LibRapid

Generated by Doxygen 1.9.3

Chapter 1

FLENS: Flexible Library for Efficient Numerical Solutions

This folder contains slightly modified code from the FLENS Library. The FLENS library is published under a BSD-3-Clause license.

Chapter 2

LibRapid

Hello!

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

Chapter 5

File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 6

Class Documentation

6.1 librapid::ArrayContainer< ShapeType_, StorageType_ > Class Template Reference

Public Types

- using **StorageType** = StorageType_
- using **ShapeType** = ShapeType_
- using **SizeType** = typename ShapeType::SizeType
- using **Scalar** = typename StorageType::Scalar
- using **Packet** = typename typetraits::TypeInfo< Scalar >::Packet

Public Member Functions

- **ArrayContainer** ()=default

  Default constructor.
- LIBRAPID_ALWAYS_INLINE ArrayContainer (const ShapeType &shape)
- LIBRAPID_ALWAYS_INLINE ArrayContainer (const ShapeType &shape, const Scalar &value)
- LIBRAPID_ALWAYS_INLINE ArrayContainer (ShapeType &&shape)
- LIBRAPID_ALWAYS_INLINE ArrayContainer (const ArrayContainer &other)=default
- LIBRAPID_ALWAYS_INLINE ArrayContainer (ArrayContainer &&other) noexcept=default
- template<typename Functor_ , typename... Args>
  LIBRAPID_ALWAYS_INLINE ArrayContainer (const detail::Function< Functor_, Args... > &function) LIBRAPID_RELEASE_NOEXCEPT
- LIBRAPID_ALWAYS_INLINE ArrayContainer & operator= (const ArrayContainer &other)=default
- LIBRAPID_ALWAYS_INLINE ArrayContainer & operator= (ArrayContainer &&other) noexcept=default
- template<typename Functor_ , typename... Args>
  LIBRAPID_ALWAYS_INLINE ArrayContainer & operator= (const detail::Function< Functor_, Args... > &function)
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE const ShapeType & shape () const noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Packet packet (size_t index) const
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Scalar scalar (size_t index) const
- LIBRAPID_ALWAYS_INLINE void writePacket (size_t index, const Packet &value)
- LIBRAPID_ALWAYS_INLINE void write (size_t index, const Scalar &value)
- template<typename Functor_ , typename... Args>
  ArrayContainer (const detail::Function< Functor_, Args... > &function) LIBRAPID_RELEASE_NOEXCEPT
- template<typename Functor_ , typename... Args>
  ArrayContainer< ShapeType_, StorageType_ > & **operator=** (const detail::Function< Functor_, Args... > &function)

Public Attributes

- ShapeType **m_shape**
- StorageType **m_storage**

6.1.1 Constructor & Destructor Documentation

6.1.1.1 ArrayContainer() [1/6]

```
template<typename ShapeType_ , typename StorageType_ >
librapid::ArrayContainer< ShapeType_, StorageType_ >::ArrayContainer (
            const ShapeType & shape )  [explicit]
```

Constructs an array container from a shape

Parameters

shape	The shape of the array container

6.1.1.2 ArrayContainer() [2/6]

```
template<typename ShapeType_ , typename StorageType_ >
librapid::ArrayContainer< ShapeType_, StorageType_ >::ArrayContainer (
            const ShapeType & shape,
            const Scalar & value )
```

Create an array container from a shape and a scalar value. The scalar value represents the value the memory is initialized with.

Parameters

shape	The shape of the array container
value	The value to initialize the memory with

6.1.1.3 ArrayContainer() [3/6]

```
template<typename ShapeType_ , typename StorageType_ >
librapid::ArrayContainer< ShapeType_, StorageType_ >::ArrayContainer (
            ShapeType && shape )  [explicit]
```

Construct an array container from a shape, which is moved, not copied.

Parameters

shape	The shape of the array container

6.1.1.4 ArrayContainer() [4/6]

```
template<typename ShapeType_ , typename StorageType_ >
LIBRAPID_ALWAYS_INLINE librapid::ArrayContainer< ShapeType_, StorageType_ >::ArrayContainer (
            const ArrayContainer< ShapeType_, StorageType_ > & other )  [default]
```

Construct an array container from another array container.

Parameters

other	The array container to copy.

6.1.1.5 ArrayContainer() [5/6]

```
template<typename ShapeType_ , typename StorageType_ >
LIBRAPID_ALWAYS_INLINE librapid::ArrayContainer< ShapeType_, StorageType_ >::ArrayContainer (
            ArrayContainer< ShapeType_, StorageType_ > && other )  [default], [noexcept]
```

Construct an array container from a temporary array container.

Parameters

other	The array container to move.

6.1.1.6 ArrayContainer() [6/6]

```
template<typename ShapeType_ , typename StorageType_ >
template<typename Functor_ , typename...  Args>
LIBRAPID_ALWAYS_INLINE librapid::ArrayContainer< ShapeType_, StorageType_ >::ArrayContainer (
            const detail::Function< Functor_, Args...  > & function )  [explicit]
```

Construct an array container from a function object. This will assign the result of the function to the array container, evaluating it accordingly.

Template Parameters

*Functor_*	The function type
Args	The argument types of the function

Parameters

function	The function to assign

6.1.2 Member Function Documentation

6.1.2.1 operator=() [1/3]

```
template<typename ShapeType_ , typename StorageType_ >
LIBRAPID_ALWAYS_INLINE ArrayContainer & librapid::ArrayContainer< ShapeType_, StorageType_ >↩
::operator= (
            ArrayContainer< ShapeType_, StorageType_ > && other )  [default], [noexcept]
```

Assign a temporary array container to this array container.

Parameters

other	The array container to move.

Returns

A reference to this array container.

6.1.2.2 operator=() [2/3]

```
template<typename ShapeType_ , typename StorageType_ >
LIBRAPID_ALWAYS_INLINE ArrayContainer & librapid::ArrayContainer< ShapeType_, StorageType_ >↩
::operator= (
            const ArrayContainer< ShapeType_, StorageType_ > & other )  [default]
```

Assign an array container to this array container.

Parameters

other	The array container to copy.

Returns

A reference to this array container.

6.1.2.3 operator=() [3/3]

```
template<typename ShapeType_ , typename StorageType_ >
template<typename Functor_ , typename... Args>
LIBRAPID_ALWAYS_INLINE ArrayContainer & librapid::ArrayContainer< ShapeType_, StorageType_ >↩
::operator= (
            const detail::Function< Functor_, Args... > & function )
```

Assign a function object to this array container. This will assign the result of the function to the array container, evaluating it accordingly.

Template Parameters

*Functor_*	The function type
Args	The argument types of the function

Parameters

function	The function to assign

Returns

A reference to this array container.

6.1.2.4 packet()

```
template<typename ShapeType_ , typename StorageType_ >
auto librapid::ArrayContainer< ShapeType_, StorageType_ >::packet (
            size_t index ) const
```

Return a Packet object from the array's storage at a specific index.

Parameters

index	The index to get the packet from

Returns

A Packet object from the array's storage at a specific index

6.1.2.5 scalar()

```
template<typename ShapeType_ , typename StorageType_ >
auto librapid::ArrayContainer< ShapeType_, StorageType_ >::scalar (
            size_t index ) const
```

Return a Scalar from the array's storage at a specific index.

Parameters

index	The index to get the scalar from

Returns

A Scalar from the array's storage at a specific index

6.1.2.6 shape()

```
template<typename ShapeType_ , typename StorageType_ >
auto librapid::ArrayContainer< ShapeType_, StorageType_ >::shape  [noexcept]
```

Return the shape of the array container. This is an immutable reference.

Returns

The shape of the array container.

6.1.2.7 write()

```
template<typename ShapeType_ , typename StorageType_ >
void librapid::ArrayContainer< ShapeType_, StorageType_ >::write (
            size_t index,
            const Scalar & value )
```

Write a Scalar to the array's storage at a specific index

Parameters

index	The index to write the scalar to
value	The value to write to the array's storage

6.1.2.8 writePacket()

```
template<typename ShapeType_ , typename StorageType_ >
void librapid::ArrayContainer< ShapeType_, StorageType_ >::writePacket (
            size_t index,
            const Packet & value )
```

Write a Packet object to the array's storage at a specific index

Parameters

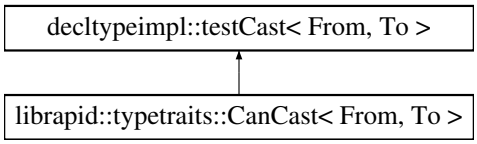
index	The index to write the packet to
value	The value to write to the array's storage

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

- librapid/include/librapid/array/arrayContainer.hpp

6.2 librapid::typetraits::CanCast< From, To > Struct Template Reference

Inheritance diagram for librapid::typetraits::CanCast< From, To >:



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

- librapid/include/librapid/core/typetraits.hpp

6.3 cxxblas::ComplexTrait< T > Struct Template Reference

Public Types

- typedef T **PrimitiveType**

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

- librapid/cxxblas/auxiliary/complextrait.h

6.4 cxxblas::ComplexTrait< std::complex< T > > Struct Template Reference

Public Types

- typedef T **PrimitiveType**

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

- librapid/cxxblas/auxiliary/complextrait.h

6.5 helper_image_internal::ConverterFromUByte< T > Struct Template Reference

Data converter from unsigned char / unsigned byte to type T.

6.5.1 Detailed Description

template<class T>
struct helper_image_internal::ConverterFromUByte< T >

Data converter from unsigned char / unsigned byte to type T.

