK_fadd_INL

```
#define __FFLASFFPACK_fadd_INL
```

13.72 fflas_fassign.h File Reference

```
#include "fflas_fassign.inl"
```

13.73 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 : $A \leftarrow B$.

13.73.1 Macro Definition Documentation

13.73.1.1 __FFLASFFPACK_fassign_INL

```
#define __FFLASFFPACK_fassign_INL
```

13.74 fflas_faxpy.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::vectorised
- namespace FFLAS::vectorised::unswitch
- namespace FFLAS::details

Macros

- #define __FFLASFFPACK_faxpy_INL

Functions

- template<class Field>
std::enable_if<!FFLAS::support_simd_mod< typenameField::Element >::value &&FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, HelperMod< Field > &H)
- template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, const size_t incX, const size_t incY, HelperMod< Field > &H)
- template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n)
- template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type axpyp (const Field &F, const typename Field::Element a, typename Field::ConstElement_ptr X, typename Field::Element_ptr Y, const size_t n, const size_t incX, const size_t incY)
- template<class Field>
std::enable_if< FFLAS::support_fast_mod< typenameField::Element >::value, void >::type faxpy (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 faxpy (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 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 : $y \leftarrow \alpha \cdot x + y$.

13.74.1 Macro Definition Documentation

13.74.1.1 __FFLASFFPACK_faxpy_INL

```
#define __FFLASFFPACK_faxpy_INL
```

13.75 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 $x^T y$.

13.75.1 Macro Definition Documentation

13.75.1.1 __FFLASFFPACK_fdot_INL

```
#define __FFLASFFPACK_fdot_INL
```

13.76 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: **F**ield **GE**neral **M**atrix **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: Squares 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)

13.76.1 Macro Definition Documentation

13.76.1.1 __FFLASFFPACK_fgemm_INL

```
#define __FFLASFFPACK_fgemm_INL
```

13.77 fgemm_classical.inl File Reference

13.78 fgemm_classical_mp.inl File Reference

matrix multiplication with multiprecision input (either over Z or over Z/pZ)

```
#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"
```

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 >, 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>
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)

13.78.1 Detailed Description

matrix multiplication with multiprecision input (either over Z or over Z/pZ)

13.78.2 Macro Definition Documentation

13.78.2.1 __FFPACK_fgemm_classical_INL

```
#define __FFPACK_fgemm_classical_INL
```

13.79 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)

13.79.1 Macro Definition Documentation

13.79.1.1 __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL

```
#define __FFLASFFPACK_fflas_fflas_fgemm_winograd_INL
```

13.79.1.2 NEWWINO

```
#define NEWWINO
```

13.80 matmul.doxy File Reference

13.81 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, 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)

13.81.1 Detailed Description

Bini implementation.

13.81.2 Macro Definition Documentation

13.81.2.1 __FFLASFFPACK_fgemm_bini_INL

```
#define __FFLASFFPACK_fgemm_bini_INL
```

13.82 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, 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, FieldTrait > &WH)

13.82.1 Macro Definition Documentation

13.82.1.1 __FFLASFFPACK_fgemm_winograd_INL

```
#define __FFLASFFPACK_fgemm_winograd_INL
```

13.83 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)

13.83.1 Macro Definition Documentation

13.83.1.1 __FFLASFFPACK_fgemm_winograd_acc_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_INL
```

13.84 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, FieldTrait > &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, 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)

13.84.1 Macro Definition Documentation

13.84.1.1 __FFLASFFPACK_fgemm_winograd_acc_ip_INL

```
#define __FFLASFFPACK_fgemm_winograd_acc_ip_INL
```

13.85 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, 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)

13.85.1 Macro Definition Documentation

13.85.1.1 __FFLASFFPACK_fgemm_winograd_ip_INL

```
#define __FFLASFFPACK_fgemm_winograd_ip_INL
```

13.86 fflas_fgemv.inl File Reference

```
#include <givaro/zring.h>
```

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fgemv_INL

Functions

- 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 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, 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)

13.86.1 Macro Definition Documentation

13.86.1.1 __FFLASFFPACK_fgemv_INL

```
#define __FFLASFFPACK_fgemv_INL
```

13.87 fflas_fgemv_mp.inl File Reference

```
#include "fflas-ffpack/field/rns-integer-mod.h"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fgemv_mp_INL

Functions

- 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)

13.87.1 Macro Definition Documentation

13.87.1.1 __FFLASFFPACK_fgemv_mp_INL

```
#define __FFLASFFPACK_fgemv_mp_INL
```

13.88 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)

13.88.1 Macro Definition Documentation

13.88.1.1 __FFLASFFPACK_fger_INL

```
#define __FFLASFFPACK_fger_INL
```

13.89 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)

13.89.1 Macro Definition Documentation

13.89.1.1 __FFPACK_fger_mp_INL

```
#define __FFPACK_fger_mp_INL
```

13.90 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 $A \leftarrow A mod F$.
- template<class Field>
  void freduce (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::Element_ptr A, const size_t lda)

  freduce for square symmetric matrices
- 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)

13.91 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 BENCMARK 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, typename Field::Element_ptr A, const size_t incX, FC)

13.91.1 Macro Definition Documentation

13.91.1.1 __FFLASFFPACK_fflas_freduce_INL

```
#define __FFLASFFPACK_fflas_freduce_INL
```

13.91.1.2 FFLASFFPACK_COPY_REDUCE

```
#define FFLASFFPACK_COPY_REDUCE 32 /* TODO TO BENCMARK LATER */
```

13.92 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)

13.92.1 Macro Definition Documentation

13.92.1.1 __FFLASFFPACK_fflas_freduce_mp_INL

```
#define __FFLASFFPACK_fflas_freduce_mp_INL
```

13.93 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: **F**reivalds **GE**neral **M**atrix **M**ultiply **R**andom **C**heck.*

13.93.1 Macro Definition Documentation

13.93.1.1 __FFLASFFPACK_freivalds_INL

```
#define __FFLASFFPACK_freivalds_INL
```

13.94 fflas_fscal.h File Reference

```
#include "fflas_fscal.inl"
```

13.95 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< 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)
- 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, const size_t &incY, 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)
- 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 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, const size_t &incY)
- 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< class Field >
  void fscalin (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)

fscalin $x \leftarrow \alpha \cdot x$.

- 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 fscalin (const Givaro::DoubleDomain &, const size_t N, const Givaro::DoubleDomain::Element a, Givaro::DoubleDomain::Element_ptr y, const size_t incy)
- template<> void fscalin (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 fscalin (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)

  fscalin $A \leftarrow a \cdot A$.
- 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 $B \leftarrow a \cdot A$.

13.95.1 Macro Definition Documentation

13.95.1.1 __FFLASFFPACK_fscal_INL

```
#define __FFLASFFPACK_fscal_INL
```

13.96 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 fscalin (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 fscalin (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 fscalin (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 fscalin (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)

13.96.1 Macro Definition Documentation

13.96.1.1 __FFLASFFPACK_fscal_mp_INL

```
#define __FFLASFFPACK_fscal_mp_INL
```

13.97 fflas_fsyr2k.inl File Reference

```
#include <fflas-ffpack/utils/fflas_io.h>
```

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

13.97.1 Macro Definition Documentation

13.97.1.1 __FFLASFFPACK_fflas_fsyr2k_INL

```
#define __FFLASFFPACK_fflas_fsyr2k_INL
```

13.98 fflas_fsyrk.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fflas_fsyrk_INL

Functions

- template<class NewField, class Field, class FieldMode>
  Field::Element_ptr fsyrk_convert (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, MMHelper< Field, MMHelperAlgo::Classic, FieldMode > &H)
- 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::ConstElement_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, const ParSeqHelper::Sequential seq)
- 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, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- 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, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::ConvertTo< ElementCategories::MachineFloatTag >, ParSeqHelper::Sequential > &H)
- template<class Field, class AlgoT, class ParSeqTrait>
  void ScalAndReduce (const Field &F, const FFLAS_UPLO UpLo, 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 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, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DelayedTag > &H)
- 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, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::LazyTag > &H)
- template<class Field, typename Mode>
  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, MMHelper< Field, MMHelperAlgo::DivideAndConquer, Mode > &H)
- 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, MMHelper< Field, MMHelperAlgo::Classic, ModeCategories::DefaultBoundedTag > &H)
- Givaro::FloatDomain::Element_ptr fsyrk (const Givaro::FloatDomain &F, const FFLAS_UPLO Up↩ Lo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const Givaro::FloatDomain::↩ Element alpha, Givaro::FloatDomain::ConstElement_ptr A, const size_t lda, const Givaro::FloatDomain↩ ::Element beta, Givaro::FloatDomain::Element_ptr C, const size_t ldc, MMHelper< Givaro::FloatDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)

- Givaro::DoubleDomain::Element_ptr fsyrk (const Givaro::DoubleDomain &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const Givaro::DoubleDomain::Element alpha, Givaro::DoubleDomain::ConstElement_ptr A, const size_t lda, const Givaro::DoubleDomain::↩ Element beta, Givaro::DoubleDomain::Element_ptr C, const size_t ldc, MMHelper< Givaro::DoubleDomain, MMHelperAlgo::Classic, ModeCategories::DefaultTag > &H)
- 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

13.98.1 Macro Definition Documentation

13.98.1.1 __FFLASFFPACK_fflas_fsyrk_INL

```
#define __FFLASFFPACK_fflas_fsyrk_INL
```

13.99 fflas_fsyrk_strassen.inl File Reference

```
#include <givaro/givintsqrootmod.h>
```

Namespaces

- namespace FFLAS
- namespace FFLAS::Protected

Macros

- #define __FFLASFFPACK_fflas_fsyrk_strassen_INL

Functions

- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, const Element &x, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreScalReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)

- template<class Field, class Element, class AlgoT, class ParSeqTrait>
  bool NeedPreAxpyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeCategories::LazyTag, ParSeqTrait > &WH)
- template<class Field, class Element, class AlgoT, class ModeT, class ParSeqTrait>
  bool NeedPreAxpyReduction (Element &Outmin, Element &Outmax, Element &Op1min, Element &Op1max, Element &Op2min, Element &Op2max, const Element &x, MMHelper< Field, AlgoT, ModeT, ParSeqTrait > &WH)
- template<class Field, class FieldTrait>
  void computeS1S2 (const Field &F, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element x, const typename Field::Element y, typename Field::Element_ptr A, const size_t lda, typename Field::Element_ptr S, const size_t lds, typename Field::Element_ptr T, const size_t ldt, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)
- 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, MMHelper< Field, MMHelperAlgo::Winograd, ModeCategories::DelayedTag, ParSeqHelper::Sequential > &H)
- template<class Field, class Mode>
  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, MMHelper< Field, MMHelperAlgo::Winograd, Mode > &H)
- template<class Field, class FieldTrait>
  Field::Element_ptr fsyrk_strassen (const Field &F, const FFLAS_UPLO uplo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element y1, const typename Field::Element y2, const typename Field::Element alpha, typename Field::Element_ptr A, const size_t lda, const typename Field::Element beta, typename Field::Element_ptr C, const size_t ldc, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &WH)

13.99.1 Macro Definition Documentation

13.99.1.1 __FFLASFFPACK_fflas_fsyrk_strassen_INL

```
#define __FFLASFFPACK_fflas_fsyrk_strassen_INL
```

13.100 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**iangular **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**iangular **M**atrix **M**ultiply with 3 operands Computes $C \leftarrow \alpha \mathrm{op}(A)B + betaC$ or $C \leftarrow \alpha B\mathrm{op}(A) + betaC$.*

13.100.1 Macro Definition Documentation

13.100.1.1 __FFLASFFPACK_ftrmm_INL

```
#define __FFLASFFPACK_ftrmm_INL
```

13.101 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)

13.101.1 Macro Definition Documentation

13.101.1.1 __FFLASFFPACK_ftrsm_INL

```
#define __FFLASFFPACK_ftrsm_INL
```

13.102 fflas_ftrsm_mp.inl File Reference

triangular system with matrix right hand side over multiprecision domain (either over Z or over Z/pZ)

```
#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_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)

13.102.1 Detailed Description

triangular system with matrix right hand side over multiprecision domain (either over Z or over Z/pZ)

13.102.2 Macro Definition Documentation

13.102.2.1 __FFPACK_ftrsm_mp_INL

```
#define __FFPACK_ftrsm_mp_INL
```

13.103 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 $X \leftarrow \mathrm{op}(A^{-1})X$

13.103.1 Macro Definition Documentation

13.103.1.1 __FFLASFFPACK_ftrsv_INL

```
#define __FFLASFFPACK_ftrsv_INL
```

13.104 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 DivideAndConquer
- struct Winograd
- struct WinogradPar
- struct Bini
- struct AlgoChooser< ModeT, ParSeq >
- struct AlgoChooser< ModeCategories::ConvertTo< ElementCategories::RNSElementTag >, ParSeq >
- struct MMHelper< Field, AlgoTrait, ModeTrait, ParSeqTrait >
- struct Recursive
- struct Iterative
- struct Hybrid
- struct TRSMHelper< RecIterTrait, 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 RecInt::rint< 6 > &k)
- template<> size_t min_types (const RecInt::rint< 7 > &k)
- template<> size_t min_types (const RecInt::rint< 8 > &k)
- template<> size_t min_types (const RecInt::rint< 9 > &k)
- template<> size_t min_types (const RecInt::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 (RecInt::rint< K > x)
- template<> bool unfit (RecInt::rint< 6 > x)

13.104.1 Macro Definition Documentation

13.104.1.1 __FFLASFFPACK_fflas_fflas_mmhelper_INL

```
#define __FFLASFFPACK_fflas_fflas_mmhelper_INL
```

13.105 igemm.doxy File Reference

13.106 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)

13.107 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)

13.107.1 Macro Definition Documentation

13.107.1.1 __FFLASFFPACK_fflas_igemm_igemm_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_INL
```

13.108 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)

13.109 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)

13.109.1 Macro Definition Documentation

13.109.1.1 __FFLASFFPACK_fflas_igemm_igemm_kernels_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_kernels_INL
```

13.110 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)

13.111 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)

13.111.1 Macro Definition Documentation

13.111.1.1 __FFLASFFPACK_fflas_igemm_igemm_tools_INL

```
#define __FFLASFFPACK_fflas_igemm_igemm_tools_INL
```

13.112 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 Initializes X *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 fscalin (const Field &F, const size_t n, const typename Field::Element alpha, typename Field::Element_ptr X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x$.
- 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)

13.112.1 Macro Definition Documentation

13.112.1.1 __FFLASFFPACK_fflas_fflas_level1_INL

```
#define __FFLASFFPACK_fflas_fflas_level1_INL
```

13.113 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>
void fzero (const Field &F, const FFLAS_UPLO shape, const FFLAS_DIAG diag, const size_t n, typename Field::Element_ptr A, const size_t lda)

    fzero : $A \leftarrow 0$ *for a triangular matrix.*
- 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 $A \leftarrow AmodF$.
- template<class Field>
void freduce (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::Element_ptr A, const size_t lda)

    freduce for square symmetric matrices
- 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, 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, 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 $\mathtt{A}$ *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 fscalin (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)

    fscalin $A \leftarrow a \cdot A$.
- 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 : $A \leftarrow B$ *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 : 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 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 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 faddin (const Field &F, const FFLAS_UPLO uplo, const size_t N, typename Field::ConstElement_ptr B, const size_t ldb, typename Field::Element_ptr C, const size_t ldc)

    fadding for symmetric matrices
- 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.
- 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 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 $X \leftarrow \mathrm{op}(A^{-1})X$
- 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: 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 $X \leftarrow \mathrm{op}(A)X$

13.113.1 Macro Definition Documentation

13.113.1.1 __FFLASFFPACK_fflas_fflas_level2_INL

```
#define __FFLASFFPACK_fflas_fflas_level2_INL
```

13.114 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 MatF2MatFl_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: **TR**iangular **S**ystem solve with **M**atrix.*
- 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**iangular **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**iangular **M**atrix **M**ultiply with 3 operands Computes $C \leftarrow \alpha \mathrm{op}(A)B + betaC$ or $C \leftarrow \alpha B\mathrm{op}(A) + betaC$.*
- 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, typename FieldTrait>
  Field::Element_ptr fsyrk_strassen (const Field &F, const FFLAS_UPLO UpLo, const FFLAS_TRANSPOSE trans, const size_t N, const size_t K, const typename Field::Element y1, const typename Field::Element y2, 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, MMHelper< Field, MMHelperAlgo::Winograd, FieldTrait > &H)
- 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: **F**ield **GE**neral **M**atrix **M**ultiply.*
- 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.

13.114.1 Macro Definition Documentation

13.114.1.1 __FFLASFFPACK_fflas_fflas_level3_INL

```
#define __FFLASFFPACK_fflas_fflas_level3_INL
```

13.114.1.2 __FFLAS__TRSM_READONLY

```
#define __FFLAS__TRSM_READONLY
```

13.115 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::RNSElementTag > >::value, typenameField::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)

13.115.1 Macro Definition Documentation

13.115.1.1 __FFLASFFPACK_fflas_pfgemm_INL

```
#define __FFLASFFPACK_fflas_pfgemm_INL
```

13.115.1.2 __FFLASFFPACK_SEQPARTHRESHOLD

```
#define __FFLASFFPACK_SEQPARTHRESHOLD 220
```

13.115.1.3 __FFLASFFPACK_DIMKPENALTY

```
#define __FFLASFFPACK_DIMKPENALTY 1
```

13.116 fflas_pftrsm.inl File Reference

```
#include "fflas-ffpack/paladin/parallel.h"
```

Namespaces

- namespace FFLAS

Macros

- #define __FFLASFFPACK_fflas_pftrsm_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)

13.116.1 Macro Definition Documentation

13.116.1.1 __FFLASFFPACK_fflas_pftrsm_INL

```
#define __FFLASFFPACK_fflas_pftrsm_INL
```

13.116.1.2 PTRSM_HYBRID_THRESHOLD

```
#define PTRSM_HYBRID_THRESHOLD 256
```

13.117 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/utils/align-allocator.h"
#include <vector>
#include "givaro/givtypestring.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

13.117.1 Macro Definition Documentation

13.117.1.1 SIMD_INT

```
#define SIMD_INT 1
```

13.117.1.2 INLINE

```
#define INLINE inline
```

13.117.1.3 CONST

```
#define CONST
```

13.117.1.4 PURE

```
#define PURE
```

13.117.1.5 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);
     \
}
```

13.117.1.6 FLOAT_MOD

```
#define FLOAT_MOD(
              C,
              P,
              INVP,
              Q)
```

Value:

```
{
     \
    Q = mul(C, INVP);
     \
    Q = floor(Q);
     \
    C = fnmadd(C, Q, P);
     \
}
```

13.117.2 Typedef Documentation

13.117.2.1 Simd

```
template<class T>
```

```
using Simd = typename SimdChooser<T>::value
```

13.118 simd.doxy File Reference

13.119 simd128.inl File Reference

```
#include "simd128_float.inl"
#include "simd128_double.inl"
```

Data Structures

- struct Simd128i_base

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd128_INL

Typedefs

- template<class T>
  using Simd128

13.119.1 Macro Definition Documentation

13.119.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_INL
```

13.119.2 Typedef Documentation

13.119.2.1 Simd128

```
template<class T>
using Simd128
```

Initial value:

```
Simd128_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std::is_signed<T>::value, sizeof(T)>
```

13.120 simd128_double.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd128_impl< true, false, true, 8 >

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL

13.120.1 Macro Definition Documentation

13.120.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_double_INL
```

13.121 simd128_float.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd128_impl< true, false, true, 4 >

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL

13.121.1 Macro Definition Documentation

13.121.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_float_INL
```

13.122 simd128_int16.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

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

13.122.1 Macro Definition Documentation

13.122.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int16_INL
```

13.123 simd128_int32.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
```

```
#include <type_traits>
```

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

13.123.1 Macro Definition Documentation

13.123.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int32_INL
```

13.124 simd128_int64.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/utils/bit_manipulation.h"
#include <vector>
#include <type_traits>
```

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

13.124.1 Macro Definition Documentation

13.124.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd128_int64_INL
```

13.124.1.2 vect_t

```
#define vect_t Simd128_impl<true,true,true,8>::vect_t
```

13.125 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

13.125.1 Macro Definition Documentation

13.125.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_INL
```

13.125.2 Typedef Documentation

13.125.2.1 Simd256

```
template<class T>
using Simd256
```

Initial value:

```
Simd256_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std::is_signed<T>::value, sizeof(T)>
```

13.126 simd256_double.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd256_impl< true, false, true, 8 >
- union Simd256_impl< true, false, true, 8 >::Converter

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL

13.126.1 Macro Definition Documentation

13.126.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_double_INL
```

13.127 simd256_float.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd256_impl< true, false, true, 4 >

Macros

- #define __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL

13.127.1 Macro Definition Documentation

13.127.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_float_INL
```

13.128 simd256_int16.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

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

13.128.1 Macro Definition Documentation

13.128.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int16_INL
```

13.129 simd256_int32.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

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

13.129.1 Macro Definition Documentation

13.129.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int32_INL
```

13.130 simd256_int64.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/utils/bit_manipulation.h"
#include <vector>
#include <type_traits>
```

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

13.130.1 Macro Definition Documentation

13.130.1.1 __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL

```
#define __FFLASFFPACK_fflas_ffpack_utils_simd256_int64_INL
```

13.130.1.2 vect_t

```
#define vect_t Simd256_impl<true, true, true, 8>::vect_t
```

13.131 simd512.inl File Reference

```
#include "simd512_float.inl"
#include "simd512_double.inl"
#include "simd512_int64.inl"
```

Data Structures

- struct Simd512i_base

Macros

- #define __FFLASFFPACK_simd512_INL

Typedefs

- template<class T>
  using Simd512

13.131.1 Macro Definition Documentation

13.131.1.1 __FFLASFFPACK_simd512_INL

```
#define __FFLASFFPACK_simd512_INL
```

13.131.2 Typedef Documentation

13.131.2.1 Simd512

```
template<class T>
using Simd512
```

Initial value:

```
Simd512_impl<std::is_arithmetic<T>::value, std::is_integral<T>::value, std::is_signed<T>::value, sizeof(T)>
```

13.132 simd512_double.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd512_impl< true, false, true, 8 >

Macros

- #define __FFLASFFPACK_simd512_double_INL

13.132.1 Macro Definition Documentation

13.132.1.1 __FFLASFFPACK_simd512_double_INL

```
#define __FFLASFFPACK_simd512_double_INL
```

13.133 simd512_float.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

Data Structures

- struct Simd512_impl< true, false, true, 4 >

Macros

- #define __FFLASFFPACK_simd512_float_INL

13.133.1 Macro Definition Documentation

13.133.1.1 __FFLASFFPACK_simd512_float_INL

```
#define __FFLASFFPACK_simd512_float_INL
```

13.134 simd512_int32.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_simd/simd512_int64.inl"
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include <vector>
#include <type_traits>
```

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

13.134.1 Macro Definition Documentation

13.134.1.1 __FFLASFFPACK_simd512_int32_INL

```
#define __FFLASFFPACK_simd512_int32_INL
```

13.135 simd512_int64.inl File Reference

```
#include "givaro/givtypestring.h"
#include "fflas-ffpack/utils/align-allocator.h"
#include "fflas-ffpack/utils/bit_manipulation.h"
#include <vector>
#include <type_traits>
```

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

13.135.1 Macro Definition Documentation

13.135.1.1 _simd512_int64_INL

```
#define _simd512_int64_INL
```

13.135.1.2 vect_t

```
#define vect_t Simd512_impl<true, true, true, 8>::vect_t
```

13.136 simd_modular.inl File Reference

Data Structures

- class FieldSimd< _Field >

13.137 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)
- #define __FFLASFFPACK_CACHE_LINE_SIZE 64
- #define assume_aligned(pout, pin, v)
- #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::MachineFloa
  >::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::MachineFloa
>::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::MachineFloa
>::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>
  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)

13.137.1 Macro Definition Documentation

13.137.1.1 index_t

```
#define index_t uint32_t
```

13.137.1.2 ROUND_DOWN

```
#define ROUND_DOWN(
              x,
              s)
```

Value:

```
((x) & ~((s)-1))
```

13.137.1.3 __FFLASFFPACK_CACHE_LINE_SIZE

```
#define __FFLASFFPACK_CACHE_LINE_SIZE 64
```

13.137.1.4 assume_aligned

```
#define assume_aligned(
              pout,
              pin,
              v)
```

Value:

```
decltype(pin) pout = pin;
```

13.137.1.5 DENSE_THRESHOLD

```
#define DENSE_THRESHOLD 0.5
```

13.138 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::MachineFloa >::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 &&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 >::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<!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 &&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::MachineFloa >::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>
  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)

13.138.1 Macro Definition Documentation

13.138.1.1 __FFLASFFPACK_fflas_fflas_sparse_INL

```
#define __FFLASFFPACK_fflas_fflas_sparse_INL
```

13.139 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_spmm.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)

13.140 coo_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_coo_spmm_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)

13.140.1 Macro Definition Documentation

13.140.1.1 __FFLASFFPACK_fflas_sparse_coo_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmm_INL
```

13.141 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)

13.141.1 Macro Definition Documentation

13.141.1.1 __FFLASFFPACK_fflas_sparse_coo_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_spmv_INL
```

13.142 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)

13.142.1 Macro Definition Documentation

13.142.1.1 __FFLASFFPACK_fflas_sparse_coo_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_coo_utils_INL
```

13.143 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_spmm.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)

13.144 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 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)

13.144.1 Macro Definition Documentation

13.144.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmm_INL
```

13.145 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 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)

13.145.1 Macro Definition Documentation

13.145.1.1 __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_pspmv_INL
```

13.146 csr_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_spmm_INL

Functions

- 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)

13.146.1 Macro Definition Documentation

13.146.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmm_INL
```

13.147 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)

13.147.1 Macro Definition Documentation

13.147.1.1 __FFLASFFPACK_fflas_sparse_CSR_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_spmv_INL
```

13.148 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)

13.149 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_spmm.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)

13.150 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)

13.150.1 Macro Definition Documentation

13.150.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmm_INL
```

13.151 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)

13.151.1 Macro Definition Documentation

13.151.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_pspmv_INL
```

13.152 csr_hyb_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL

Functions

- 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)

13.152.1 Macro Definition Documentation

13.152.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmm_INL
```

13.153 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)

13.153.1 Macro Definition Documentation

13.153.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_spmv_INL
```

13.154 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)

13.154.1 Macro Definition Documentation

13.154.1.1 __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_CSR_HYB_utils_INL
```

13.155 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_spmm.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)

13.156 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)

13.156.1 Macro Definition Documentation

13.156.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmm_INL
```

13.157 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)

13.157.1 Macro Definition Documentation

13.157.1.1 __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_pspmv_INL
```

13.158 ell_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_ELL_spmm_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)

13.158.1 Macro Definition Documentation

13.158.1.1 __FFLASFFPACK_fflas_sparse_ELL_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_spmm_INL
```

13.159 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)

13.159.1 Macro Definition Documentation

13.159.1.1 __FFLASFFPACK_fflas_sparse_ELL_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_spmv_INL
```

13.160 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)

13.160.1 Macro Definition Documentation

13.160.1.1 __FFLASFFPACK_fflas_sparse_ELL_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_utils_INL
```

13.161 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)

13.162 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)

13.162.1 Macro Definition Documentation

13.162.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_pspmv_INL
```

13.163 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)

13.163.1 Macro Definition Documentation

13.163.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_spmv_INL
```

13.164 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)

13.164.1 Macro Definition Documentation

13.164.1.1 __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_ELL_simd_utils_INL
```

13.165 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_spmm.inl"
```

Namespaces

- namespace FFLAS

13.166 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, 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::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, typename Field::ConstElement_ptr x, int ldx, typename Field::Element_ptr y, int ldy, uint64_t kmax)