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

- librapid/include/librapid/cuda/helper_image.h

6.6 helper_image_internal::ConverterFromUByte< float > Struct Reference

Data converter from unsigned char / unsigned byte to float.

`#include <helper_image.h>`

Public Member Functions

- float operator() (const unsigned char &val)

6.6.1 Detailed Description

Data converter from unsigned char / unsigned byte to float.

6.6.2 Member Function Documentation

6.6.2.1 operator()()

```
float helper_image_internal::ConverterFromUByte< float >::operator() (
            const unsigned char & val )  [inline]
```

Conversion operator

Returns

converted value

Parameters

val	value to convert

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

- librapid/include/librapid/cuda/helper_image.h

6.7 helper_image_internal::ConverterFromUByte< unsigned char > Struct Reference

Data converter from unsigned char / unsigned byte.

`#include <helper_image.h>`

Public Member Functions

- float operator() (const unsigned char &val)

6.7.1 Detailed Description

Data converter from unsigned char / unsigned byte.

6.7.2 Member Function Documentation

6.7.2.1 operator()()

```
float helper_image_internal::ConverterFromUByte< unsigned char >::operator() (
            const unsigned char & val )  [inline]
```

Conversion operator

Returns

converted value

Parameters

val	value to convert

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

- librapid/include/librapid/cuda/helper_image.h

6.8 helper_image_internal::ConverterToUByte< T > Struct Template Reference

Data converter from unsigned char / unsigned byte to type T.

6.8.1 Detailed Description

template<class T>
struct helper_image_internal::ConverterToUByte< T >

Data converter from unsigned char / unsigned byte to type T.

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

- librapid/include/librapid/cuda/helper_image.h

6.9 helper_image_internal::ConverterToUByte< float > Struct Reference

Data converter from unsigned char / unsigned byte to unsigned int.

`#include <helper_image.h>`

Public Member Functions

- unsigned char operator() (const float &val)

6.9.1 Detailed Description

Data converter from unsigned char / unsigned byte to unsigned int.

6.9.2 Member Function Documentation

6.9.2.1 operator()()

```
unsigned char helper_image_internal::ConverterToUByte< float >::operator() (
            const float & val )  [inline]
```

Conversion operator

Returns

converted value

Parameters

val	value to convert

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

- librapid/include/librapid/cuda/helper_image.h

6.10 helper_image_internal::ConverterToUByte< unsigned char > Struct Reference

Data converter from unsigned char / unsigned byte to unsigned int.

`#include <helper_image.h>`

Public Member Functions

- unsigned char operator() (const unsigned char &val)

6.10.1 Detailed Description

Data converter from unsigned char / unsigned byte to unsigned int.

6.10.2 Member Function Documentation

6.10.2.1 operator()()

```
unsigned char helper_image_internal::ConverterToUByte< unsigned char >::operator() (
            const unsigned char & val )  [inline]
```

Conversion operator (essentially a passthru

Returns

converted value

Parameters

val	value to convert

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

- librapid/include/librapid/cuda/helper_image.h

6.11 librapid::device::CPU Struct Reference

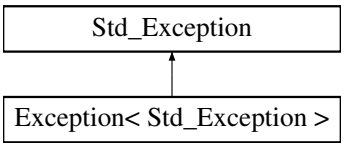
The documentation for this struct was generated from the following file:

- librapid/include/librapid/core/cudaConfig.hpp

6.12 Exception< Std_Exception > Class Template Reference

`#include <exception.h>`

Inheritance diagram for Exception< Std_Exception >:



Public Member Functions

- virtual ~**Exception** () throw ()
  Destructor.

Static Public Member Functions

- static void throw_it (const char *file, const int line, const char *detailed="-")
  Static construction interface.
- static void throw_it (const char *file, const int line, const std::string &detailed)

6.12.1 Detailed Description

template<class Std_Exception>
class Exception< Std_Exception >

Exception wrapper.

Parameters

Std_Exception	Exception out of namespace std for easy typing.

6.12.2 Member Function Documentation

6.12.2.1 throw_it() [1/2]

```
template<class Std_Exception >
void Exception< Std_Exception >::throw_it (
            const char * file,
            const int line,
            const char * detailed = "-" )  [static]
```

Static construction interface.

Implementation.

Returns

Alwayss throws (Located_Exception<Exception>)

Parameters

file	file in which the Exception occurs
line	line in which the Exception occurs
detailed	details on the code fragment causing the Exception

Static construction interface.

Parameters

Exception	causing code fragment (file and line) and detailed infos.

6.12.2.2 throw_it() [2/2]

```
template<class Std_Exception >
void Exception< Std_Exception >::throw_it (
            const char * file,
            const int line,
            const std::string & msg )  [static]
```

Static construction interface

Returns

Alwayss throws (Located_Exception<Exception>)

Parameters

file	file in which the Exception occurs
line	line in which the Exception occurs
detailed	details on the code fragment causing the Exception

Static construction interface.

Parameters

Exception	causing code fragment (file and line) and detailed infos.

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

- librapid/include/librapid/cuda/exception.h

6.13 librapid::detail::Function< Functor_, Args > Class Template Reference

Public Types

- using **Type** = Function< Functor_, Args... >
- using **Functor** = Functor_
- using **Scalar** = typename typetraits::TypeInfo< Type >::Scalar
- using **Packet** = typename typetraits::TypeInfo< Scalar >::Packet

Public Member Functions

- LIBRAPID_ALWAYS_INLINE Function (Functor &&functor, Args &&...args)
- LIBRAPID_ALWAYS_INLINE Function (const Function &other)=default
- LIBRAPID_ALWAYS_INLINE Function (Function &&other) noexcept=default
- LIBRAPID_ALWAYS_INLINE Function & operator= (const Function &other)=default
- LIBRAPID_ALWAYS_INLINE Function & operator= (Function &&other) noexcept=default
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE auto **shape** () const
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Packet packet (size_t index) const
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Scalar scalar (size_t index) const
- template<size_t... I>
  Function< Functor, Args... >::Packet **packetImpl** (std::index_sequence< I... >, size_t index) const
- template<size_t... I>
  auto **scalarImpl** (std::index_sequence< I... >, size_t index) const -> Scalar

6.13.1 Constructor & Destructor Documentation

6.13.1.1 Function() [1/3]

```
template<typename Functor , typename... Args>
librapid::detail::Function< Functor, Args >::Function (
            Functor && functor,
            Args &&... args ) [explicit]
```

Constructs a function from a functor and arguments.

Parameters

functor	The functor to use.
args	The arguments to use.

6.13.1.2 Function() [2/3]

```
template<typename Functor_ , typename...  Args>
LIBRAPID_ALWAYS_INLINE librapid::detail::Function< Functor_, Args >::Function (
            const Function< Functor_, Args > & other )  [default]
```

Constructs a function from another function.

Parameters

other	The function to copy.

6.13.1.3 Function() [3/3]

```
template<typename Functor_ , typename...  Args>
LIBRAPID_ALWAYS_INLINE librapid::detail::Function< Functor_, Args >::Function (
            Function< Functor_, Args > && other )  [default], [noexcept]
```

Construct a function from a temporary function.

Parameters

other	The function to move.

6.13.2 Member Function Documentation

6.13.2.1 operator=() [1/2]

```
template<typename Functor_ , typename...  Args>
LIBRAPID_ALWAYS_INLINE Function & librapid::detail::Function< Functor_, Args >::operator= (
            const Function< Functor_, Args > & other )  [default]
```

Assigns a function to this function.

Parameters

other	The function to copy.

Returns

A reference to this function.

6.13.2.2 operator=() [2/2]

```
template<typename Functor_ , typename...  Args>
LIBRAPID_ALWAYS_INLINE Function & librapid::detail::Function< Functor_, Args >::operator= (
            Function< Functor_, Args > && other )  [default], [noexcept]
```

Assigns a temporary function to this function.

Parameters

other	The function to move.

Returns

A reference to this function.

6.13.2.3 packet()

```
template<typename Functor , typename...  Args>
Function< Functor, Args...  >::Packet librapid::detail::Function< Functor, Args >::packet (
            size_t index ) const
```

Evaluates the function at the given index, returning a Packet result.

Parameters

index	The index to evaluate at.

Returns

The result of the function (vectorized).

6.13.2.4 scalar()

```
template<typename Functor , typename...  Args>
auto librapid::detail::Function< Functor, Args >::scalar (
            size_t index ) const
```

Evaluates the function at the given index, returning a Scalar result.

Parameters

index	The index to evaluate at.

Returns

The result of the function (scalar).