13.166.1 Macro Definition Documentation

13.166.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmm_INL
```

13.167 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)

13.167.1 Macro Definition Documentation

13.167.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_pspmv_INL
```

13.168 hyb_zo_spmm.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_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)

13.168.1 Macro Definition Documentation

13.168.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmm_INL
```

13.169 hyb_zo_spmv.inl File Reference

Namespaces

- namespace FFLAS
- namespace FFLAS::sparse_details_impl

Macros

- #define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_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)

13.169.1 Macro Definition Documentation

13.169.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_spmv_INL
```

13.170 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)

13.170.1 Macro Definition Documentation

13.170.1.1 __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_HYB_ZO_utils_INL
```

13.171 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 &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 > 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)

13.171.1 Macro Definition Documentation

13.171.1.1 DNS_BIN_VER

```
#define DNS_BIN_VER 0
```

13.171.1.2 mask_t

```
#define mask_t uint64_t
```

13.172 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

13.173 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)

13.173.1 Macro Definition Documentation

13.173.1.1 __FFLASFFPACK_fflas_sparse_sell_pspmv_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_pspmv_INL
```

13.174 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)

13.174.1 Macro Definition Documentation

13.174.1.1 __FFLASFFPACK_fflas_sparse_sell_spmv_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_spmv_INL
```

13.175 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)

13.175.1 Macro Definition Documentation

13.175.1.1 __FFLASFFPACK_fflas_sparse_sell_utils_INL

```
#define __FFLASFFPACK_fflas_sparse_sell_utils_INL
```

13.176 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

13.177 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)

13.178 fflas_transpose.h File Reference

transpose the storage of the matrix (switch between row and col major mode)

```
#include "fflas-ffpack/utils/debug.h"
#include "fflas-ffpack/fflas/fflas.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include "fflas-ffpack/fflas/fflas_simd.h"
```

Data Structures

- struct BlockTransposeSIMD< Field, Simd, >

Namespaces

- namespace FFLAS
- namespace FFLAS::_ftranspose_impl

Macros

- #define FFLAS_TRANSPOSE_BLOCKSIZE 32
- #define LD(i)
- #define ST(i)

Functions

- template<size_t bs, typename Field, typename BTSimd>
  void not_inplace (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)
- template<size_t bs, typename Field, typename BTSimd>
  void square_inplace (const Field &F, const BTSimd &BTS, const size_t m, typename Field::Element_ptr A, const size_t lda)
- template<size_t bs, typename Field, typename BTSimd>
  void nonsquare_inplace_v1 (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::Element_ptr A)
- template<size_t bs, typename Field, typename BTSimd>
  void nonsquare_inplace_v2 (const Field &F, const BTSimd &BTS, const size_t m, const size_t n, typename Field::Element_ptr A)
- template<typename Field, typename Simd = Simd<typename Field::Element>, size_t bs = FFLAS_TRANSPOSE_BLOCKSIZE, typename std::enable_if< Simd::template is_same_element< Field >::value >::type * = nullptr, typename std::enable_if< bs > = 1 && bs % Simd::vect_size == 0, ::type * = nullptr>
  Field::Element_ptr ftranspose (const Field &F, const size_t m, const size_t n, typename Field::ConstElement_ptr A, const size_t lda, typename Field::Element_ptr B, const size_t ldb)

13.178.1 Detailed Description

transpose the storage of the matrix (switch between row and col major mode)

13.178.2 Macro Definition Documentation

13.178.2.1 FFLAS_TRANSPOSE_BLOCKSIZE

```
#define FFLAS_TRANSPOSE_BLOCKSIZE 32
```

13.178.2.2 LD

```
#define LD(
            i)
```

Value:

```
R##i=Simd::loadu(A+lda*i)
```

13.178.2.3 ST

```
#define ST(
            i)
```

Value:

```
Simd::storeu(B+ldb*i,R##i)
```

13.179 ffpack.doxy File Reference

13.180 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_bruhatgen.inl"
#include "ffpack.inl"
```

Data Structures

- class CharpolyFailed

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **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 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>
  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 A X = B or X A = 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 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 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_THRESHOL
- 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_
- 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 LuTag=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 LuTag=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 lda, size_t *P, size_t *Qt, const bool transform=false, const FFPACK_LU_TAG Lu↩Tag=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>
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>
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>
std::list< Polynomial > & KellerGehrig (const Field &F, std::list< Polynomial > &charp, const size_t N, typename 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, 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 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, typename 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 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 & 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 $(v, Av, ..., A^N v)$.
- 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)
- 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 L X = B or X L = 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*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*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\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\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\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\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\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\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\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\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>
  size_t LTBruhatGen (const Field &Fi, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)

  LTBruhatGen Suppose A is Left Triangular Matrix This procedure computes the Bruhat Representation of A and return the rank of A.
- template<class Field>
  void getLTBruhatGen (const Field &Fi, const size_t N, const size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr R, const size_t ldr)

  GetLTBruhatGen This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.
- template<class Field>
  void getLTBruhatGen (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, 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)

  GetLTBruhatGen This procedure computes the matrix L or U f the Bruhat Representation Suppose that A is the bruhat representation of a matrix.
- size_t LTQSorder (const size_t N, const size_t r, const size_t *P, const size_t *Q)

  LTQSorder This procedure computes the order of quasiseparability of a matrix.
- template<class Field>
  size_t CompressToBlockBiDiagonal (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size_t s, size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t *K, size_t *M, size_t *T)

  CompressToBlockBiDiagonal This procedure compress a compact representation of a row echelon form or column echelon form.
- template<class Field>
  void ExpandBlockBiDiagonalToBruhat (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size_t s, size_t r, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t NbBlocks, size_t *K, size_t *M, size_t *T)

  ExpandBlockBiDiagonal This procedure expand a compact representation of a row echelon form or column echelon form.
- void Bruhat2EchelonPermutation (size_t N, size_t R, const size_t *P, const size_t *Q, size_t *M)

  Bruhat2EchelonPermutation (N,R,P,Q) Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Rpresentation.
- size_t * TInverter (size_t *T, size_t r)

- template<class Field>
  void ComputeRPermutation (const Field &Fi, size_t N, size_t r, const size_t *P, const size_t *Q, size_t *R, size_t *MU, size_t *ML)
- template<class Field>
  void productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, const size_t *P, const size_t *Q, const typename Field::Element_ptr Xu, size_t ldu, size_t NbBlocksU, size_t *Ku, size_t *Tu, size_t *MU, const typename Field::Element_ptr Xl, size_t ldl, size_t NbBlocksL, size_t *Kl, size_t *Tl, size_t *ML, typename Field::Element_ptr B, size_t t, size_t ldb, typename Field::Element_ptr C, size_t ldc)

  productBruhatxTS Comput the product between the CRE compact representation of a matrix A and B a tall matrix
- 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.

13.180.1 Detailed Description

Set of elimination based routines for dense linear algebra.
Matrices are supposed over finite prime field of characteristic less than 2^26.

13.180.2 Macro Definition Documentation

13.180.2.1 __FFLASFFPACK_FTRSTR_THRESHOLD

```
#define __FFLASFFPACK_FTRSTR_THRESHOLD 64
```

13.180.2.2 __FFLASFFPACK_FTRSSYR2K_THRESHOLD

```
#define __FFLASFFPACK_FTRSSYR2K_THRESHOLD 64
```

13.181 ffpack.inl File Reference

Namespaces

- namespace FFPACK

  F*inite* ***F****ield* ***PACK*** *Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_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 L X = B or X L = 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)

13.181.1 Macro Definition Documentation

13.181.1.1 __FFLASFFPACK_ffpack_INL

```
#define __FFLASFFPACK_ffpack_INL
```

13.182 ffpack_bruhatgen.inl File Reference

Namespaces

- namespace FFPACK

  *Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_ffpack_bruhatgen_inl

Functions

- template<class Field>
  size_t LTBruhatGen (const Field &Fi, const FFLAS::FFLAS_DIAG diag, const size_t N, typename Field::Element_ptr A, const size_t lda, size_t *P, size_t *Q)

  LTBruhatGen Suppose A is Left Triangular Matrix This procedure computes the Bruhat Representation of A and return the rank of A.
- template<class Field>
  void getLTBruhatGen (const Field &Fi, const size_t N, const size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr R, const size_t ldr)

  GetLTBruhatGen This procedure Computes the Rank Revealing Matrix based on the Bruhta representation of a Matrix.
- template<class Field>
  void getLTBruhatGen (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, const FFLAS::FFLAS_DIAG diag, 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)

  GetLTBruhatGen This procedure computes the matrix L or U f the Bruhat Representation Suppose that A is the bruhat representation of a matrix.
- size_t LTQSorder (const size_t N, const size_t r, const size_t *P, const size_t *Q)

  LTQSorder This procedure computes the order of quasiseparability of a matrix.
- template<class Field>
  size_t CompressToBlockBiDiagonal (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size_t s, size_t r, const size_t *P, const size_t *Q, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t *K, size_t *M, size_t *T)

  CompressToBlockBiDiagonal This procedure compress a compact representation of a row echelon form or column echelon form.
- template<class Field>
  void ExpandBlockBiDiagonalToBruhat (const Field &Fi, const FFLAS::FFLAS_UPLO Uplo, size_t N, size_t s, size_t r, typename Field::Element_ptr A, size_t lda, typename Field::Element_ptr X, size_t ldx, size_t NbBlocks, size_t *K, size_t *M, size_t *T)

  ExpandBlockBiDiagonal This procedure expand a compact representation of a row echelon form or column echelon form.
- void Bruhat2EchelonPermutation (size_t N, size_t R, const size_t *P, const size_t *Q, size_t *M)

  Bruhat2EchelonPermutation (N,R,P,Q) Compute M such that LM or MU is in echelon form where L or U are factors of the Bruhat Rpresentation.
- size_t * TInverter (const size_t *T, size_t r)
- template<class Field>
  void ComputeRPermutation (const Field &Fi, size_t N, size_t r, const size_t *P, const size_t *Q, size_t *R, const size_t *MU, const size_t *ML)
- template<class Field>
  Field::Element_ptr expandLCRE (const Field &Fi, size_t N, size_t s, size_t r, size_t *R, size_t i, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t *Ku, const size_t *Tuinv, typename Field::ConstElement_ptr Xl, size_t ldl, size_t NbBlocksL, const size_t *Kl, const size_t *Tlinv, typename Field::Element_ptr CRE, size_t ldcre)

  Expands an anti-diagonal block of a left triangular matrix from its compact Bruhat representation.
- template<class Field>
  void productBruhatxTS (const Field &Fi, size_t N, size_t s, size_t r, size_t t, const size_t *P, const size_t *Q, typename Field::ConstElement_ptr Xu, size_t ldu, size_t NbBlocksU, const size_t *Ku, const size_t *Tu, const size_t *MU, typename Field::ConstElement_ptr Xl, size_t ldl, size_t NbBlocksL, const size_t *Kl, const size_t *Tl, const size_t *ML, typename Field::Element_ptr B, size_t ldb, const typename Field::Element beta, typename Field::Element_ptr D, size_t ldd)

  Compute the product of a left-triangular quasi-separable matrix A, represented by a compact Bruhat generator, with a dense rectangular matrix B: $C \leftarrow A \times B + betaC$.

13.182.1 Macro Definition Documentation

13.182.1.1 __FFLASFFPACK_ffpack_bruhatgen_inl

```
#define __FFLASFFPACK_ffpack_bruhatgen_inl
```

13.183 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)

13.183.1 Macro Definition Documentation

13.183.1.1 __FFLASFFPACK_charpoly_INL

```
#define __FFLASFFPACK_charpoly_INL
```

13.184 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)

13.184.1 Macro Definition Documentation

13.184.1.1 __FFLASFFPACK_ffpack_charpoly_danilveski_INL

```
#define __FFLASFFPACK_ffpack_charpoly_danilveski_INL
```

13.185 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)

13.185.1 Macro Definition Documentation

13.185.1.1 __FFLASFFPACK_ffpack_charpoly_kgfast_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfast_INL
```

13.186 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)

13.186.1 Macro Definition Documentation

13.186.1.1 __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kgfastgeneralized_INL
```

13.187 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)

13.187.1 Macro Definition Documentation

13.187.1.1 __FFLASFFPACK_ffpack_charpoly_kglu_INL

```
#define __FFLASFFPACK_ffpack_charpoly_kglu_INL
```

13.188 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 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)

13.188.1 Macro Definition Documentation

13.188.1.1 __FFPACK_charpoly_mp_INL

```
#define __FFPACK_charpoly_mp_INL
```

13.189 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

  F*inite* ***F****ield* ***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 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)

13.189.1 Macro Definition Documentation

13.189.1.1 __FFPACK_det_mp_INL

```
#define __FFPACK_det_mp_INL
```

13.190 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\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\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\U of rank R obtained by RowEchelonForm or Column↩ EchelonForm.
- 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\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\U of rank R obtained by RowEchelon↩ Form 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 Lu↩ Tag=FfpackSlabRecursive)

  Extracts a matrix in echelon form from a compact storage A=L\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\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\U of rank R obtained by RowEchelonForm or ColumnEchelonForm.

13.190.1 Macro Definition Documentation

13.190.1.1 __FFLASFFPACK_ffpack_echelon_forms_INL

```
#define __FFLASFFPACK_ffpack_echelon_forms_INL
```

13.190.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 256
```

13.191 ffpack_fgesv.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **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.

13.191.1 Macro Definition Documentation

13.191.1.1 __FFLASFFPACK_ffpack_fgesv_INL

```
#define __FFLASFFPACK_ffpack_fgesv_INL
```

13.192 ffpack_fgetrs.inl File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **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 ldx, typename Field::ConstElement_ptr B, const size_t ldb, int *info)

  Solve the system A X = B or X A = B.

13.192.1 Macro Definition Documentation

13.192.1.1 __FFLASFFPACK_ffpack_fgetrs_INL

```
#define __FFLASFFPACK_ffpack_fgetrs_INL
```

13.193 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)

13.194 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_THRESHOL
- 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_

- 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)

13.194.1 Macro Definition Documentation

13.194.1.1 __FFLASFFPACK_ffpack_fsytrf_INL

```
#define __FFLASFFPACK_ffpack_fsytrf_INL
```

13.195 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.

13.195.1 Macro Definition Documentation

13.195.1.1 __FFLASFFPACK_ffpack_ftrssyr2k_INL

```
#define __FFLASFFPACK_ffpack_ftrssyr2k_INL
```

13.196 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.

13.196.1 Macro Definition Documentation

13.196.1.1 __FFLASFFPACK_ffpack_ftrstr_INL

```
#define __FFLASFFPACK_ffpack_ftrstr_INL
```

13.197 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)

13.197.1 Macro Definition Documentation

13.197.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.197.1.2 __FFLASFFPACK_ffpack_ftrtr_INL

```
#define __FFLASFFPACK_ffpack_ftrtr_INL
```

13.198 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.

13.198.1 Macro Definition Documentation

13.198.1.1 __FFLASFFPACK_ffpack_invert_INL

```
#define __FFLASFFPACK_ffpack_invert_INL
```

13.199 ffpack_krylovelim.inl File Reference

Macros

- #define __FFLASFFPACK_ffpack_krylovelim_INL

13.199.1 Macro Definition Documentation

13.199.1.1 __FFLASFFPACK_ffpack_krylovelim_INL

```
#define __FFLASFFPACK_ffpack_krylovelim_INL
```

13.200 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)

13.200.1 Macro Definition Documentation

13.200.1.1 __FFLASFFPACK_ffpack_ludivine_INL

```
#define __FFLASFFPACK_ffpack_ludivine_INL
```

13.201 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)

13.201.1 Macro Definition Documentation

13.201.1.1 __FFPACK_ludivine_mp_INL

```
#define __FFPACK_ludivine_mp_INL
```

13.202 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 $(v, Av, ..., A^N v)$.
- 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)

13.202.1 Macro Definition Documentation

13.202.1.1 __FFLASFFPACK_ffpack_minpoly_INL

```
#define __FFLASFFPACK_ffpack_minpoly_INL
```

13.203 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, 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 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 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.
- void composePermutationsLLM (size_t *MathP, const size_t *P1, const size_t *P2, const size_t R, const size_t N)

  Computes P1 x 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)

13.203.1 Macro Definition Documentation

13.203.1.1 __FFLASFFPACK_ffpack_permutation_INL

```
#define __FFLASFFPACK_ffpack_permutation_INL
```

13.203.1.2 FFLASFFPACK_PERM_BKSIZE

```
#define FFLASFFPACK_PERM_BKSIZE 32
```

13.204 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.

13.204.1 Macro Definition Documentation

13.204.1.1 __FFLASFFPACK_ffpack_pluq_INL

```
#define __FFLASFFPACK_ffpack_pluq_INL
```

13.204.1.2 CROUT

```
#define CROUT
```

13.205 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)

13.205.1 Macro Definition Documentation

13.205.1.1 __FFPACK_pluq_mp_INL

```
#define __FFPACK_pluq_mp_INL
```

13.206 ffpack_ppluq.inl File Reference

Namespaces

- namespace FFPACK
*Finite **F**ield **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_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, 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 > &PSHelper)
- 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)

13.206.1 Macro Definition Documentation

13.206.1.1 __FFLASFFPACK_ffpack_ppluq_INL

```
#define __FFLASFFPACK_ffpack_ppluq_INL
```

13.206.1.2 __FFLAS__TRSM_READONLY

```
#define __FFLAS__TRSM_READONLY
```

13.206.1.3 PBASECASE_K

```
#define PBASECASE_K 256
```

13.207 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∗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∗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.

13.207.1 Macro Definition Documentation

13.207.1.1 __FFLASFFPACK_ffpack_rank_profiles_INL

```
#define __FFLASFFPACK_ffpack_rank_profiles_INL
```

13.208 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::recInt.
- 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< RecInt::rint< K > >
- struct ElementTraits< RecInt::ruint< K > >
- struct ElementTraits< RecInt::rmint< K, MG > >
- struct ElementTraits< FFPACK::rns_double_elt >
- struct ModeTraits< Field >

  ModeTraits.
- struct ModeTraits< Givaro::Modular< Element, Compute > >
- struct ModeTraits< Givaro::Modular< int64_t, uint64_t > >
- 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< RecInt::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< RecInt::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 RecInt
- 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

Functions

- template<class Field, class enable = void>
  Field::Residu_t maxCardinality ()
- template<> uint64_t maxCardinality< Givaro::Modular< int64_t > > ()
- template<> uint32_t maxCardinality< Givaro::Modular< int32_t > > ()
- template<class Field>
  Field::Residu_t minCardinality ()

13.208.1 Detailed Description

Field Traits.

13.209 field.doxy File Reference

13.210 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)

13.210.1 Detailed Description

rns elt structure with double support

13.211 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

13.211.1 Macro Definition Documentation

13.211.1.1 __FFLASFFPACK_field_rns_double_recint_INL

```
#define __FFLASFFPACK_field_rns_double_recint_INL
```

13.212 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)

Functions

- template<> void fflas_delete (FFPACK::rns_double_elt_ptr A)
- template<> void fflas_delete (FFPACK::rns_double_elt_cstptr A)

13.212.1 Detailed Description

rns structure with double support

13.212.2 Macro Definition Documentation

13.212.2.1 ROUND_DOWN

```
#define ROUND_DOWN(
                x,
                s)
```

Value:

```
((x) & ~((s)-1))
```

13.213 rns-double.inl File Reference

```
#include "fflas-ffpack/fflas/fflas_freduce.h"
```

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Macros

- #define __FFLASFFPACK_field_rns_double_INL

13.213.1 Macro Definition Documentation

13.213.1.1 __FFLASFFPACK_field_rns_double_INL

```
#define __FFLASFFPACK_field_rns_double_INL
```

13.214 rns-integer-mod.h File Reference

representation of `Z/pZ` 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)

13.214.1 Detailed Description

representation of `Z/pZ` using RNS representation (note: fixed precision)

13.215 rns-integer.h File Reference

representation of `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)

13.215.1 Detailed Description

representation of $\mathbb{Z}$ using RNS representation (note: fixed precision)

13.216 rns.h File Reference

Namespaces

- namespace FFPACK

  *Finite Field **PACK** Set of elimination based routines for dense linear algebra.*

13.217 rns.inl File Reference

```
#include "rns-double.h"
#include "rns-integer.h"
#include "rns-integer-mod.h"
```

Macros

- #define __FFLASFFPACK_field_rns_INL

13.217.1 Macro Definition Documentation

13.217.1.1 __FFLASFFPACK_field_rns_INL

```
#define __FFLASFFPACK_field_rns_INL
```

13.218 interfaces.doxy File Reference

13.219 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 }

  Storage by row or col ?
- enum FFLAS_C_TRANSPOSE { FflasNoTrans = 111 , FflasTrans = 112 }

  Is matrix transposed ?
- enum FFLAS_C_UPLO { FflasUpper = 121 , FflasLower = 122 }

  Is triangular matrix's shape upper ?
- enum FFLAS_C_DIAG { FflasNonUnit = 131 , FflasUnit = 132 }

  Is the triangular matrix implicitly unit diagonal ?
- enum FFLAS_C_SIDE { 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 fscalin_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)
- 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)
- 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 ∗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 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 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)

13.219.1 Macro Definition Documentation

13.219.1.1 FFLAS_COMPILED

```
#define FFLAS_COMPILED
```

13.219.2 Enumeration Type Documentation

13.219.2.1 FFLAS_C_ORDER

```
enum FFLAS_C_ORDER
```

Storage by row or col ?

Enumerator

FflasRowMajor	row major
FflasColMajor	col major

13.219.2.2 FFLAS_C_TRANSPOSE

```
enum FFLAS_C_TRANSPOSE
```

Is matrix transposed ?

Enumerator

FflasNoTrans	Matrix is not transposed.
FflasTrans	Matrix is transposed.

13.219.2.3 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 $T_{i,j} = 0$)

13.219.2.4 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 ($T_{i,i} = 1$)

13.219.2.5 FFLAS_C_SIDE

enum FFLAS_C_SIDE
On what side ?

Enumerator

FflasLeft	Operator applied on the left.
FflasRight	Operator applied on the rigth.

13.219.2.6 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 precison BLAS
FflasGeneric	for any other domain, that can not be converted to floating point integers

13.219.3 Function Documentation

13.219.3.1 freducein_1_modular_double()

```
void freducein_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive)
```

13.219.3.2 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)
```

13.219.3.3 fnegin_1_modular_double()

```
void fnegin_1_modular_double (
            const double F,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive)
```

13.219.3.4 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)
```

13.219.3.5 fzero_1_modular_double()

```
void fzero_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive)
```

13.219.3.6 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)
```

13.219.3.7 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)
```

13.219.3.8 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)
```

13.219.3.9 fscalin_1_modular_double()

```
void fscalin_1_modular_double (
            const double p,
            const size_t n,
            const double alpha,
            double * X,
            const size_t incX,
            bool positive)
```

13.219.3.10 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)
```

13.219.3.11 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)
```

13.219.3.12 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)
```

13.219.3.13 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)
```

13.219.3.14 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)
```

13.219.3.15 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)
```

13.219.3.16 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)
```

13.219.3.17 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)
```

13.219.3.18 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)
```

13.219.3.19 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)
```

13.219.3.20 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)
```

13.219.3.21 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)
```

13.219.3.22 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)
```

13.219.3.23 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)
```

13.219.3.24 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)
```

13.219.3.25 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)
```

13.219.3.26 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)
```

13.219.3.27 fscalin_2_modular_double()

```
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)
```

13.219.3.28 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)
```

13.219.3.29 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)
```

13.219.3.30 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)
```

13.219.3.31 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)
```

13.219.3.32 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)
```

13.219.3.33 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)
```

13.219.3.34 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)
```

13.219.3.35 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)
```

13.219.3.36 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)
```

13.219.3.37 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)
```

13.219.3.38 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)
```

13.219.3.39 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)
```

13.219.3.40 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)
```

13.219.3.41 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)
```

13.220 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

13.220.1 Macro Definition Documentation

13.220.1.1 __FFLAS_L1_INST_C

```
#define __FFLAS_L1_INST_C
```

13.220.1.2 INST_OR_DECL

```
#define INST_OR_DECL
```

13.220.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.220.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.220.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.220.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.220.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.220.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.220.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.220.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.221 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

13.221.1 Macro Definition Documentation

13.221.1.1 INST_OR_DECL

```
#define INST_OR_DECL <>
```

13.221.1.2 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.221.1.3 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.221.1.4 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.221.1.5 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.221.1.6 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.221.1.7 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.221.1.8 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.221.1.9 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.222 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 fscalin (const FFLAS_FIELD< FFLAS_ELT > &F, const size_t n, const FFLAS_ELT alpha, FFLAS_ELT ∗X, const size_t incX)

  fscalin $x \leftarrow \alpha \cdot x.$
- 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)

  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)

  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)

13.223 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

13.223.1 Macro Definition Documentation

13.223.1.1 __FFLAS_L2_INST_C

```
#define __FFLAS_L2_INST_C
```

13.223.1.2 INST_OR_DECL

```
#define INST_OR_DECL
```

13.223.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.223.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.223.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.223.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.223.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.223.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.223.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.223.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.224 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

13.224.1 Macro Definition Documentation

13.224.1.1 INST_OR_DECL

```
#define INST_OR_DECL <>
```

13.224.1.2 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.224.1.3 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.224.1.4 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.224.1.5 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.224.1.6 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.224.1.7 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.224.1.8 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.224.1.9 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.225 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 fscalin (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)

  fscalin $A \leftarrow a \cdot A$.
- 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)

  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 : 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 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: TRiangular System solve with Vector Computes $X \leftarrow \mathrm{op}(A^{-1})X$

13.226 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

13.226.1 Macro Definition Documentation

13.226.1.1 __FFLAS_L3_INST_C

```
#define __FFLAS_L3_INST_C
```

13.226.1.2 INST_OR_DECL

```
#define INST_OR_DECL
```

13.226.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.226.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.226.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.226.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.226.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.226.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.226.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.226.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.227 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

13.227.1 Macro Definition Documentation

13.227.1.1 INST_OR_DECL

```
#define INST_OR_DECL <>
```

13.227.1.2 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.227.1.3 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.227.1.4 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.227.1.5 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.227.1.6 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.227.1.7 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.227.1.8 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.227.1.9 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.228 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.

13.228.1 Macro Definition Documentation

13.228.1.1 __FFLAS__TRSM_READONLY

```
#define __FFLAS__TRSM_READONLY
```

13.229 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 fscalin_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)

13.229.1 Detailed Description

C functions calls for level 1 FFLAS in fflas-c.h.

Author

Brice Boyer

See also

fflas/fflas_level1.inl

13.229.2 Function Documentation

13.229.2.1 freducein_1_modular_double()

```
void freducein_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive)
```

13.229.2.2 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)
```

13.229.2.3 fnegin_1_modular_double()

```
void fnegin_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive)
```

13.229.2.4 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)
```

13.229.2.5 fzero_1_modular_double()

```
void fzero_1_modular_double (
            const double p,
            const size_t n,
            double * X,
            const size_t incX,
            bool positive)
```

13.229.2.6 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)
```

13.229.2.7 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)
```

13.229.2.8 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)
```

13.229.2.9 fscalin_1_modular_double()

```
void fscalin_1_modular_double (
            const double p,
            const size_t n,
            const double alpha,
            double * X,
            const size_t incX,
            bool positive)
```

13.229.2.10 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)
```

13.229.2.11 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)
```

13.229.2.12 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)
```

13.229.2.13 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)
```

13.229.2.14 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)
```

13.229.2.15 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)
```

13.229.2.16 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)
```

13.229.2.17 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)
```

13.230 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 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 ∗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)

13.230.1 Detailed Description

C functions calls for level 2 FFLAS in fflas-c.h.