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

- librapid/include/librapid/array/function.hpp

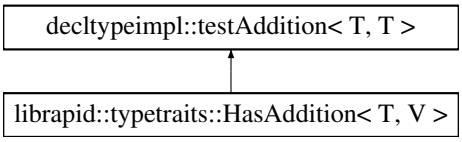
6.14 librapid::device::GPU Struct Reference

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

- librapid/include/librapid/core/cudaConfig.hpp

6.15 librapid::typetraits::HasAddition< T, V > Struct Template Reference

Inheritance diagram for librapid::typetraits::HasAddition< T, V >:

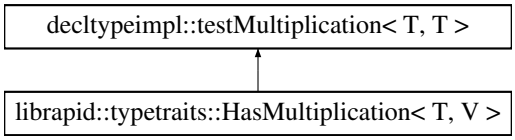


The documentation for this struct was generated from the following file:

- librapid/include/librapid/core/typetraits.hpp

6.16 librapid::typetraits::HasMultiplication< T, V > Struct Template Reference

Inheritance diagram for librapid::typetraits::HasMultiplication< T, V >:

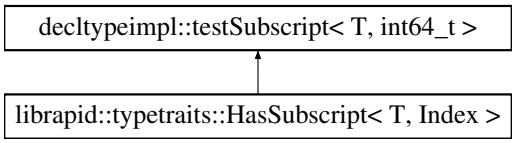


The documentation for this struct was generated from the following file:

- librapid/include/librapid/core/typetraits.hpp

6.17 librapid::typetraits::HasSubscript< T, Index > Struct Template Reference

Inheritance diagram for librapid::typetraits::HasSubscript< T, Index >:



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

- librapid/include/librapid/core/typetraits.hpp

6.18 cxxblas::If< Any > Struct Template Reference

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

- librapid/cxxblas/drivers/drivers.h

6.19 cxxblas::If< int > Struct Reference

Public Types

- typedef void **isBlasCompatibleInteger**

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

- librapid/cxxblas/drivers/drivers.h

6.20 cxxblas::If< long > Struct Reference

Public Types

- typedef void **isBlasCompatibleInteger**

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

- librapid/cxxblas/drivers/drivers.h

6.21 cxxblas::IsComplex< T > Struct Template Reference

Static Public Attributes

- static const bool **value** = !IsNotComplex<T>::value

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

- librapid/cxxblas/auxiliary/iscomplex.h

6.22 cxxblas::IsNotComplex< T > Struct Template Reference

Static Public Attributes

- static const bool **value** = true

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

- librapid/cxxblas/auxiliary/iscomplex.h

6.23 cxxblas::IsNotComplex< std::complex< T > > Struct Template Reference

Static Public Attributes

- static const bool **value** = false

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

- librapid/cxxblas/auxiliary/iscomplex.h

6.24 cxxblas::IsSame< Args > Struct Template Reference

Static Public Attributes

- static const bool **value** = false

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

- librapid/cxxblas/auxiliary/issame.h

6.25 cxxblas::IsSame< T > Struct Template Reference

Static Public Attributes

- static const bool **value** = true

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

- librapid/cxxblas/auxiliary/issame.h

6.26 cxxblas::IsSame< T, T > Struct Template Reference

Static Public Attributes

- static const bool **value** = true

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

- librapid/cxxblas/auxiliary/issame.h

6.27 cxxblas::IsSame< T, T, Args... > Struct Template Reference

Static Public Attributes

- static const bool **value** = IsSame<T, Args...>::value

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

- librapid/cxxblas/auxiliary/issame.h

6.28 librapid::detail::PreMain Class Reference

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

- librapid/include/librapid/core/preMain.hpp

6.29 cxxblas::RestrictTo< b, T > Struct Template Reference

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

- librapid/cxxblas/auxiliary/restrictto.h

6.30 cxxblas::RestrictTo< true, T > Struct Template Reference

Public Types

- typedef std::remove_reference< T >::type **Type**

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

- librapid/cxxblas/auxiliary/restrictto.h

6.31 librapid::Shape< T, N > Class Template Reference

Public Types

- using **SizeType** = T

Public Member Functions

- **Shape** ()=default

  Default constructor.
- template<typename V , typename typetraits::EnableIf< typetraits::CanCast< V, T >::value > = 0>
  Shape (const std::initializer_list< V > &vals)
- template<typename V , typename typetraits::EnableIf< typetraits::CanCast< V, T >::value > = 0>
  Shape (const std::vector< V > &vals)
- Shape (const Shape &other)=default
- Shape (Shape &&other) noexcept=default
- template<typename V , size_t Dim>
  Shape (const Shape< V, Dim > &other)
- template<typename V , size_t Dim>
  Shape (Shape< V, Dim > &&other) noexcept
- template<typename V , typename typetraits::EnableIf< typetraits::CanCast< V, T >::value > = 0>
  Shape & operator= (const std::initializer_list< V > &vals)

- template<typename V , typename typetraits::EnableIf< typetraits::CanCast< V, T >::value > = 0>
  Shape & operator= (const std::vector< V > &vals)
- Shape & operator= (Shape &&other) noexcept=default
- template<typename Index >
  LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE const T & operator[] (Index index) const
- template<typename Index >
  LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE T & operator[] (Index index)
- LIBRAPID_ALWAYS_INLINE bool operator== (const Shape &other) const
- LIBRAPID_ALWAYS_INLINE bool operator!= (const Shape &other) const
- LIBRAPID_NODISCARD T ndim () const
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE T size () const
- LIBRAPID_NODISCARD std::string str () const
- template<typename V , typename typetraits::EnableIf< typetraits::CanCast< V, T >::value > >
  Shape< T, N > & **operator=** (const std::initializer_list< V > &vals)
- template<typename V , typename typetraits::EnableIf< typetraits::CanCast< V, T >::value > >
  Shape< T, N > & **operator=** (const std::vector< V > &vals)

Static Public Member Functions

- static Shape zeros (size_t dims)
- static Shape ones (size_t dims)

Static Public Attributes

- static constexpr size_t **MaxDimensions** = N

6.31.1 Constructor & Destructor Documentation

6.31.1.1 Shape() [1/6]

```
template<typename T , size_t N>
template<typename V , typename typetraits::EnableIf< typetraits::CanCast< V, T >::value > >
librapid::Shape< T, N >::Shape (
            const std::initializer_list< V > & vals )
```

Create a Shape object from a list of values

Template Parameters

V	Scalar type of the values

Parameters

vals	The dimensions for the object

6.31.1.2 Shape() [2/6]

```
template<typename T , size_t N>
template<typename V , typename typetraits::EnableIf< typetraits::CanCast< V, T >::value > >
librapid::Shape< T, N >::Shape (
            const std::vector< V > & vals )  [explicit]
```

Create a Shape object from a vector of values

Template Parameters

V	Scalar type of the values

Parameters

vals	The dimensions for the object

6.31.1.3 Shape() [3/6]

```
template<typename T = size_t, size_t N = 32>
librapid::Shape< T, N >::Shape (
            const Shape< T, N > & other )  [default]
```

Create a copy of a Shape object

Parameters

other	Shape object to copy

6.31.1.4 Shape() [4/6]

```
template<typename T = size_t, size_t N = 32>
librapid::Shape< T, N >::Shape (
            Shape< T, N > && other )  [default], [noexcept]
```

Create a Shape from an RValue

Parameters

other	Temporary Shape object to copy

6.31.1.5 Shape() [5/6]

```
template<typename T , size_t N>
template<typename V , size_t Dim>
librapid::Shape< T, N >::Shape (
            const Shape< V, Dim > & other )
```

Create a Shape object from one with a different type and number of dimensions.

Template Parameters

V	Scalar type of the values
Dim	Number of dimensions

Parameters

other	Shape object to copy

6.31.1.6 Shape() [6/6]

```
template<typename T , size_t N>
template<typename V , size_t Dim>
librapid::Shape< T, N >::Shape (
            Shape< V, Dim > && other )  [noexcept]
```

Create a Shape object from one with a different type and number of dimensions, moving it instead of copying it.

Template Parameters

V	Scalar type of the values
Dim	Number of dimensions

Parameters

other	Temporary Shape object to move

6.31.2 Member Function Documentation

6.31.2.1 ndim()

```
template<typename T , size_t N>
LIBRAPID_NODISCARD T librapid::Shape< T, N >::ndim
```

Return the number of dimensions in the Shape object

Returns

Number of dimensions

6.31.2.2 ones()

```
template<typename T , size_t N>
Shape< T, N > librapid::Shape< T, N >::ones (
            size_t dims )  [static]
```

Return a Shape object with `dims` dimensions, all initialized to one.

Parameters

dims	Number of dimensions

Returns

New Shape object

6.31.2.3 operator"!=()

```
template<typename T , size_t N>
LIBRAPID_ALWAYS_INLINE bool librapid::Shape< T, N >::operator!= (
            const Shape< T, N > & other ) const
```

Compare two Shape objects, returning true if and only if they are not identical

Parameters

other	Shape object to compare

Returns

true if the objects are not identical

6.31.2.4 operator=() [1/3]

```
template<typename T = size_t, size_t N = 32>
template<typename V , typename typetraits::EnableIf< typetraits::CanCast< V, T >::value > =
0>
Shape & librapid::Shape< T, N >::operator= (
            const std::initializer_list< V > & vals )
```

Assign a Shape object to this object

Template Parameters

V	Scalar type of the Shape

Parameters

vals	Dimensions of the Shape

Returns

*this

6.31.2.5 operator=() [2/3]

```
template<typename T = size_t, size_t N = 32>
template<typename V , typename typetraits::EnableIf< typetraits::CanCast< V, T >::value > =
0>
Shape & librapid::Shape< T, N >::operator= (
            const std::vector< V > & vals )
```

Assign a Shape object to this object

Template Parameters

V	Scalar type of the Shape

Parameters

vals	Dimensions of the Shape

Returns

*this

6.31.2.6 operator=() [3/3]

```
template<typename T = size_t, size_t N = 32>
Shape & librapid::Shape< T, N >::operator= (
            Shape< T, N > && other )  [default], [noexcept]
```

Assign an RValue Shape to this object

Parameters

other	RValue to move

Returns

6.31.2.7 operator==()

```
template<typename T , size_t N>
LIBRAPID_ALWAYS_INLINE bool librapid::Shape< T, N >::operator== (
            const Shape< T, N > & other ) const
```

Compare two Shape objects, returning true if and only if they are identical

Parameters

other	Shape object to compare

Returns

true if the objects are identical

6.31.2.8 operator[]() [1/2]

```
template<typename T , size_t N>
template<typename Index >
LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE T & librapid::Shape< T, N >::operator[] (
            Index index )
```

Access an element of the Shape object

Template Parameters

Index	Typename of the index

Parameters

index	Index to access

Returns

A reference to the value at the index

6.31.2.9 operator[]() [2/2]

```
template<typename T , size_t N>
template<typename Index >
LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE const T & librapid::Shape< T, N >::operator[] (
            Index index ) const
```

Access an element of the Shape object

Template Parameters

Index	Typename of the index

Parameters

index	Index to access

Returns

The value at the index

6.31.2.10 size()

```
template<typename T , size_t N>
LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE T librapid::Shape< T, N >::size
```

Return the number of elements the Shape object represents

Returns

Number of elements

6.31.2.11 str()

```
template<typename T , size_t N>
std::string librapid::Shape< T, N >::str
```

Convert a Shape object into a string representation

Returns

A string representation of the Shape object

6.31.2.12 zeros()

```
template<typename T , size_t N>
Shape< T, N > librapid::Shape< T, N >::zeros (
            size_t dims ) [static]
```

Return a Shape object with `dims` dimensions, all initialized to zero.