Author

Brice Boyer

See also

fflas/fflas_level2.inl

13.230.2 Function Documentation

13.230.2.1 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)
```

13.230.2.2 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)
```

13.230.2.3 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)
```

13.230.2.4 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)
```

13.230.2.5 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)
```

13.230.2.6 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)
```

13.230.2.7 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)
```

13.230.2.8 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)
```

13.230.2.9 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)
```

13.230.2.10 fscalin_2_modular_double()

```
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)
```

13.230.2.11 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)
```

13.230.2.12 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)
```

13.230.2.13 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)
```

13.230.2.14 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)
```

13.230.2.15 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)
```

13.230.2.16 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)
```

13.230.2.17 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)
```

13.230.2.18 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)
```

13.230.2.19 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)
```

13.230.2.20 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)
```

13.231 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)

13.231.1 Detailed Description

C functions calls for level 3 FFLAS in fflas-c.h.

Author

Brice Boyer

See also

fflas/fflas_level3.inl

13.231.2 Function Documentation

13.231.2.1 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)
```

13.231.2.2 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)
```

13.231.2.3 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)
```

13.231.2.4 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 betA,
            double * C,
            const size_t ldC,
            bool positive)
```

13.232 fflas_sparse.C File Reference

C functions calls for level 1.5 and 2.5 FFLAS in fflas-c.h.

13.232.1 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

13.233 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 ldx, 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 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)

- 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 ldx, 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 ldx, 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)

13.233.1 Detailed Description

C functions calls for FFPACK in ffpack-c.h.

Author

Brice Boyer

See also

ffpack/ffpack.h

13.233.2 Function Documentation

13.233.2.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
            size_t * MathP,
            const size_t * LapackP,
            const size_t N)
```

13.233.2.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
            size_t * LapackP,
            const size_t * MathP,
            const size_t N)
```

13.233.2.3 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)
```

13.233.2.4 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)
```

13.233.2.5 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)
```

13.233.2.6 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)
```

13.233.2.7 composePermutationsLLM()

```
void composePermutationsLLM (
            size_t * MathP,
            const size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.233.2.8 composePermutationsLLL()

```
void composePermutationsLLL (
            size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.233.2.9 composePermutationsMLM()

```
void composePermutationsMLM (
            size_t * MathP1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.233.2.10 cyclic_shift_mathPerm()

```
void cyclic_shift_mathPerm (
            size_t * P,
            const size_t s)
```

13.233.2.11 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)
```

13.233.2.12 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)
```

13.233.2.13 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)
```

13.233.2.14 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)
```

13.233.2.15 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)
```

13.233.2.16 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)
```

13.233.2.17 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)
```

13.233.2.18 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)
```

13.233.2.19 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)
```

13.233.2.20 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)
```

13.233.2.21 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)
```

13.233.2.22 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)
```

13.233.2.23 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)
```

13.233.2.24 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)
```

13.233.2.25 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)
```

13.233.2.26 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)
```

13.233.2.27 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)
```

13.233.2.28 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)
```

13.233.2.29 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)
```

13.233.2.30 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)
```

13.233.2.31 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)
```

13.233.2.32 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)
```

13.233.2.33 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)
```

13.233.2.34 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)
```

13.233.2.35 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)
```

13.233.2.36 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)
```

13.233.2.37 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)
```

13.233.2.38 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)
```

13.233.2.39 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)
```

13.233.2.40 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)
```

13.233.2.41 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)
```

13.233.2.42 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)
```

13.233.2.43 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)
```

13.233.2.44 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)
```

13.233.2.45 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)
```

13.233.2.46 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)
```

13.233.2.47 Invertin_modular_double()

```
double * Invertin_modular_double (
            const double p,
            const size_t M,
            double * A,
            const size_t lda,
            int * nullity,
            bool positive)
```

13.233.2.48 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)
```

13.233.2.49 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)
```

13.233.2.50 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)
```

13.233.2.51 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)
```

13.233.2.52 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)
```

13.233.2.53 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)
```

13.233.2.54 Det_modular_double()

```
double Det_modular_double (
            const double p,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive)
```

13.233.2.55 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)
```

13.233.2.56 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)
```

13.233.2.57 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)
```

13.233.2.58 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)
```

13.233.2.59 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)
```

13.233.2.60 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)
```

13.233.2.61 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)
```

13.233.2.62 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)
```

13.233.2.63 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)
```

13.233.2.64 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)
```

13.233.2.65 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)
```

13.233.2.66 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)
```

13.233.2.67 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)
```

13.233.2.68 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)
```

13.233.2.69 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)
```

13.233.2.70 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)
```

13.233.2.71 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)
```

13.233.2.72 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)
```

13.233.2.73 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)
```

13.233.2.74 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)
```

13.233.2.75 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)
```

13.233.2.76 PLUQtoEchelonPermutation()

```
void PLUQtoEchelonPermutation (
            const size_t N,
            const size_t R,
            const size_t * P,
            size_t * outPerm)
```

13.234 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 }
- enum FFLAS_C_TRANSPOSE { FflasNoTrans = 111 , FflasTrans = 112 }
- enum FFLAS_C_UPLO { FflasUpper = 121 , FflasLower = 122 }
- enum FFLAS_C_DIAG { FflasNonUnit = 131 , FflasUnit = 132 }
- enum FFLAS_C_SIDE { 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 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)
- 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 ftrtrm_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 ldx, 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 ldx, 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)

13.234.1 Macro Definition Documentation

13.234.1.1 FFPACK_COMPILED

```
#define FFPACK_COMPILED
```

13.234.2 Enumeration Type Documentation

13.234.2.1 FFLAS_C_ORDER

```
enum FFLAS_C_ORDER
```

Enumerator

FflasRowMajor	
FflasColMajor	

13.234.2.2 FFLAS_C_TRANSPOSE

```
enum FFLAS_C_TRANSPOSE
```

Enumerator

FflasNoTrans	
FflasTrans	

13.234.2.3 FFLAS_C_UPLO

```
enum FFLAS_C_UPLO
```

Enumerator

FflasUpper	
FflasLower	

13.234.2.4 FFLAS_C_DIAG

```
enum FFLAS_C_DIAG
```

Enumerator

FflasNonUnit	
FflasUnit	

13.234.2.5 FFLAS_C_SIDE

```
enum FFLAS_C_SIDE
```

Enumerator

FflasLeft	
FflasRight	

13.234.2.6 FFPACK_C_LU_TAG

```
enum FFPACK_C_LU_TAG
```

Enumerator

FfpackSlabRecursive	
FfpackTileRecursive	
FfpackSingular	

13.234.2.7 FFPACK_C_CHARPOLY_TAG

```
enum FFPACK_C_CHARPOLY_TAG
```

Enumerator

FfpackLUK	
FfpackKG	
FfpackHybrid	
FfpackKGFast	
FfpackDanilevski	
FfpackArithProg	
FfpackKGFastG	

13.234.2.8 FFPACK_C_MINPOLY_TAG

```
enum FFPACK_C_MINPOLY_TAG
```

Enumerator

FfpackDense	
FfpackKGF	

13.234.3 Function Documentation

13.234.3.1 LAPACKPerm2MathPerm()

```
void LAPACKPerm2MathPerm (
            size_t * MathP,
            const size_t * LapackP,
            const size_t N)
```

13.234.3.2 MathPerm2LAPACKPerm()

```
void MathPerm2LAPACKPerm (
            size_t * LapackP,
            const size_t * MathP,
            const size_t N)
```

13.234.3.3 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)
```

13.234.3.4 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)
```

13.234.3.5 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)
```

13.234.3.6 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)
```

13.234.3.7 composePermutationsLLM()

```
void composePermutationsLLM (
            size_t * MathP,
            const size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.234.3.8 composePermutationsLLL()

```
void composePermutationsLLL (
            size_t * P1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.234.3.9 composePermutationsMLM()

```
void composePermutationsMLM (
            size_t * MathP1,
            const size_t * P2,
            const size_t R,
            const size_t N)
```

13.234.3.10 cyclic_shift_mathPerm()

```
void cyclic_shift_mathPerm (
            size_t * P,
            const size_t s)
```

13.234.3.11 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)
```

13.234.3.12 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)
```

13.234.3.13 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)
```

13.234.3.14 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)
```

13.234.3.15 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)
```

13.234.3.16 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)
```

13.234.3.17 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)
```

13.234.3.18 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)
```

13.234.3.19 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)
```

13.234.3.20 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)
```

13.234.3.21 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)
```

13.234.3.22 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)
```

13.234.3.23 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)
```

13.234.3.24 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)
```

13.234.3.25 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)
```

13.234.3.26 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)
```

13.234.3.27 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)
```

13.234.3.28 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)
```

13.234.3.29 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)
```

13.234.3.30 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)
```

13.234.3.31 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)
```

13.234.3.32 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)
```

13.234.3.33 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)
```

13.234.3.34 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)
```

13.234.3.35 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)
```

13.234.3.36 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)
```

13.234.3.37 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)
```

13.234.3.38 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)
```

13.234.3.39 Invertin_modular_double()

```
double * Invertin_modular_double (
            const double p,
            const size_t M,
            double * A,
            const size_t lda,
            int * nullity,
            bool positive)
```

13.234.3.40 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)
```

13.234.3.41 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)
```

13.234.3.42 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)
```

13.234.3.43 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)
```

13.234.3.44 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)
```

13.234.3.45 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)
```

13.234.3.46 Det_modular_double()

```
double Det_modular_double (
            const double p,
            const size_t N,
            double * A,
            const size_t lda,
            bool positive)
```

13.234.3.47 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)
```

13.234.3.48 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)
```

13.234.3.49 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)
```

13.234.3.50 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)
```

13.234.3.51 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)
```

13.234.3.52 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)
```

13.234.3.53 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)
```

13.234.3.54 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)
```

13.234.3.55 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)
```

13.234.3.56 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)
```

13.234.3.57 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)
```

13.234.3.58 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)
```

13.234.3.59 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)
```

13.234.3.60 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)
```

13.234.3.61 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)
```

13.234.3.62 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)
```

13.234.3.63 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)
```

13.234.3.64 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)
```

13.234.3.65 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)
```

13.234.3.66 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)
```

13.234.3.67 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)
```

13.234.3.68 PLUQtoEchelonPermutation()

```
void PLUQtoEchelonPermutation (
            const size_t N,
            const size_t R,
```

```
const size_t * P,
size_t * outPerm)
```

13.235 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

13.235.1 Macro Definition Documentation

13.235.1.1 __FFPACK_INST_C

```
#define __FFPACK_INST_C
```

13.235.1.2 FFLAS_COMPILED

```
#define FFLAS_COMPILED
```

13.235.1.3 INST_OR_DECL

```
#define INST_OR_DECL
```

13.235.1.4 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.235.1.5 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.235.1.6 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.235.1.7 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.235.1.8 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.235.1.9 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.235.1.10 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.235.1.11 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.236 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

13.236.1 Macro Definition Documentation

13.236.1.1 FFLAS_COMPILED

```
#define FFLAS_COMPILED
```

13.236.1.2 INST_OR_DECL

```
#define INST_OR_DECL <>
```

13.236.1.3 FFLAS_FIELD [1/2]

```
#define FFLAS_FIELD Givaro::ModularBalanced
```

13.236.1.4 FFLAS_ELT [1/6]

```
#define FFLAS_ELT double
```

13.236.1.5 FFLAS_ELT [2/6]

```
#define FFLAS_ELT float
```

13.236.1.6 FFLAS_ELT [3/6]

```
#define FFLAS_ELT int64_t
```

13.236.1.7 FFLAS_FIELD [2/2]

```
#define FFLAS_FIELD Givaro::Modular
```

13.236.1.8 FFLAS_ELT [4/6]

```
#define FFLAS_ELT double
```

13.236.1.9 FFLAS_ELT [5/6]

```
#define FFLAS_ELT float
```

13.236.1.10 FFLAS_ELT [6/6]

```
#define FFLAS_ELT int64_t
```

13.237 ffpack_inst_implem.inl File Reference

Namespaces

- namespace FFPACK

    ***F**inite **F**ield **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 x 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 composePermutationsLLL (size_t ∗P1, const size_t ∗P2, const size_t R, const size_t N)

    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.
- 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<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)
- 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)

13.238 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)

13.238.1 Macro Definition Documentation

13.238.1.1 __FFLASFFPACK_fflas_blockcuts_INL

```
#define __FFLASFFPACK_fflas_blockcuts_INL
```

13.238.1.2 __FFLASFFPACK_MINBLOCKCUTS

```
#define __FFLASFFPACK_MINBLOCKCUTS ((size_t)256)
```

13.239 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)

13.240 kaapi_routines.inl File Reference

Macros

- #define __FFLASFFPACK_KAAPI_ROUTINES_INL

13.240.1 Macro Definition Documentation

13.240.1.1 __FFLASFFPACK_KAAPI_ROUTINES_INL

```
#define __FFLASFFPACK_KAAPI_ROUTINES_INL
```

13.241 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, I)
- #define WAIT
- #define CHECK_DEPENDENCIES
- #define BARRIER
- #define PAR_BLOCK
- #define SYNCH_GROUP(Args...)
- #define THREAD_INDEX 0
- #define NUM_THREADS 1
- #define SET_THREADS(num_threads)
- #define MAX_THREADS 1
- #define READ(Args...)
- #define WRITE(Args...)

- #define READWRITE(Args...)
- #define CONSTREFERENCE(...)
- #define VALUE(...)
- #define BEGIN_PARALLEL_MAIN(Args...)
- #define END_PARALLEL_MAIN(void)
- #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...)
- #define PARFOR2D(i, j, m, n, Helper, Args...)
- #define COMMA ,
- #define MODE(...)
- #define RETURNPARAM(f, P1, Args...)
- #define NUMARGS(...)
- #define PP_NARG_(...)
- #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, ...)
- #define PP_RSEQ_N()
- #define NOSPLIT()
- #define splitting_0()
- #define splitting_1(a)
- #define splitting_2(a, c)
- #define splitting_3(a, b, c)
- #define splitt(_1, _2, _3, NAME, ...)
- #define SPLITTER(...)

13.241.1 Macro Definition Documentation

13.241.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.241.1.2 index_t

```
#define index_t size_t
```

13.241.1.3 TASK

```
#define TASK(
            M,
            I)
```

Value:

```
{I;}
```

13.241.1.4 WAIT

```
#define WAIT
```

13.241.1.5 CHECK_DEPENDENCIES

```
#define CHECK_DEPENDENCIES
```

13.241.1.6 BARRIER

```
#define BARRIER
```

13.241.1.7 PAR_BLOCK

```
#define PAR_BLOCK
```

13.241.1.8 SYNCH_GROUP

```
#define SYNCH_GROUP(
              Args...)
```

Value:

```
{{Args};}
```

13.241.1.9 THREAD_INDEX

```
#define THREAD_INDEX 0
```

13.241.1.10 NUM_THREADS

```
#define NUM_THREADS 1
```

13.241.1.11 SET_THREADS

```
#define SET_THREADS(
              num_threads)
```

Value:

```
{}
```

13.241.1.12 MAX_THREADS

```
#define MAX_THREADS 1
```

13.241.1.13 READ

```
#define READ(
              Args...)
```

13.241.1.14 WRITE

```
#define WRITE(
              Args...)
```

13.241.1.15 READWRITE

```
#define READWRITE(
              Args...)
```

13.241.1.16 CONSTREFERENCE

```
#define CONSTREFERENCE(
              ...)
```

13.241.1.17 VALUE

```
#define VALUE(
              ...)
```

13.241.1.18 BEGIN_PARALLEL_MAIN

```
#define BEGIN_PARALLEL_MAIN(
              Args...)
```

Value:

```
int main(Args)  {
```

13.241.1.19 END_PARALLEL_MAIN

```
#define END_PARALLEL_MAIN(
              void)
```

Value:

```
return 0; }
```

13.241.1.20 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;} }
```

13.241.1.21 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; })
```

13.241.1.22 PARFORBLOCK1D

```
#define PARFORBLOCK1D(
              iter,
              m,
              Helper,
              Args...)
```

Value:

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter)      \
{ Args; }
```

13.241.1.23 PARFOR1D

```
#define PARFOR1D(
              iter,
              m,
              Helper,
              Args...)
```

Value:

```
for(std::remove_const<decltype(m)>::type iter=0; iter<m; ++iter)      \
{ Args; }
```

13.241.1.24 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; } }
```

13.241.1.25 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; })
```

13.241.1.26 PARFORBLOCK2D

```
#define PARFORBLOCK2D(
                iter,
                m,
                n,
                Helper,
                Args...)
```

Value:

```
FORBLOCK2D(iter, m, n, Helper, Args)
```

13.241.1.27 PARFOR2D

```
#define PARFOR2D(
                i,
                j,
                m,
                n,
                Helper,
                Args...)
```

Value:

```
FOR2D(i, j, m, n, Helper, Args)
```

13.241.1.28 COMMA

```
#define COMMA ,
```

13.241.1.29 MODE

```
#define MODE(
                ...)
```

Value:

```
__VA_ARGS__
```

13.241.1.30 RETURNPARAM

```
#define RETURNPARAM(
                f,
                P1,
                Args...)
```

Value:

```
P1=f(Args)
```

13.241.1.31 NUMARGS

```
#define NUMARGS(
                ...)
```

Value:

```
PP_NARG_(__VA_ARGS__,PP_RSEQ_N())
```

13.241.1.32 PP_NARG_

```
#define PP_NARG_(
                ...)
```

Value:

```
PP_ARG_N(__VA_ARGS__)
```

13.241.1.33 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,
...)

Value:

```
N
```

13.241.1.34 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
```

13.241.1.35 NOSPLIT

```
#define NOSPLIT()
```

Value:

```
FFLAS::ParSeqHelper::Sequential()
```

13.241.1.36 splitting_0

```
#define splitting_0()
```

Value:

```
FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::Threads>()
```

13.241.1.37 splitting_1

```
#define splitting_1(
              a)
```

Value:

```
FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,FFLAS::StrategyParameter::Threads>(a)
```

13.241.1.38 splitting_2

```
#define splitting_2(
              a,
              c)
```

Value:

```
FFLAS::ParSeqHelper::Parallel<FFLAS::CuttingStrategy::Block,c>(a)
```

13.241.1.39 splitting_3

```
#define splitting_3(
              a,
              b,
              c)
```

Value:

```
FFLAS::ParSeqHelper::Parallel<b,c>(a)
```

13.241.1.40 splitt

```
#define splitt(
              _1,
              _2,
              _3,
              NAME,
              ...)
```

Value:

```
NAME
```

13.241.1.41 SPLITTER

```
#define SPLITTER(
              ...)
```

Value:

```
splitt(__VA_ARGS__, splitting_3, splitting_2, splitting_1, splitting_0)(__VA_ARGS__)
```

13.242 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, 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)

13.243 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)

13.244 align-allocator.h File Reference

```
#include "fflas-ffpack/config.h"
```

13.245 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}
- char * getArgumentValue (int argc, char **argv, int i)

  Get the value of an argument and avoid core dump when no value was given after an argument.
- std::ostream & writeCommandString (std::ostream &os, Argument *args, const char *programName=nullptr)

  writes the values of all arguments, preceded by the programName

13.245.1 Macro Definition Documentation

13.245.1.1 TYPE_BOOL

```
#define TYPE_BOOL TYPE_NONE
```

13.245.1.2 END_OF_ARGUMENTS

```
#define END_OF_ARGUMENTS { '\0', "\0", "\0", TYPE_NONE, NULL }
```

13.245.1.3 type_integer

```
#define type_integer long int
```

13.245.2 Enumeration Type Documentation

13.245.2.1 ArgumentType

```
enum ArgumentType
```

Enumerator

TYPE_NONE	
TYPE_INT	
TYPE_UINT64	
TYPE_LONGLONG	
TYPE_INTEGER	
TYPE_DOUBLE	
TYPE_INTLIST	
TYPE_STR	

13.245.3 Function Documentation

13.245.3.1 printHelpMessage()

```
void printHelpMessage (
            const char * program,
            Argument * args,
            bool printDefaults = false)
```

13.245.3.2 findArgument()

```
Argument * findArgument (
            Argument * args,
            char c)
```

13.245.3.3 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

outlist	list once converted
instring	list to be converted

Returns

status message.

13.246 bit_manipulation.h File Reference

```
#include <givaro/udl.h>
#include "fflas-ffpack/fflas-ffpack-config.h"
```

Macros

- #define __has_builtin(x)

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)

13.246.1 Macro Definition Documentation

13.246.1.1 __has_builtin

```
#define __has_builtin(
                x)
```

Value:

```
0
```

13.246.2 Function Documentation

13.246.2.1 clz() [1/2]

```
int32_t clz (
            uint64_t val)  [inline]
```

13.246.2.2 clz() [2/2]

```
int32_t clz (
            uint32_t val)  [inline]
```

13.246.2.3 ctz() [1/2]

```
int32_t ctz (
            uint32_t val)  [inline]
```

13.246.2.4 ctz() [2/2]

```
int32_t ctz (
            uint64_t val)  [inline]
```

13.247 cast.h File Reference

Namespaces

- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Functions

- template<class T, class CT = const T>
  T fflas_const_cast (CT x)

13.248 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 precondtion failed.

Namespaces

- namespace FFPACK

  F*inite* ***F****ield* ***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)

13.248.1 Detailed Description

Various utilities for debugging.

Todo we should put vector printing elsewhere.

13.248.2 Macro Definition Documentation

13.248.2.1 FFLASFFPACK_check

```
#define FFLASFFPACK_check(
                check)
```

Value:

```
if (!(check)) {\
    FFPACK::failure()(__func__, __FILE__, __LINE__, #check); \
    throw std::runtime_error(#check); \
}
```

13.248.2.2 FFLASFFPACK_abort

```
#define FFLASFFPACK_abort(
                msg)
```

Value:

```
{\
    FFPACK::failure()(__func__, __FILE__, __LINE__, msg); \
    throw std::runtime_error(msg); \
}
```

13.249 fflas_intrinsic.h File Reference

13.250 fflas_io.h File Reference

```
#include <cstring>
#include <stdio.h>
#include <stdlib.h>
#include <fstream>
#include "fflas-ffpack/fflas/fflas.h"
```

```
#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.

13.251 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 ()

13.252 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 **F**ield **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 $\{0, \ldots, N-1\}$ 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 x 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 x N : a symmetric matrix with only R non-zeros, all equal to one, in a random rook placement.
- void RandomLTQSRankProfileMatrix (size_t n, size_t r, size_t t, size_t ∗rows, size_t ∗cols)
- 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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 x 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 RandIter>
  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)

  Random Matrix with prescribed det.
- template<class Field, class RandIter>
  Field::Element_ptr RandomLTQSMatrixWithRankandQSorder (Field &F, size_t n, size_t r, size_t t, typename Field::Element_ptr A, size_t lda, RandIter &G)

13.253 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< RecInt::ruint< K > >
- struct limits< RecInt::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)

13.253.1 Function Documentation

13.253.1.1 in_range() [1/3]

```
template<class T, class E>
std::enable_if< std::is_signed< T >::value==std::is_signed< E >::value, bool >::type in_↩
range (
            E e)
```

13.253.1.2 in_range() [2/3]

```
template<class T, class E>
std::enable_if<(std::is_signed< T >::value)&&!(std::is_signed< E >::value), bool >::type
in_range (
            E e)
```

13.253.1.3 in_range() [3/3]

```
template<class T, class E>
std::enable_if<!(std::is_signed< T >::value)&&(std::is_signed< E >::value), bool >::type
in_range (
            E e)
```

13.254 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)

13.254.1 Function Documentation

13.254.1.1 read_field()

```
template<class Field>
Field::Element_ptr read_field (
            const Field & F,
            const char * mat_file,
            size_t * tni,
            size_t * tnj)
```

13.254.1.2 write_field()

```
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)
```

13.255 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 <givaro/givtimer.h>
#include <random>
#include <functional>
```

Namespaces

- namespace FFLAS
- namespace FFPACK

  ***F**inite **F**ield **PACK** Set of elimination based routines for dense linear algebra.*

Functions

- uint64_t getSeed ()
- 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)

13.256 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

13.257 cblas.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

Macros

- #define __FFLASFFPACK_CONFIGURATION
- #define __FFLASFFPACK_HAVE_CBLAS 1

Functions

- int main ()

13.257.1 Macro Definition Documentation

13.257.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

13.257.1.2 __FFLASFFPACK_HAVE_CBLAS

```
#define __FFLASFFPACK_HAVE_CBLAS 1
```

13.257.2 Function Documentation

13.257.2.1 main()

```
int main (
            void )
```

13.258 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 ()

13.258.1 Macro Definition Documentation

13.258.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

13.258.1.2 __FFLASFFPACK_HAVE_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

13.258.1.3 __FFLASFFPACK_HAVE_CLAPACK

```
#define __FFLASFFPACK_HAVE_CLAPACK 1
```

13.258.2 Function Documentation

13.258.2.1 main()

```
int main (
            void )
```

13.259 cuda.C File Reference

```
#include <stdio.h>
#include <cuda_runtime.h>
#include <cusparse.h>
```

Functions

- int main ()

13.259.1 Function Documentation

13.259.1.1 main()

```
int main (
            void )
```

13.260 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 ()

13.260.1 Macro Definition Documentation

13.260.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

13.260.2 Function Documentation

13.260.2.1 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 * )
```

13.260.2.2 main()

```
int main (
            void )
```

13.261 lapack.C File Reference

```
#include "fflas-ffpack/config-blas.h"
```

Macros

- #define __FFLASFFPACK_CONFIGURATION
- #define __FFLASFFPACK_HAVE_LAPACK 1

Functions

- int main ()

13.261.1 Macro Definition Documentation

13.261.1.1 __FFLASFFPACK_CONFIGURATION

```
#define __FFLASFFPACK_CONFIGURATION
```

13.261.1.2 __FFLASFFPACK_HAVE_LAPACK

```
#define __FFLASFFPACK_HAVE_LAPACK 1
```

13.261.2 Function Documentation

13.261.2.1 main()

```
int main (
            void )
```

13.262 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 ()

13.262.1 Function Documentation

13.262.1.1 check1()

```
bool check1 ()
```

13.262.1.2 check2()

```
bool check2 ()
```

13.262.1.3 check3()

```
bool check3 ()
```

13.262.1.4 check4()

```
bool check4 ()
```

13.262.1.5 checkZeroDimCharpoly()

```
bool checkZeroDimCharpoly ()
```

13.262.1.6 checkZeroDimMinPoly()

```
bool checkZeroDimMinPoly ()
```

13.262.1.7 gf2ModularBalanced()

```
bool gf2ModularBalanced ()
```

13.262.1.8 main()

```
int main (
            void )
```

13.263 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)

13.263.1 Macro Definition Documentation

13.263.1.1 ENABLE_CHECKER_charpoly

```
#define ENABLE_CHECKER_charpoly 1
```

13.263.1.2 TIME_CHECKER_CHARPOLY

```
#define TIME_CHECKER_CHARPOLY 1
```

13.263.2 Function Documentation

13.263.2.1 printPolynomial()

```
template<class Field, class Polynomial>
void printPolynomial (
            const Field & F,
            Polynomial & v)
```

13.263.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.264 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)

13.264.1 Function Documentation

13.264.1.1 launch_test()

```
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)
```

13.264.1.2 run_with_field()

```
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)
```

13.264.1.3 main()

```
int main (
    int argc,
    char ** argv)
```

13.265 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)

13.265.1 Typedef Documentation

13.265.1.1 Field

```
typedef Givaro::Modular<double> Field
```

13.265.2 Function Documentation

13.265.2.1 printvect()

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

Bug does not belong here

13.265.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.266 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)

13.266.1 Macro Definition Documentation

13.266.1.1 ENABLE_CHECKER_Det