Parameters

dims	Number of dimensions

Returns

New Shape object

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

- librapid/include/librapid/array/sizetype.hpp

6.32 sharedMemoryInfo_st Struct Reference

Public Attributes

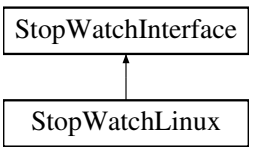
- void * **addr**
- size_t **size**
- int **shmFd**

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

- librapid/include/librapid/cuda/helper_multiprocess.h

6.33 StopWatchInterface Class Reference

Inheritance diagram for StopWatchInterface:



Public Member Functions

- virtual void start ()=0

  Start time measurement.
- virtual void stop ()=0

  Stop time measurement.
- virtual void reset ()=0

  Reset time counters to zero.
- virtual float getTime ()=0
- virtual float getAverageTime ()=0

6.33.1 Member Function Documentation

6.33.1.1 getAverageTime()

```
virtual float StopWatchInterface::getAverageTime ( )  [pure virtual]
```

Mean time to date based on the number of times the stopwatch has been *stopped* (ie finished sessions) and the current total time

Implemented in StopWatchLinux.

6.33.1.2 getTime()

```
virtual float StopWatchInterface::getTime ( )  [pure virtual]
```

Time in msec. after start. If the stop watch is still running (i.e. there was no call to stop()) then the elapsed time is returned, otherwise the time between the last start() and stop call is returned

Implemented in StopWatchLinux.

6.33.1.3 reset()

```
virtual void StopWatchInterface::reset ( )  [pure virtual]
```

Reset time counters to zero.

Implemented in StopWatchLinux.

6.33.1.4 start()

```
virtual void StopWatchInterface::start ( )  [pure virtual]
```

Start time measurement.

Implemented in StopWatchLinux.

6.33.1.5 stop()

```
virtual void StopWatchInterface::stop ( )  [pure virtual]
```

Stop time measurement.

Implemented in StopWatchLinux.

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

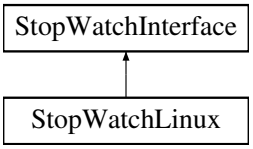
- librapid/include/librapid/cuda/helper_timer.h

6.34 StopWatchLinux Class Reference

Windows specific implementation of StopWatch.

```
#include <helper_timer.h>
```

Inheritance diagram for StopWatchLinux:



Public Member Functions

- **StopWatchLinux** ()
  Constructor, default.
- void start ()
  Start time measurement.
- void stop ()
  Stop time measurement.
- void reset ()
  Reset time counters to zero.
- float getTime ()
- float getAverageTime ()

6.34.1 Detailed Description

Windows specific implementation of StopWatch.

6.34.2 Member Function Documentation

6.34.2.1 getAverageTime()

```
float StopWatchLinux::getAverageTime ( )  [inline], [virtual]
```

Mean time to date based on the number of times the stopwatch has been *stopped* (ie finished sessions) and the current total time

Time in msec. for a single run based on the total number of COMPLETED runs and the total time.

Implements StopWatchInterface.

6.34.2.2 getTime()

```
float StopWatchLinux::getTime ( )  [inline], [virtual]
```

Time in msec. after start. If the stop watch is still running (i.e. there was no call to stop()) then the elapsed time is returned, otherwise the time between the last start() and stop call is returned

Time in msec. after start. If the stop watch is still running (i.e. there was no call to stop()) then the elapsed time is returned added to the current diff_time sum, otherwise the current summed time difference alone is returned.

Implements StopWatchInterface.

6.34.2.3 reset()

```
void StopWatchLinux::reset ( )  [inline], [virtual]
```

Reset time counters to zero.

Reset the timer to 0. Does not change the timer running state but does recapture this point in time as the current start time if it is running.

Implements StopWatchInterface.

6.34.2.4 start()

```
void StopWatchLinux::start ( )  [inline], [virtual]
```

Start time measurement.

Implements StopWatchInterface.

6.34.2.5 stop()

```
void StopWatchLinux::stop ( )  [inline], [virtual]
```

Stop time measurement.

Stop time measurement and increment add to the current diff_time summation variable. Also increment the number of times this clock has been run.

Implements StopWatchInterface.

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

- librapid/include/librapid/cuda/helper_timer.h

6.35 librapid::Storage< Scalar_, Allocator_ > Class Template Reference

Public Types

- using **Allocator** = Allocator_
- using **Scalar** = Scalar_
- using **Pointer** = typename std::allocator_traits< Allocator >::pointer
- using **ConstPointer** = typename std::allocator_traits< Allocator >::const_pointer
- using **Reference** = Scalar &
- using **ConstReference** = const Scalar &
- using **SizeType** = typename std::allocator_traits< Allocator >::size_type
- using **DifferenceType** = typename std::allocator_traits< Allocator >::difference_type
- using **Iterator** = Pointer
- using **ConstIterator** = ConstPointer
- using **ReverseIterator** = std::reverse_iterator< Iterator >
- using **ConstReverseIterator** = std::reverse_iterator< ConstIterator >

Public Member Functions

- **Storage** ()=default

  Default constructor.
- LIBRAPID_ALWAYS_INLINE Storage (SizeType size, const Allocator &alloc=Allocator())
- LIBRAPID_ALWAYS_INLINE Storage (SizeType size, ConstReference value, const Allocator &alloc=Allocator())
- LIBRAPID_ALWAYS_INLINE Storage (const Storage &other, const Allocator &alloc=Allocator())
- LIBRAPID_ALWAYS_INLINE Storage (Storage &&other) noexcept
- template<typename V >
  LIBRAPID_ALWAYS_INLINE Storage (const std::initializer_list< V > &list, const Allocator &alloc=Allocator())
- template<typename V >
  LIBRAPID_ALWAYS_INLINE Storage (const std::vector< V > &vec, const Allocator &alloc=Allocator())
- LIBRAPID_ALWAYS_INLINE Storage & operator= (const Storage &other)
- LIBRAPID_ALWAYS_INLINE Storage & operator= (Storage &&other) noexcept
- ~**Storage** ()

  Free a Storage object.
- LIBRAPID_ALWAYS_INLINE void resize (SizeType newSize)
- LIBRAPID_ALWAYS_INLINE void resize (SizeType newSize, int)
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE SizeType **size** () const noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstReference **operator[ ]** (SizeType index) const
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Reference **operator[ ]** (SizeType index)
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Iterator **begin** () noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Iterator **end** () noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstIterator **begin** () const noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstIterator **end** () const noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstIterator **cbegin** () const noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstIterator **cend** () const noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ReverseIterator **rbegin** () noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ReverseIterator **rend** () noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstReverseIterator **rbegin** () const noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstReverseIterator **rend** () const noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstReverseIterator **crbegin** () const noexcept
- LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstReverseIterator **crend** () const noexcept
- template<typename V >
  Storage (const std::initializer_list< V > &list, const Allocator &alloc)
- template<typename V >
  Storage (const std::vector< V > &vector, const Allocator &alloc)
- template<typename P >
  void **initData** (P begin, P end)

6.35.1 Constructor & Destructor Documentation

6.35.1.1 Storage() [1/6]

```
template<typename T , typename A >
librapid::Storage< T, A >::Storage (
            SizeType size,
            const Allocator & alloc = Allocator() )  [explicit]
```

Create a Storage object with `size` elements and, optionally, a custom allocator.

Parameters

size	Number of elements to allocate
alloc	Allocator to use

6.35.1.2 Storage() [2/6]

```
template<typename T , typename A >
librapid::Storage< T, A >::Storage (
            SizeType size,
            ConstReference value,
            const Allocator & alloc = Allocator() )
```

Create a Storage object with `size` elements, each initialized to `value`. Optionally, a custom allocator can be used.

Parameters

size	Number of elements to allocate
value	Value to initialize each element to
alloc	Allocator to use

6.35.1.3 Storage() [3/6]

```
template<typename T , typename A >
librapid::Storage< T, A >::Storage (
            const Storage< Scalar_, Allocator_ > & other,
            const Allocator & alloc = Allocator() )
```

Create a Storage object from another Storage object. Additionally a custom allocator can be used.

Parameters

other	Storage object to copy
alloc	Allocator to use

6.35.1.4 Storage() [4/6]

```
template<typename T , typename A >
librapid::Storage< T, A >::Storage (
            Storage< Scalar_, Allocator_ > && other )  [noexcept]
```

Move a Storage object into this object.