```
#define ENABLE_CHECKER_Det 1
```

13.266.1.2 TIME_CHECKER_Det

```
#define TIME_CHECKER_Det 1
```

13.266.2 Function Documentation

13.266.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.267 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)

13.267.1 Function Documentation

13.267.1.1 test_det()

```
template<class Field, class RandIter>
bool test_det (
            Field & F,
            size_t n,
            int iter,
            RandIter & G)
```

Todo test with stride

13.267.1.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.268 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_redcolechelon (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 Lu←
  Tag, 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)

13.268.1 Macro Definition Documentation

13.268.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.268.1.2 __FFLASFFPACK_GAUSSJORDAN_BASECASE

```
#define __FFLASFFPACK_GAUSSJORDAN_BASECASE 25
```

13.268.1.3 __FFLASFFPACK_PLUQ_THRESHOLD

```
#define __FFLASFFPACK_PLUQ_THRESHOLD 25
```

13.268.2 Function Documentation

13.268.2.1 test_colechelon()

```
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)
```

Todo check lda

13.268.2.2 test_rowechelon()

```
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)
```

Todo check lda

13.268.2.3 test_redcolechelon()

```
template<class Field, class RandIter>
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 lda

13.268.2.4 test_redrowechelon()

```
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)
```

Todo check lda

13.268.2.5 run_with_field()

```
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)
```

13.268.2.6 main()

```
int main (
            int argc,
            char ** argv)
```

13.269 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)

13.269.1 Function Documentation

13.269.1.1 test_fadd()

```
template<class Field>
bool test_fadd (
             const Field & F,
             size_t m,
             size_t k,
             size_t n,
             bool timing,
             uint64_t seed)
```

13.269.1.2 test_faddin()

```
template<class Field>
bool test_faddin (
             const Field & F,
             size_t m,
             size_t k,
             size_t n,
             bool timing,
             uint64_t seed)
```

13.269.1.3 test_fsub()

```
template<class Field>
bool test_fsub (
             const Field & F,
             size_t m,
             size_t k,
             size_t n,
             bool timing,
             uint64_t seed)
```

13.269.1.4 test_fsubin()

```
template<class Field>
bool test_fsubin (
             const Field & F,
             size_t m,
             size_t k,
             size_t n,
             bool timing,
             uint64_t seed)
```

13.269.1.5 main()

```
int main (
            int ac,
            char ** av)
```

13.270 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)

13.270.1 Macro Definition Documentation

13.270.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.270.2 Function Documentation

13.270.2.1 check_fdot()

```
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)
```

13.270.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
```

```
            Givaro::Integer q,
            size_t BS,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.270.2.3 run_with_Integer()

```
bool run_with_Integer (
            size_t BS,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.270.2.4 main()

```
int main (
            int argc,
            char ** argv)
```

13.271 test-fgemm-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

- 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)

13.271.1 Macro Definition Documentation

13.271.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.271.2 Function Documentation

13.271.2.1 launch_MM_dispatch()

```
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)
```

Bug test for ldX equal

Bug

Bug test for transpo

Bug

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

13.271.2.2 run_with_field()

```
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)
```

13.271.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.272 test-fgemm.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include "fflas-ffpack/utils/fflas_io.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)

13.272.1 Macro Definition Documentation

13.272.1.1 ENABLE_CHECKER_fgemm

```
#define ENABLE_CHECKER_fgemm 1
```

13.272.2 Function Documentation

13.272.2.1 check_MM()

```
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)
```

13.272.2.2 launch_MM()

```
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)
```

13.272.2.3 launch_MM_dispatch()

```
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)
```

Bug test for ldX equal

Bug

Bug test for transpo

Bug

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

13.272.2.4 run_with_field()

```
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)
```

13.272.2.5 main()

```
int main (
            int argc,
            char ** argv)
```

13.273 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)

13.273.1 Function Documentation

13.273.1.1 check_MV()

```
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)
```

13.273.1.2 launch_MV()

```
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)
```

13.273.1.3 launch_MV_dispatch()

```
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)
```

13.273.1.4 run_with_field()

```
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)
```

13.273.1.5 main()

```
int main (
            int argc,
            char ** argv)
```

13.274 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 RandIter>
  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)
- template<class Field, class RandIter>
  bool launch_fger_dispatch (const Field &F, const size_t nn, const typename Field::Element alpha, const size_t iters, RandIter &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)

13.274.1 Macro Definition Documentation

13.274.1.1 TIME

```
#define TIME 1
```

13.274.2 Function Documentation

13.274.2.1 check_fger()

```
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)
```

13.274.2.2 launch_fger()

```
template<class Field, class RandIter>
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)
```

13.274.2.3 launch_fger_dispatch()

```
template<class Field, class RandIter>
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 ?) ?

13.274.2.4 run_with_field()

```
template<class Field>
bool run_with_field (
            int64_t q,
            uint64_t b,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.274.2.5 main()

```
int main (
            int argc,
            char ** argv)
```

13.275 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)

13.275.1 Function Documentation

13.275.1.1 test_square_fgesv()

```
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)
```

13.275.1.2 test_rect_fgesv()

```
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)
```

13.275.1.3 run_with_field()

```
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)
```

13.275.1.4 main()

```
int main (
            int argc,
            char ** argv)
```

13.276 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)

13.276.1 Function Documentation

13.276.1.1 test_freduce()

```
template<class Field>
bool test_freduce (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed)
```

13.276.1.2 run_with_field()

```
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)
```

13.276.1.3 main()

```
int main (
            int ac,
            char ** av)
```

13.277 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)

- 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.

13.277.1 Function Documentation

13.277.1.1 test_fscal() [1/2]

```
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)
```

13.277.1.2 test_fscal() [2/2]

```
template<class Field>
bool test_fscal (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed)
```

13.277.1.3 test_fscalin() [1/2]

```
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)
```

13.277.1.4 test_fscalin() [2/2]

```
template<class Field>
bool test_fscalin (
            const Field & F,
            size_t m,
            size_t k,
            size_t n,
            bool timing,
            uint64_t seed)
```

13.277.1.5 main()

```
int main (
            int ac,
            char ** av)
```

13.277.1.6 RandomMatrix()

```
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) [inline]
```

Random Matrix.
Creates a m x 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 x lda field elements)
	lda	leading dimension of A
	G	a random iterator

Returns

A.

13.278 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)

13.278.1 Macro Definition Documentation

13.278.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.278.2 Function Documentation

13.278.2.1 check_fsyr2k()

```
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)
```

13.278.2.2 run_with_field()

```
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)
```

13.278.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.279 test-fsyrk.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_fsyrk (const Field &F, size_t n, size_t k, size_t w, 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_UPLO uplo, FFLAS_TRANSPOSE trans, RandIter &Rand)
- template<class Field, class RandIter>
  bool check_computeS1S2 (const Field &F, size_t N, size_t K, FFLAS_TRANSPOSE trans, RandIter &G)
- template<class Field>
  bool run_with_field (Givaro::Integer q, size_t b, size_t n, size_t k, size_t w, int a, int c, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.279.1 Macro Definition Documentation

13.279.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.279.2 Function Documentation

13.279.2.1 check_fsyrk()

```
template<typename Field, class RandIter>
bool check_fsyrk (
            const Field & F,
            size_t n,
            size_t k,
            size_t w,
            const typename Field::Element & alpha,
            const typename Field::Element & beta,
            FFLAS::FFLAS_UPLO uplo,
            FFLAS::FFLAS_TRANSPOSE trans,
            RandIter & Rand)
```

13.279.2.2 check_fsyrk_diag()

```
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)
```

13.279.2.3 check_fsyrk_bkdiag()

```
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_UPLO uplo,
```

```
              FFLAS_TRANSPOSE trans,
              RandIter & Rand)
```

13.279.2.4 check_computeS1S2()

```
template<class Field, class RandIter>
bool check_computeS1S2 (
              const Field & F,
              size_t N,
              size_t K,
              FFLAS_TRANSPOSE trans,
              RandIter & G)
```

13.279.2.5 run_with_field()

```
template<class Field>
bool run_with_field (
              Givaro::Integer q,
              size_t b,
              size_t n,
              size_t k,
              size_t w,
              int a,
              int c,
              size_t iters,
              uint64_t seed)
```

13.279.2.6 main()

```
int main (
              int argc,
              char ** argv)
```

13.280 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/utils/args-parser.h"
#include <iomanip>
#include <random>
#include <chrono>
#include <givaro/modular.h>
#include "fflas-ffpack/utils/test-utils.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)

13.280.1 Function Documentation

13.280.1.1 operator<<()

```
template<typename T>
std::ostream & operator<< (
            std::ostream & os,
            const std::vector< T > & x)
```

13.280.1.2 test_RPM_fsytrf()

```
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)
```

13.280.1.3 test_generic_fsytrf()

```
template<class Field, class RandIter>
bool test_generic_fsytrf (
            Field & F,
            FFLAS_UPLO uplo,
            string file,
            size_t n,
            RandIter & G,
            size_t threshold)
```

13.280.1.4 run_with_field()

```
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)
```

13.280.1.5 main()

```
int main (
            int argc,
            char ** argv)
```

13.281 test-ftrmm.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_ftrmm (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)

13.281.1 Macro Definition Documentation

13.281.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.281.2 Function Documentation

13.281.2.1 check_ftrmm()

```
template<typename Field, class RandIter>
bool check_ftrmm (
            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)
```

13.281.2.2 run_with_field()

```
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)
```

13.281.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.282 test-ftrmv.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_ftrmv (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)

13.282.1 Macro Definition Documentation

13.282.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.282.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.282.2 Function Documentation

13.282.2.1 check_ftrmv()

```
template<typename Field, class RandIter>
bool check_ftrmv (
            const Field & F,
            size_t n,
            FFLAS_UPLO uplo,
            FFLAS_TRANSPOSE trans,
            FFLAS_DIAG diag,
            RandIter & Rand)
```

13.282.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.282.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.283 test-ftrsm-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)

13.283.1 Macro Definition Documentation

13.283.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.283.2 Function Documentation

13.283.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.284 test-ftrsm.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_ftrsm (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)

13.284.1 Macro Definition Documentation

13.284.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.284.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.284.2 Function Documentation

13.284.2.1 check_ftrsm()

```
template<typename Field, class RandIter>
bool check_ftrsm (
            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)
```

13.284.2.2 run_with_field()

```
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)
```

13.284.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.285 test-ftrssyr2k.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_ftrssyr2k (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)

13.285.1 Macro Definition Documentation

13.285.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.285.2 Function Documentation

13.285.2.1 check_ftrssyr2k()

```
template<typename Field, class RandIter>
bool check_ftrssyr2k (
            const Field & F,
            size_t n,
            FFLAS::FFLAS_UPLO uplo,
            FFLAS::FFLAS_DIAG diagA,
            RandIter & Rand)
```

13.285.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
```

```
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.285.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.286 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)

13.286.1 Macro Definition Documentation

13.286.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.286.2 Function Documentation

13.286.2.1 check_ftrstr()

```
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)
```

13.286.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.286.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.287 test-ftrsv.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_ftrsv (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)

13.287.1 Macro Definition Documentation

13.287.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.287.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.287.2 Function Documentation

13.287.2.1 check_ftrsv()

```
template<typename Field, class RandIter>
bool check_ftrsv (
```

```
            const Field & F,
            size_t n,
            FFLAS_UPLO uplo,
            FFLAS_TRANSPOSE trans,
            FFLAS_DIAG diag,
            RandIter & Rand)
```

13.287.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.287.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.288 test-ftrtri.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)

13.288.1 Macro Definition Documentation

13.288.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.288.1.2 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.288.2 Function Documentation

13.288.2.1 check_ftrtri()

```
template<typename Field, class RandIter>
bool check_ftrtri (
            const Field & F,
            size_t n,
            FFLAS_UPLO uplo,
            FFLAS_DIAG diag,
            RandIter & Rand)
```

13.288.2.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.288.2.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.289 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 ()

13.289.1 Function Documentation

13.289.1.1 main()

```
int main (
            void )
```

13.290 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)

13.290.1 Macro Definition Documentation

13.290.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.290.2 Function Documentation

13.290.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.291 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)

13.291.1 Function Documentation

13.291.1.1 run_with_field()

```
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)
```

13.291.1.2 main()

```
int main (
    int argc,
    char ** argv)
```

13.292 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::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 RandIter>
  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<class Field, FFLAS_DIAG diag, FFLAS_TRANSPOSE trans, class RandIter>
  bool launch_test (const Field &F, size_t r, size_t m, size_t n, RandIter &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

13.292.1 Macro Definition Documentation

13.292.1.1 BASECASE_K

```
#define BASECASE_K 37
```

13.292.1.2 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.292.1.3 __LUDIVINE_CUTOFF

```
#define __LUDIVINE_CUTOFF 1
```

13.292.2 Function Documentation

13.292.2.1 test_LUdivine()

```
template<class Field, FFLAS::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 Parameters

Field	Field
Diag	Unit diagonal in U
Trans	

Parameters

F	field
A	Matrix (preallocated)
r	rank of A
m	rows
n	cols
lda	leading dim of A

Returns

0 iff correct, 1 otherwise

13.292.2.2 verifPLUQ()

```
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 Parameters

Field	Field
Diag	Unit diagonal in U

Parameters

F	field
A	Matrix (preallocated)
r	rank of A
m	rows
n	cols
lda	leading dim of A

Returns

0 iff correct, 1 otherwise

13.292.2.3 test_pluq()

```
template<class Field, FFLAS_DIAG diag, class RandIter>
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

Field	Field
Diag	Unit diagonal in U
Trans	

Parameters

F	field
A	Matrix (preallocated)
r	rank of A
m	rows
n	cols
lda	leading dim of A

Returns

0 iff correct, 1 otherwise

13.292.2.4 launch_test()

```
template<class Field, FFLAS_DIAG diag, FFLAS_TRANSPOSE trans, class RandIter>
bool launch_test (
            const Field & F,
            size_t r,
            size_t m,
            size_t n,
            RandIter & G)
```