Parameters

other	Storage object to move

6.35.1.5 Storage() [5/6]

```
template<typename Scalar_ , typename Allocator_ = std::allocator<Scalar_>>
template<typename V >
LIBRAPID_ALWAYS_INLINE librapid::Storage< Scalar_, Allocator_ >::Storage (
            const std::initializer_list< V > & list,
            const Allocator & alloc = Allocator() )
```

Create a Storage object from an std::initializer_list

Template Parameters

V	Type of the elements in the initializer list

Parameters

list	Initializer list to copy
alloc	Allocator to use

6.35.1.6 Storage() [6/6]

```
template<typename Scalar_ , typename Allocator_ = std::allocator<Scalar_>>
template<typename V >
LIBRAPID_ALWAYS_INLINE librapid::Storage< Scalar_, Allocator_ >::Storage (
            const std::vector< V > & vec,
            const Allocator & alloc = Allocator() )  [explicit]
```

Create a Storage object from a std::vector

Template Parameters

V	Type of the elements in the vector

Parameters

vec	Vector to copy
alloc	Allocator to use

6.35.2 Member Function Documentation

6.35.2.1 operator=() [1/2]

```
template<typename T , typename A >
Storage< T, A > & librapid::Storage< T, A >::operator= (
            const Storage< Scalar_, Allocator_ > & other )
```

Assignment operator for a Storage object

Parameters

other	Storage object to copy

Returns

∗this

6.35.2.2 operator=() [2/2]

```
template<typename T , typename A >
Storage< T, A > & librapid::Storage< T, A >::operator= (
            Storage< Scalar_, Allocator_ > && other )  [noexcept]
```

Move assignment operator for a Storage object

Parameters

other	Storage object to move

Returns

∗this

6.35.2.3 resize() [1/2]

```
template<typename T , typename A >
void librapid::Storage< T, A >::resize (
            SizeType newSize )
```

Resize a Storage object to `size` elements. Existing elements are preserved.

Parameters

size	New size of the Storage object

6.35.2.4 resize() [2/2]

```
template<typename T , typename A >
void librapid::Storage< T, A >::resize (
             SizeType newSize,
             int  )
```

Resize a Storage object to `size` elements. Existing elements are not preserved

Parameters

size	New size of the Storage object

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

- librapid/include/librapid/array/storage.hpp

6.36 testOpts Struct Reference

Public Attributes

- char * **sparse_mat_filename**
- const char * **testFunc**
- const char * **reorder**
- int **lda**

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

- librapid/include/librapid/cuda/helper_cusolver.h

6.37 librapid::typetraits::TypeInfo< T > Struct Template Reference

```
#include <traits.hpp>
```

Public Types

- using **Scalar** = T
- using **Packet** = std::false_type

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = 1
- static constexpr char **name** [ ] = "[NO DEFINED TYPE]"
- static constexpr bool **supportsArithmetic** = true
- static constexpr bool **supportsLogical** = true
- static constexpr bool **supportsBinary** = true
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

6.37.1 Detailed Description

template<typename T>
struct librapid::typetraits::TypeInfo< T >

Provides compile-time information about a data type, allowing for easier function switching and compile-time evaluation

Template Parameters

T	The type to get information about

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

- librapid/include/librapid/core/traits.hpp

6.38 librapid::typetraits::TypeInfo< ArrayContainer< ShapeType_, StorageType_ > > Struct Template Reference

Public Types

- using **Scalar** = typename TypeInfo< StorageType_ >::Scalar

Static Public Attributes

- static constexpr bool **isLibRapidType** = true

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

- librapid/include/librapid/array/arrayContainer.hpp

6.39 librapid::typetraits::TypeInfo< bool > Struct Reference

Public Types

- using **Scalar** = bool
- using **Packet** = std::false_type

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = 1
- static constexpr char **name** [ ] = "char"
- static constexpr bool **supportsArithmetic** = false
- static constexpr bool **supportsLogical** = false
- static constexpr bool **supportsBinary** = true
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.40 librapid::typetraits::TypeInfo< char > Struct Reference

Public Types

- using **Scalar** = char
- using **Packet** = std::false_type

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = 1
- static constexpr char **name** [ ] = "bool"
- static constexpr bool **supportsArithmetic** = false
- static constexpr bool **supportsLogical** = false
- static constexpr bool **supportsBinary** = true
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.41 librapid::typetraits::TypeInfo< double > Struct Reference

Public Types

- using **Scalar** = double
- using **Packet** = Vc::Vector< double >

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = Packet::size()
- static constexpr char **name** [ ] = "double"
- static constexpr bool **supportsArithmetic** = true
- static constexpr bool **supportsLogical** = true
- static constexpr bool **supportsBinary** = false
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.42 librapid::typetraits::TypeInfo< float > Struct Reference

Public Types

- using **Scalar** = float
- using **Packet** = Vc::Vector< float >

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = Packet::size()
- static constexpr char **name** [ ] = "float"
- static constexpr bool **supportsArithmetic** = true
- static constexpr bool **supportsLogical** = true
- static constexpr bool **supportsBinary** = false
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.43 librapid::typetraits::TypeInfo< int16_t > Struct Reference

Public Types

- using **Scalar** = int16_t
- using **Packet** = Vc::Vector< int16_t >

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = Packet::size()
- static constexpr char **name** [ ] = "int16_t"
- static constexpr bool **supportsArithmetic** = true
- static constexpr bool **supportsLogical** = true
- static constexpr bool **supportsBinary** = true
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.44 librapid::typetraits::TypeInfo< int32_t > Struct Reference

Public Types

- using **Scalar** = int32_t
- using **Packet** = Vc::Vector< int32_t >

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = Packet::size()
- static constexpr char **name** [ ] = "int32_t"
- static constexpr bool **supportsArithmetic** = true
- static constexpr bool **supportsLogical** = true
- static constexpr bool **supportsBinary** = true
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.45 librapid::typetraits::TypeInfo< int64_t > Struct Reference

Public Types

- using **Scalar** = int64_t
- using **Packet** = Vc::Vector< int64_t >

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = Packet::size()
- static constexpr char **name** [ ] = "int64_t"
- static constexpr bool **supportsArithmetic** = true
- static constexpr bool **supportsLogical** = true
- static constexpr bool **supportsBinary** = true
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.46 librapid::typetraits::TypeInfo< int8_t > Struct Reference

Public Types

- using **Scalar** = int8_t
- using **Packet** = Vc::Vector< int8_t >

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = Packet::size()
- static constexpr char **name** [ ] = "int8_t"
- static constexpr bool **supportsArithmetic** = true
- static constexpr bool **supportsLogical** = true
- static constexpr bool **supportsBinary** = true
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.47 librapid::typetraits::TypeInfo< Storage< Scalar_, Allocator_ > > Struct Template Reference

Public Types

- using **Scalar** = Scalar_

Static Public Attributes

- static constexpr bool **isLibRapidType** = true

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

- librapid/include/librapid/array/storage.hpp

6.48 librapid::typetraits::TypeInfo< uint16_t > Struct Reference

Public Types

- using **Scalar** = uint16_t
- using **Packet** = Vc::Vector< uint16_t >

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = Packet::size()
- static constexpr char **name** [ ] = "uint16_t"
- static constexpr bool **supportsArithmetic** = true
- static constexpr bool **supportsLogical** = true
- static constexpr bool **supportsBinary** = true
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.49 librapid::typetraits::TypeInfo< uint32_t > Struct Reference

Public Types

- using **Scalar** = uint32_t
- using **Packet** = Vc::Vector< uint32_t >

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = Packet::size()
- static constexpr char **name** [ ] = "uint32_t"
- static constexpr bool **supportsArithmetic** = true
- static constexpr bool **supportsLogical** = true
- static constexpr bool **supportsBinary** = true
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.50 librapid::typetraits::TypeInfo< uint64_t > Struct Reference

Public Types

- using **Scalar** = uint64_t
- using **Packet** = Vc::Vector< uint64_t >

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = Packet::size()
- static constexpr char **name** [ ] = "uint64_t"
- static constexpr bool **supportsArithmetic** = true
- static constexpr bool **supportsLogical** = true
- static constexpr bool **supportsBinary** = true
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.51 librapid::typetraits::TypeInfo< uint8_t > Struct Reference

Public Types

- using **Scalar** = uint8_t
- using **Packet** = Vc::Vector< uint8_t >

Static Public Attributes

- static constexpr bool **isLibRapidType** = false
- static constexpr int64_t **packetWidth** = Packet::size()
- static constexpr char **name** [ ] = "uint8_t"
- static constexpr bool **supportsArithmetic** = true
- static constexpr bool **supportsLogical** = true
- static constexpr bool **supportsBinary** = true
- static constexpr bool **canAlign** = true
- static constexpr bool **canMemcpy** = true

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

- librapid/include/librapid/core/traits.hpp

6.52 librapid::typetraits::TypeInfo<::librapid::detail::Function< Functor_, Args... > > Struct Template Reference

Public Types

- using **Scalar** = decltype(std::declval< Functor_ >()(std::declval< typename TypeInfo< std::decay_t< Args > >::Scalar >()...))

Static Public Attributes

- static constexpr bool **isLibRapidType** = true
- static constexpr bool **supportsArithmetic** = TypeInfo<Scalar>::supportsArithmetic
- static constexpr bool **supportsLogical** = TypeInfo<Scalar>::supportsLogical
- static constexpr bool **supportsBinary** = TypeInfo<Scalar>::supportsBinary

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

- librapid/include/librapid/array/function.hpp

6.53 librapid::typetraits::detail::TypeNameHolder< T > Struct Template Reference

Static Public Attributes

- static constexpr auto **value** = typeNameArray<T>()

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

- librapid/include/librapid/core/traits.hpp

Chapter 7

File Documentation

7.1 auxiliary.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_AUXILIARY_AUXILIARY_H
34 #define CXXBLAS_AUXILIARY_AUXILIARY_H 1
35
36 #include "cxxblas/auxiliary/complex.h"
37 #include "cxxblas/auxiliary/complextrait.h"
38 #include "cxxblas/auxiliary/debugmacro.h"
39 #include "cxxblas/auxiliary/fakeuse.h"
40 #include "cxxblas/auxiliary/iscomplex.h"
41 #include "cxxblas/auxiliary/ismpfrreal.h"
42 #include "cxxblas/auxiliary/issame.h"
43 #include "cxxblas/auxiliary/pow.h"
44 #include "cxxblas/auxiliary/restrictto.h"
45
46 #endif // CXXBLAS_AUXILIARY_AUXILIARY_H
```

7.2 complex.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
```

```
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_AUXILIARY_COMPLEX_H
34 #define CXXBLAS_AUXILIARY_COMPLEX_H 1
35
36 #include "cxxblas/typedefs.h"
37 #include "cxxblas/auxiliary/restrictto.h"
38
39 namespace cxxblas {
40
41     template<typename T>
42     typename cxxblas::RestrictTo<std::is_arithmetic<T>::value, const T &>::Type
43     conjugate(const T &x);
44
45     template<typename T>
46     typename cxxblas::RestrictTo<std::is_arithmetic<T>::value, std::complex<T»::Type
47     conjugate(const std::complex<T> &x);
48
49     template<typename T>
50     typename cxxblas::RestrictTo<std::is_arithmetic<T>::value, const T &>::Type real(const T &x);
51
52     template<typename T>
53     typename cxxblas::RestrictTo<std::is_arithmetic<T>::value, const T>::Type
54     real(const std::complex<T> &x);
55
56     template<typename T>
57     typename cxxblas::RestrictTo<std::is_arithmetic<T>::value, const T>::Type imag(const T &x);
58
59     template<typename T>
60     typename cxxblas::RestrictTo<std::is_arithmetic<T>::value, const T>::Type
61     imag(const std::complex<T> &x);
62
63     template<typename T>
64     T abs1(const std::complex<T> &x);
65
66 } // namespace cxxblas
67
68 #endif // CXXBLAS_AUXILIARY_COMPLEX_H
```

7.3 complextrait.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
```

```
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_AUXILIARY_COMPLEXTRAIT_H
34 #define CXXBLAS_AUXILIARY_COMPLEXTRAIT_H 1
35
36 #include <complex>
37
38 namespace cxxblas {
39
40     template<typename T>
41     struct ComplexTrait {
42         typedef T PrimitiveType;
43     };
44
45     template<typename T>
46     struct ComplexTrait<std::complex<T>> {
47         typedef T PrimitiveType;
48     };
49
50 } // namespace cxxblas
51
52 #endif // CXXBLAS_AUXILIARY_COMPLEXTRAIT_H
```

7.4 debugmacro.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_AUXILIARY_DEBUGMACRO_H
34 #define CXXBLAS_AUXILIARY_DEBUGMACRO_H 1
35
36 #include <iostream>
37
38 //-- CXXBLAS_DEBUG_OUT ----------------------------------------------------------
39 #ifdef CXXBLAS_DEBUG
40 #   ifndef CXXBLAS_DEBUG_OUT
41 #       define CXXBLAS_DEBUG_OUT(msg) std::cerr « "CXXBLAS: " « msg « std::endl
42 #   endif // CXXBLAS_DEBUG_OUT
43 #else
44 #   ifndef CXXBLAS_DEBUG_OUT
45 #       define CXXBLAS_DEBUG_OUT(msg)
```

```
46 #   endif // CXXBLAS_DEBUG_OUT
47 #endif     // CXXBLAS_DEBUG
48
49 #include <cassert>
50
51 //-- ASSERT ------------------------------------------------------------------
52
53 // ASSERT which prints out a trace back of the call
54 #if defined(TRACEBACK_ASSERT) && !defined(NDEBUG)
55 #   include <execinfo.h>
56 #   ifndef ASSERT
57 #       define ASSERT(x)                                                                    \
58             if (!(x)) {                                                                     \
59                 void *callstack[128];                                                       \
60                 int frames  = backtrace(callstack, 128);                                    \
61                 char **strs = backtrace_symbols(callstack, frames);                         \
62                 for (int i = 0; i < frames; ++i) { std::cerr << strs[i] << std::endl; }     \
63                 free(strs);                                                                 \
64             }                                                                               \
65             assert(x);
66 #   endif
67 #endif
68
69 // Default ASSERT Macro
70 #ifndef ASSERT
71 #   define ASSERT(x) assert(x)
72 #endif
73
74 // Prevent warings because some function parameters are only used in debug
75 // mode within assertions.  In non-debug mode this causes warnings because
76 // of unused variables.
77
78 #ifndef NDEBUG
79
80 #   ifndef DEBUG_VAR
81 #       define DEBUG_VAR(x) x
82 #   endif
83
84 #   ifndef FAKE_USE_NDEBUG
85 #       define FAKE_USE_NDEBUG(x)
86 #   endif
87
88 #else
89
90 #   ifndef FAKE_USE_NDEBUG
91 #       define FAKE_USE_NDEBUG(x) (void)x
92 #   endif
93
94 #endif
95
96 #endif // CXXBLAS_AUXILIARY_DEBUGMACRO_H
```

7.5 fakeuse.h

```
1 /*
2  *   Copyright (c) 2013, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
```

```
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_AUXILIARY_FAKEUSE_H
34 #define CXXBLAS_AUXILIARY_FAKEUSE_H 1
35
36 #ifndef FAKE_USE
37 #   define FAKE_USE(X) (void)X
38 #endif
39
40 #endif // CXXBLAS_AUXILIARY_FAKEUSE_H
```

7.6 iscomplex.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_AUXILIARY_ISCOMPLEX_H
34 #define CXXBLAS_AUXILIARY_ISCOMPLEX_H 1
35
36 #include <complex>
37
38 namespace cxxblas {
39
40     template<typename T>
41     struct IsNotComplex {
42         static const bool value = true;
43     };
44
45     template<typename T>
46     struct IsNotComplex<std::complex<T>> {
47         static const bool value = false;
48     };
49
50     template<typename T>
51     struct IsComplex {
52         static const bool value = !IsNotComplex<T>::value;
53     };
54
55 } // namespace cxxblas
56
57 #endif // CXXBLAS_AUXILIARY_ISCOMPLEX_H
```

7.7 ismpfrreal.h

```
1 /*
2  *   Copyright (c) 2014, Michael Lehn
3  *
4  *   All rights reserved.
5  *
```

```
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *    THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *    "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *    LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *    A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *    OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *    SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *    LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *    DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *    THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *    (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *    OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_AUXILIARY_ISMPFRREAL_H
34 #define CXXBLAS_AUXILIARY_ISMPFRREAL_H 1
35
36 #ifdef WITH_MPFR
37
38 #   include <complex>
39 #   include <external/real.hpp>
40
41 namespace cxxblas {
42
43     template<typename T>
44     struct IsMpfrReal {
45         static const bool value = false;
46     };
47
48     template<mpfr::real_prec_t prec, mpfr::real_rnd_t rnd>
49     struct IsMpfrReal<mpfr::real<prec, rnd» {
50         static const bool value = true;
51     };
52
53 } // namespace cxxblas
54
55 #endif // WITH_MPFR
56
57 #endif // CXXBLAS_AUXILIARY_ISMPFRREAL_H
```

7.8 issame.h

```
1  /*
2  *   Copyright (c) 2004, Alexander Stippler
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *    THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *    "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *    LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *    A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *    OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *    SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *    LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *    DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *    THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
```

```
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_AUXILIARY_ISSAME_H
34 #define CXXBLAS_AUXILIARY_ISSAME_H 1
35
36 namespace cxxblas {
37
38     template<typename... Args>
39     struct IsSame {
40         static const bool value = false;
41     };
42
43     template<typename T>
44     struct IsSame<T> {
45         static const bool value = true;
46     };
47
48     template<typename T>
49     struct IsSame<T, T> {
50         static const bool value = true;
51     };
52
53     template<typename T, typename... Args>
54     struct IsSame<T, T, Args...> {
55         static const bool value = IsSame<T, Args...>::value;
56     };
57
58 } // namespace cxxblas
59
60 #endif // CXXBLAS_AUXILIARY_ISSAME_H
```

7.9 pow.h

```
1  /*
2   *   Copyright (c) 2014, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_AUXILIARY_POW_H
34 #define CXXBLAS_AUXILIARY_POW_H 1
35
36 #include "cxxblas/auxiliary/iscomplex.h"
37 #include "cxxblas/auxiliary/ismpfrreal.h"
38 #include "cxxblas/auxiliary/issame.h"
39 #include "cxxblas/auxiliary/restrictto.h"
40
41 #ifdef WITH_MPFR
42 #   include <external/real.hpp>
43 #endif
44
45 namespace cxxblas {
46
47     template<typename T>
48     typename RestrictTo<IsSame<T, int>::value, T>::Type pow(const T &base, const T &exponent);
```