13.292.2.5 run_with_field()

```
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)
```

13.292.2.6 main()

```
int main (
            int argc,
            char ** argv)
```

13.292.3 Variable Documentation

13.292.3.1 tperm

```
Givaro::Timer tperm
```

13.292.3.2 tgemm

```
Givaro::Timer tgemm
```

13.292.3.3 tBC

```
Givaro::Timer tBC
```

13.292.3.4 ttrsm

```
Givaro::Timer ttrsm
```

13.292.3.5 trest

```
Givaro::Timer trest
```

13.292.3.6 timtot

```
Givaro::Timer timtot
```

13.292.3.7 mvcnt

```
size_t mvcnt = 0
```

13.293 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)

Functions

- template<class Field>
  bool test (Givaro::Integer p, size_t kmax)
- int main ()

13.293.1 Macro Definition Documentation

13.293.1.1 MAX_WITH_SIZE_T

```
#define MAX_WITH_SIZE_T(
        x)
```

Value:

```
( (static_cast<uint64_t>(std::numeric_limits<size_t>::max()) <  x)? std::numeric_limits<size_t>::max() : x )
```

13.293.2 Function Documentation

13.293.2.1 test()

```
template<class Field>
bool test (
        Givaro::Integer p,
        size_t kmax)
```

13.293.2.2 main()

```
int main (
        void )
```

13.294 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/utils/args-parser.h"
#include "fflas-ffpack/utils/fflas_randommatrix.h"
#include "fflas-ffpack/utils/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)

13.294.1 Function Documentation

13.294.1.1 check_minpoly()

```
template<typename Field, class RandIter>
bool check_minpoly (
            const Field & F,
            size_t n,
            RandIter & G)
```

13.294.1.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t n,
            size_t iters,
            uint64_t seed)
```

13.294.1.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.295 test-multifile1.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

13.296 test-multifile2.C File Reference

```
#include "fflas-ffpack/fflas-ffpack.h"
```

Functions

- int main (void)

13.296.1 Function Documentation

13.296.1.1 main()

```
int main (
            void )
```

13.297 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)

13.297.1 Function Documentation

13.297.1.1 checkingMessage()

```
template<class Field>
std::string checkingMessage (
            const Field & F)
```

13.297.1.2 readOrRandomMatrixWithRankAndRandomRPM()

```
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.

Otherwise, generate a random matrix of size m x n with random lda.

13.297.1.3 test_nullspace()

```
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)
```

13.297.1.4 run_with_field()

```
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)
```

13.297.1.5 main()

```
int main (
            int argc,
            char ** argv)
```

13.298 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

13.298.1 Function Documentation

13.298.1.1 checkMonotonicApplyP()

```
bool checkMonotonicApplyP (
            FFLAS_SIDE Side,
            FFLAS_TRANSPOSE trans,
            size_t * P,
            size_t N,
            size_t R)
```

13.298.1.2 main()

```
int main (
            void )
```

13.298.2 Variable Documentation

13.298.2.1 tperm

```
Givaro::Timer tperm
```

13.298.2.2 tgemm

```
Givaro::Timer tgemm
```

13.298.2.3 tBC

```
Givaro::Timer tBC
```

13.298.2.4 ttrsm

```
Givaro::Timer ttrsm
```

13.298.2.5 trest

```
Givaro::Timer trest
```

13.298.2.6 timtot

```
Givaro::Timer timtot
```

13.299 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)

13.299.1 Macro Definition Documentation

13.299.1.1 ENABLE_ALL_CHECKINGS

```
#define ENABLE_ALL_CHECKINGS 1
```

13.299.2 Function Documentation

13.299.2.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.300 test-quasisep.C File Reference

```
#include "fflas-ffpack/fflas-ffpack-config.h"
#include <givaro/modular-balanced.h>
#include <iostream>
#include <iomanip>
#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>
```

Functions

- template<class Field, FFLAS_DIAG diag, class RandIter>
  bool test_BruhatGenerator (const Field &F, size_t n, size_t r, size_t t, typename Field::ConstElement_ptr A, size_t lda, typename Field::Element_ptr TS, size_t l, RandIter &G)
- template<class Field, FFLAS_DIAG diag, class RandIter>
  bool launch_test (const Field &F, size_t n, size_t r, size_t t, size_t l, RandIter &G)
- template<class Field, class RandGen>
  bool testLTQSRPM (const Field &F, size_t n, size_t r, size_t t, RandGen &G)
- template<class Field>
  bool run_with_field (Givaro::Integer q, uint64_t b, size_t n, size_t r, size_t t, size_t l, size_t iters, uint64_t seed)
- int main (int argc, char **argv)

13.300.1 Function Documentation

13.300.1.1 test_BruhatGenerator()

```
template<class Field, FFLAS_DIAG diag, class RandIter>
bool test_BruhatGenerator (
            const Field & F,
            size_t n,
            size_t r,
            size_t t,
            typename Field::ConstElement_ptr A,
            size_t lda,
            typename Field::Element_ptr TS,
            size_t l,
            RandIter & G)
```

13.300.1.2 launch_test()

```
template<class Field, FFLAS_DIAG diag, class RandIter>
bool launch_test (
            const Field & F,
            size_t n,
            size_t r,
            size_t t,
            size_t l,
            RandIter & G)
```

13.300.1.3 testLTQSRPM()

```
template<class Field, class RandGen>
bool testLTQSRPM (
            const Field & F,
            size_t n,
            size_t r,
            size_t t,
            RandGen & G)
```

13.300.1.4 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            uint64_t b,
            size_t n,
            size_t r,
            size_t t,
            size_t l,
            size_t iters,
            uint64_t seed)
```

13.300.1.5 main()

```
int main (
            int argc,
            char ** argv)
```

13.301 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)

13.301.1 Macro Definition Documentation

13.301.1.1 __FFLASFFPACK_SEQUENTIAL

```
#define __FFLASFFPACK_SEQUENTIAL
```

13.301.2 Function Documentation

13.301.2.1 run_with_field()

```
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)
```

13.301.2.2 main()

```
int main (
            int argc,
            char ** argv)
```

13.302 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)

13.302.1 Function Documentation

13.302.1.1 checkRPM()

```
bool checkRPM (
            size_t M,
            size_t N,
            size_t R)
```

13.302.1.2 checkSymmetricRPM()

```
bool checkSymmetricRPM (
            size_t N,
            size_t R)
```

13.302.1.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.303 test-simd.C File Reference

```
#include "givaro/givinteger.h"
#include "givaro/modular.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 ALL< true, v... >
- struct ALL< false, v... >
- struct ALL<>
- struct count_nonconst_lvalue_reference< T, O... >
- struct count_nonconst_lvalue_reference< T &, O... >
- struct count_nonconst_lvalue_reference< const T &, O... >
- struct count_nonconst_lvalue_reference<>
- struct is_all_same< T, Args... >
- struct is_all_same<>
- struct width< T >
- struct width< float >
- struct width< double >
- class TestOneMethod< Simd >
- struct ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >
- class ScalFunctionsBase< Element, typename enable_if< is_floating_point< Element >::value >::type >::FloatingPointTestD
- struct ScalFunctionsBase< Element, typename enable_if< is_integral< Element >::value >::type >
- struct ScalFunctions< Element >
- class string

  STL class.
- class vector< T >

  STL class.
- class array< T >

  STL class.

Macros

- #define _TEST_ONE(K, f1, f2, r, n)
- #define TEST_ONE_OP(f)
- #define TEST_ONE_OP_WZ(f)
- #define TEST_IMPL(SIZE, Elt)

Functions

- template<typename Element>
  enable_if< is_integral< Element >::value, bool >::type check_eq (Element x, Element y)
- template<typename Element>
  enable_if< is_floating_point< Element >::value, bool >::type check_eq (Element x, Element y)
- template<typename Element>
  bool cmp (vector< Element > out_scal, vector< Element > out_simd)
- template<typename Ret, typename T>
  Ret eval_func_on_array (function< Ret()> f, array< T, 0 > &arr)
- template<typename T, typename... TArgs>
  void eval_func_on_array (function< void(T, TArgs...)> f, array< typename decay< T >::type, sizeof...(TArgs)+1 > &arr)
- template<typename Ret, typename T, typename... TArgs>
  Ret eval_func_on_array (function< Ret(T, TArgs...)> f, array< typename decay< T >::type, sizeof...(TArgs)+1 > &arr)
- template<typename E>
  std::ostream & operator<< (std::ostream &o, const vector< E > &V)
- template<typename Simd, typename Element>
  enable_if< is_floating_point< Element >::value, bool >::type test_impl_base ()
- template<typename Simd, typename Element>
  enable_if< is_integral< Element >::value, bool >::type test_impl_base ()
- template<typename Simd, typename Element>
  bool test_impl ()
- int main (int argc, char ∗argv[ ])

13.303.1 Macro Definition Documentation

13.303.1.1 _TEST_ONE

```
#define _TEST_ONE(
                K,
                f1,
                f2,
                r,
                n)
```

Value:

```
        do {      \
        K T(f1, f2, r, n);              \
        bool b = T.writeResultLine();   \
        if (b == false)                 \
            T.writeDebugData();         \
        btest &= b;                     \
    } while (0)
```

13.303.1.2 TEST_ONE_OP

```
#define TEST_ONE_OP(
                f)
```

Value:

```
        _TEST_ONE(TestOneMethod<Simd>,              \
        function<decltype(Simd::f)>(Simd::f),                     \
        function<decltype(Scal::f)>(Scal::f),                     \
        function<decltype(Scal::genInputs)>(Scal::genInputs), #f)
```

13.303.1.3 TEST_ONE_OP_WZ

```
#define TEST_ONE_OP_WZ(
                f)
```

Value:

```
_TEST_ONE(TestOneMethod<Simd>,                        \
function<decltype(Simd::f)>(Simd::f),                      \
function<decltype(Scal::f)>(Scal::f),                      \
function<decltype(Scal::genInputsWithZero)>(Scal::genInputsWithZero),\
#f " test with zero")
```

13.303.1.4 TEST_IMPL

```
#define TEST_IMPL(
                SIZE,
                Elt)
```

Value:

```
    do {        \
    pass &= test_impl<Simd##SIZE<Elt>, Elt>(); \
    cout « endl; \
} while (0)
```

13.303.2 Function Documentation

13.303.2.1 check_eq() [1/2]

```
template<typename Element>
enable_if< is_integral< Element >::value, bool >::type check_eq (
            Element x,
            Element y)
```

13.303.2.2 check_eq() [2/2]

```
template<typename Element>
enable_if< is_floating_point< Element >::value, bool >::type check_eq (
            Element x,
            Element y)
```

13.303.2.3 cmp()

```
template<typename Element>
bool cmp (
            vector< Element > out_scal,
            vector< Element > out_simd)
```

13.303.2.4 eval_func_on_array() [1/3]

```
template<typename Ret, typename T>
Ret eval_func_on_array (
            function< Ret()> f,
            array< T, 0 > & arr)
```

13.303.2.5 eval_func_on_array() [2/3]

```
template<typename T, typename... TArgs>
void eval_func_on_array (
            function< void(T, TArgs...)> f,
            array< typename decay< T >::type, sizeof...(TArgs)+1 > & arr)
```

13.303.2.6 eval_func_on_array() [3/3]

```
template<typename Ret, typename T, typename... TArgs>
Ret eval_func_on_array (
```

```
            function< Ret(T, TArgs...)> f,
            array< typename decay< T >::type, sizeof...(TArgs)+1 > & arr)
```

13.303.2.7 operator<<()

```
template<typename E>
std::ostream & operator<< (
            std::ostream & o,
            const vector< E > & V)
```

13.303.2.8 test_impl_base() [1/2]

```
template<typename Simd, typename Element>
enable_if< is_floating_point< Element >::value, bool >::type test_impl_base ()
```

13.303.2.9 test_impl_base() [2/2]

```
template<typename Simd, typename Element>
enable_if< is_integral< Element >::value, bool >::type test_impl_base ()
```

13.303.2.10 test_impl()

```
template<typename Simd, typename Element>
bool test_impl ()
```

13.303.2.11 main()

```
int main (
            int argc,
            char * argv[])
```

13.304 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)

13.304.1 Function Documentation

13.304.1.1 check_solve()

```
template<typename Field, class RandIter>
bool check_solve (
            const Field & F,
```

```
            size_t m,
            RandIter & Rand,
            bool isParallel)
```

13.304.1.2 run_with_field()

```
template<class Field>
bool run_with_field (
            Givaro::Integer q,
            size_t b,
            size_t m,
            size_t iters,
            uint64_t seed)
```

13.304.1.3 main()

```
int main (
            int argc,
            char ** argv)
```

13.305 test-storage-transpose.C File Reference

```
#include <iomanip>
#include <iostream>
#include <random>
#include "fflas-ffpack/utils/args-parser.h"
#include "fflas-ffpack/utils/test-utils.h"
#include "fflas-ffpack/utils/fflas_io.h"
#include "fflas-ffpack/utils/fflas_memory.h"
#include <givaro/modular.h>
#include <recint/rint.h>
#include "fflas-ffpack/fflas/fflas_transpose.h"
```

Data Structures

- class Test< Elt >

Functions

- int main (int argc, char **argv)

13.305.1 Function Documentation

13.305.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.306 101-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)

13.306.1 Function Documentation

13.306.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.307 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)

13.307.1 Function Documentation

13.307.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.308 2x2-ftrsv.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)

13.308.1 Function Documentation

13.308.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.309 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)

13.309.1 Function Documentation

13.309.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.310 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)

13.310.1 Function Documentation

13.310.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.311 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)

13.311.1 Function Documentation

13.311.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.312 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)

13.312.1 Function Documentation

13.312.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.313 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)

13.313.1 Function Documentation

13.313.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.314 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)

13.314.1 Function Documentation

13.314.1.1 main()

```
int main (
            int argc,
            char ** argv)
```

13.315 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)

13.315.1 Function Documentation

13.315.1.1 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*x will not be equal to b for the later verification stage

Index