```
49
50 #ifdef WITH_MPFR
51     template<typename T>
52     typename RestrictTo<!IsSame<T, int>::value && !IsComplex<T>::value && !IsMpfrReal<T>::value,
53                         T>::Type
54     pow(const T &base, int exponent);
55 #else
56     template<typename T>
57     typename RestrictTo<!IsSame<T, int>::value && !IsComplex<T>::value, T>::Type pow(const T &base,
58                                                                                      int exponent);
59 #endif
60
61     template<typename T>
62     std::complex<T> pow(const std::complex<T> &base, int exponent);
63
64 } // namespace cxxblas
65
66 #endif // CXXBLAS_AUXILIARY_POW_H
```

7.10 restrictto.h

```
1 /*
2  *   Copyright (c) 2004, Alexander Stippler
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_AUXILIARY_RESTRICTTO_H
34 #define CXXBLAS_AUXILIARY_RESTRICTTO_H 1
35
36 #ifdef INCLUDE_TYPE_TRAITS
37 #   include <type_traits>
38 #endif
39
40 namespace cxxblas {
41
42     template<bool b, typename T>
43     struct RestrictTo {};
44
45     template<typename T>
46     struct RestrictTo<true, T> {
47         typedef typename std::remove_reference<T>::type Type;
48     };
49
50 } // namespace cxxblas
51
52 #endif // CXXBLAS_AUXILIARY_RESTRICTTO_H
```

7.11 cxxblas.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
```

```
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_CXXBLAS_H
34 #define CXXBLAS_CXXBLAS_H 1
35
36 // Ensure LibRapid is already present
37 #include "librapid/core/core.hpp"
38
39 #include "cxxblas/auxiliary/auxiliary.h"
40 #include "cxxblas/drivers/drivers.h"
41 #include "cxxblas/typedefs.h"
42
43 #include "cxxblas/level1/level1.h"
44 #include "cxxblas/level1extensions/level1extensions.h"
45 #include "cxxblas/level2/level2.h"
46 #include "cxxblas/level2extensions/level2extensions.h"
47 #include "cxxblas/level3/level3.h"
48 #include "cxxblas/level3extensions/level3extensions.h"
49
50 #include "cxxblas/sparselevel2/sparselevel2.h"
51 #include "cxxblas/sparselevel3/sparselevel3.h"
52
53 #include "cxxblas/tinylevel1/tinylevel1.h"
54 #include "cxxblas/tinylevel2/tinylevel2.h"
55
56 #endif // CXXBLAS_CXXBLAS_H
```

7.12 atlas.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
```

```
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_DRIVERS_ATLAS_H
34 #define CXXBLAS_DRIVERS_ATLAS_H 1
35
36 #define HAVE_CBLAS 1
37 #define CBLAS_INT  int
38 #define BLAS_IMPL  "ATLAS"
39 #ifndef CBLAS_INDEX
40 #   define CBLAS_INDEX int
41 #endif // CBLAS_INDEX
42
43 // BLAS extensions
44 #ifndef HAVE_CBLAS_AXPBY
45 #   define HAVE_CBLAS_AXPBY
46 #   define BLAS_EXT(x) catlas_##x
47 #endif
48
49 #endif // CXXBLAS_DRIVERS_ATLAS_H
```

7.13 cblas.h

```
1  /*
2   *   File taken (with minor modifications) from cblas:
3   *     http://www.netlib.org/blas/
4   */
5
6  #ifndef CXXBLAS_DRIVERS_CBLAS_H
7  #define CXXBLAS_DRIVERS_CBLAS_H 1
8
9  #ifndef CBLAS_INT
10 #   define CBLAS_INT int
11 #endif // CBLAS_INT
12
13 #ifndef CBLAS_INDEX
14 #   define CBLAS_INDEX int
15 #endif // CBLAS_INT
16
17 #ifdef __cplusplus
18 extern "C" {
19 #endif
20
21 enum CBLAS_ORDER { CblasRowMajor = 101, CblasColMajor = 102 };
22 enum CBLAS_TRANSPOSE {
23     CblasNoTrans     = 111,
24     CblasTrans       = 112,
25     CblasConjTrans   = 113,
26     CblasConjNoTrans = 114
27 };
28 enum CBLAS_UPLO { CblasUpper = 121, CblasLower = 122 };
29 enum CBLAS_DIAG { CblasNonUnit = 131, CblasUnit = 132 };
30 enum CBLAS_SIDE { CblasLeft = 141, CblasRight = 142 };
31
32 //-- LEVEL 1 ---------------------------------------------------------------
33
34 // asum
35 float cblas_sasum(CBLAS_INT n, const float *x, CBLAS_INT incX);
36
37 double cblas_dasum(CBLAS_INT n, const double *x, CBLAS_INT incX);
38
39 float cblas_scasum(CBLAS_INT n, const float *x, CBLAS_INT incX);
40
41 double cblas_dzasum(CBLAS_INT n, const double *x, CBLAS_INT incX);
42
43 // axpy
44 void cblas_saxpy(CBLAS_INT n, float alpha, const float *x, CBLAS_INT incX, float *y,
45                  CBLAS_INT incY);
46
47 void cblas_daxpy(CBLAS_INT n, double alpha, const double *x, CBLAS_INT incX, double *y,
48                  CBLAS_INT incY);
49
50 void cblas_caxpy(CBLAS_INT n, const float *alpha, const float *x, CBLAS_INT incX, float *y,
51                  CBLAS_INT incY);
52
53 void cblas_zaxpy(CBLAS_INT n, const double *alpha, const double *x, CBLAS_INT incX, double *y,
54                  CBLAS_INT incY);
55
56 // axpby
57 #ifdef HAVE_CBLAS_AXPBY
58
```

```
59 void BLAS_EXT(saxpby)(CBLAS_INT n, float alpha, const float *x, CBLAS_INT incX, float beta,
60                        float *y, CBLAS_INT incY);
61
62 void BLAS_EXT(daxpby)(CBLAS_INT n, double alpha, const double *x, CBLAS_INT incX, double beta,
63                        double *y, CBLAS_INT incY);
64
65 void BLAS_EXT(caxpby)(CBLAS_INT n, const float *alpha, const float *x, CBLAS_INT incX,
66                        const float *beta, float *y, CBLAS_INT incY);
67
68 void BLAS_EXT(zaxpby)(CBLAS_INT n, const double *alpha, const double *x, CBLAS_INT incX,
69                        const double *beta, double *y, CBLAS_INT incY);
70
71 #endif // HAVE_CBLAS_AXPBY
72
73 // copy
74 void cblas_scopy(CBLAS_INT n, const float *x, CBLAS_INT incX, float *y, CBLAS_INT incY);
75
76 void cblas_dcopy(CBLAS_INT n, const double *x, CBLAS_INT incX, double *y, CBLAS_INT incY);
77
78 void cblas_ccopy(CBLAS_INT n, const float *x, CBLAS_INT incX, float *y, CBLAS_INT incY);
79
80 void cblas_zcopy(CBLAS_INT n, const double *x, CBLAS_INT incX, double *y, CBLAS_INT incY);
81
82 // dot
83 float cblas_sdsdot(CBLAS_INT n, float alpha, const float *x, CBLAS_INT incX, const float *y,
84                    CBLAS_INT incY);
85
86 double cblas_dsdot(CBLAS_INT n, const float *x, CBLAS_INT incX, const float *y, CBLAS_INT incY);
87
88 float cblas_sdot(CBLAS_INT n, const float *x, CBLAS_INT incX, const float *y, CBLAS_INT incY);
89
90 double cblas_ddot(CBLAS_INT n, const double *x, CBLAS_INT incX, const double *y, CBLAS_INT incY);
91
92 void cblas_cdotu_sub(CBLAS_INT n, const float *x, CBLAS_INT incX, const float *y, CBLAS_INT incY,
93                      float *result);
94
95 void cblas_cdotc_sub(CBLAS_INT n, const float *x, CBLAS_INT incX, const float *y, CBLAS_INT incY,
96                      float *result);
97
98 void cblas_zdotu_sub(CBLAS_INT n, const double *x, CBLAS_INT incX, const double *y, CBLAS_INT incY,
99                      double *result);
100
101 void cblas_zdotc_sub(CBLAS_INT n, const double *x, CBLAS_INT incX, const double *y, CBLAS_INT incY,
102                      double *result);
103
104 // iamax
105 CBLAS_INDEX
106 cblas_isamax(CBLAS_INT n, const float *x, CBLAS_INT incX);
107
108 CBLAS_INDEX
109 cblas_idamax(CBLAS_INT n, const double *x, CBLAS_INT incX);
110
111 CBLAS_INDEX
112 cblas_icamax(CBLAS_INT n, const float *x, CBLAS_INT incX);
113
114 CBLAS_INDEX
115 cblas_izamax(CBLAS_INT n, const double *x, CBLAS_INT incX);
116
117 // nrm2
118 float cblas_snrm2(CBLAS_INT n, const float *X, CBLAS_INT incX);
119
120 double cblas_dnrm2(CBLAS_INT n, const double *X, CBLAS_INT incX);
121
122 float cblas_scnrm2(CBLAS_INT n, const float *X, CBLAS_INT incX);
123
124 double cblas_dznrm2(CBLAS_INT n, const double *X, CBLAS_INT incX);
125
126 // rot
127 void cblas_srot(CBLAS_INT n, float *X, CBLAS_INT incX, float *Y, CBLAS_INT incY, float c, float s);
128
129 void cblas_drot(CBLAS_INT n, double *X, CBLAS_INT incX, double *Y, CBLAS_INT incY, double c,
130                 double s);
131
132 void cblas_srotg(float *a, float *b, float *c, float *s);
133
134 void cblas_drotg(double *a, double *b, double *c, double *s);
135
136 // rotm
137 void cblas_srotm(CBLAS_INT n, float *X, CBLAS_INT incX, float *Y, CBLAS_INT incY, const float *P);
138
139 void cblas_drotm(CBLAS_INT n, double *X, CBLAS_INT incX, double *Y, CBLAS_INT incY,
140                  const double *P);
141
142 void cblas_srotmg(float *d1, float *d2, float *b1, float *b2, float *P);
143
144 void cblas_drotmg(double *d1, double *d2, double *b1, double *b2, double *P);
145
```

```
146 // scal
147 void cblas_sscal(CBLAS_INT n, float alpha, float *x, CBLAS_INT incX);
148
149 void cblas_dscal(CBLAS_INT n, double alpha, double *x, CBLAS_INT incX);
150
151 void cblas_cscal(CBLAS_INT n, const float *alpha, float *x, CBLAS_INT incX);
152
153 void cblas_zscal(CBLAS_INT n, const double *alpha, double *x, CBLAS_INT incX);
154
155 void cblas_csscal(CBLAS_INT n, float alpha, float *x, CBLAS_INT incX);
156
157 void cblas_zdscal(CBLAS_INT n, double alpha, double *x, CBLAS_INT incX);
158
159 // swap
160 void cblas_sswap(CBLAS_INT n, float *x, CBLAS_INT incX, float *y, CBLAS_INT incY);
161
162 void cblas_dswap(CBLAS_INT n, double *x, CBLAS_INT incX, double *y, CBLAS_INT incY);
163
164 void cblas_cswap(CBLAS_INT n, float *x, CBLAS_INT incX, float *y, CBLAS_INT incY);
165
166 void cblas_zswap(CBLAS_INT n, double *x, CBLAS_INT incX, double *y, CBLAS_INT incY);
167
168 //-- LEVEL 2 ------------------------------------------------------------------
169
170 // gbmv
171 void cblas_sgbmv(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE trans, CBLAS_INT m, CBLAS_INT n,
172                  CBLAS_INT kl, CBLAS_INT ku, float alpha, const float *A, CBLAS_INT ldA,
173                  const float *x, CBLAS_INT incX, float beta, float *y, CBLAS_INT incY);
174
175 void cblas_dgbmv(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE trans, CBLAS_INT m, CBLAS_INT n,
176                  CBLAS_INT kl, CBLAS_INT ku, double alpha, const double *A, CBLAS_INT ldA,
177                  const double *x, CBLAS_INT incX, double beta, double *y, CBLAS_INT incY);
178
179 void cblas_cgbmv(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE trans, CBLAS_INT m, CBLAS_INT n,
180                  CBLAS_INT kl, CBLAS_INT ku, const float *alpha, const float *A, CBLAS_INT ldA,
181                  const float *x, CBLAS_INT incX, const float *beta, float *y, CBLAS_INT incY);
182
183 void cblas_zgbmv(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE trans, CBLAS_INT m, CBLAS_INT n,
184                  CBLAS_INT kl, CBLAS_INT ku, const double *alpha, const double *A, CBLAS_INT ldA,
185                  const double *x, CBLAS_INT incX, const double *beta, double *y, CBLAS_INT incY);
186
187 // gemv
188 void cblas_sgemv(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE trans, CBLAS_INT m, CBLAS_INT n,
189                  float alpha, const float *A, CBLAS_INT ldA, const float *x, CBLAS_INT incX,
190                  float beta, float *y, CBLAS_INT incY);
191
192 void cblas_dgemv(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE trans, CBLAS_INT m, CBLAS_INT n,
193                  double alpha, const double *A, CBLAS_INT ldA, const double *x, CBLAS_INT incX,
194                  double beta, double *y, CBLAS_INT incY);
195
196 void cblas_cgemv(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE trans, CBLAS_INT m, CBLAS_INT n,
197                  const float *alpha, const float *A, CBLAS_INT ldA, const float *x, CBLAS_INT incX,
198                  const float *beta, float *y, CBLAS_INT incY);
199
200 void cblas_zgemv(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE trans, CBLAS_INT m, CBLAS_INT n,
201                  const double *alpha, const double *A, CBLAS_INT ldA, const double *x,
202                  CBLAS_INT incX, const double *beta, double *y, CBLAS_INT incY);
203
204 // sbmv
205 void cblas_ssbmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, CBLAS_INT k,
206                  float alpha, const float *A, CBLAS_INT ldA, const float *x, CBLAS_INT incX,
207                  float beta, float *y, CBLAS_INT incY);
208
209 void cblas_dsbmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, CBLAS_INT k,
210                  double alpha, const double *A, CBLAS_INT ldA, const double *x, CBLAS_INT incX,
211                  double beta, double *y, CBLAS_INT incY);
212
213 // symv
214 void cblas_ssymv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, float alpha,
215                  const float *A, CBLAS_INT ldA, const float *x, CBLAS_INT incX, float beta,
216                  float *y, CBLAS_INT incY);
217
218 void cblas_dsymv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, double alpha,
219                  const double *A, CBLAS_INT ldA, const double *x, CBLAS_INT incX, double beta,
220                  double *y, CBLAS_INT incY);
221
222 // spmv
223 void cblas_sspmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, float alpha,
224                  const float *Ap, const float *x, CBLAS_INT incX, float beta, float *y,
225                  CBLAS_INT incY);
226
227 void cblas_dspmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, double alpha,
228                  const double *Ap, const double *x, CBLAS_INT incX, double beta, double *y,
229                  CBLAS_INT incY);
230
231 // hbmv
232 void cblas_chbmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, CBLAS_INDEX k,
```

```
233                 const float *alpha, const float *A, CBLAS_INT ldA, const float *x, CBLAS_INT incX,
234                 const float *beta, float *y, CBLAS_INT incY);
235
236 void cblas_zhbmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, CBLAS_INDEX k,
237                 const double *alpha, const double *A, CBLAS_INT ldA, const double *x,
238                 CBLAS_INT incX, const double *beta, double *y, CBLAS_INT incY);
239
240 // hemv
241 void cblas_chemv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, const float *alpha,
242                 const float *A, CBLAS_INT ldA, const float *x, CBLAS_INT incX, const float *beta,
243                 float *y, CBLAS_INT incY);
244
245 void cblas_zhemv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, const double *alpha,
246                 const double *A, CBLAS_INT ldA, const double *x, CBLAS_INT incX,
247                 const double *beta, double *y, CBLAS_INT incY);
248
249 // hpmv
250 void cblas_chpmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, const float *alpha,
251                 const float *Ap, const float *x, CBLAS_INT incX, const float *beta, float *y,
252                 CBLAS_INT incY);
253
254 void cblas_zhpmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, const double *alpha,
255                 const double *Ap, const double *x, CBLAS_INT incX, const double *beta, double *y,
256                 CBLAS_INT incY);
257
258 // tbsv
259 void cblas_stbsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
260                 enum CBLAS_DIAG diag, CBLAS_INT n, CBLAS_INT k, const float *A, CBLAS_INT lda,
261                 float *X, CBLAS_INT incX);
262
263 void cblas_dtbsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
264                 enum CBLAS_DIAG diag, CBLAS_INT n, CBLAS_INT k, const double *A, CBLAS_INT lda,
265                 double *X, CBLAS_INT incX);
266
267 void cblas_ctbsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
268                 enum CBLAS_DIAG diag, CBLAS_INT n, CBLAS_INT k, const float *A, CBLAS_INT lda,
269                 float *X, CBLAS_INT incX);
270
271 void cblas_ztbsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
272                 enum CBLAS_DIAG diag, CBLAS_INT n, CBLAS_INT k, const double *A, CBLAS_INT lda,
273                 double *X, CBLAS_INT incX);
274
275 // trsv
276 void cblas_strsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
277                 enum CBLAS_DIAG diag, CBLAS_INT n, const float *A, CBLAS_INT lda, float *X,
278                 CBLAS_INT incX);
279
280 void cblas_dtrsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
281                 enum CBLAS_DIAG diag, CBLAS_INT n, const double *A, CBLAS_INT lda, double *X,
282                 CBLAS_INT incX);
283
284 void cblas_ctrsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
285                 enum CBLAS_DIAG diag, CBLAS_INT n, const float *A, CBLAS_INT lda, float *X,
286                 CBLAS_INT incX);
287
288 void cblas_ztrsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
289                 enum CBLAS_DIAG diag, CBLAS_INT n, const double *A, CBLAS_INT lda, double *X,
290                 CBLAS_INT incX);
291
292 // tpsv
293 void cblas_stpsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
294                 enum CBLAS_DIAG diag, CBLAS_INT n, const float *A, float *X, CBLAS_INT incX);
295
296 void cblas_dtpsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
297                 enum CBLAS_DIAG diag, CBLAS_INT n, const double *A, double *X, CBLAS_INT incX);
298
299 void cblas_ctpsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
300                 enum CBLAS_DIAG diag, CBLAS_INT n, const float *A, float *X, CBLAS_INT incX);
301
302 void cblas_ztpsv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
303                 enum CBLAS_DIAG diag, CBLAS_INT n, const double *A, double *X, CBLAS_INT incX);
304
305 // tbmv
306 void cblas_stbmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
307                 enum CBLAS_DIAG diag, CBLAS_INT n, CBLAS_INDEX k, const float *A, CBLAS_INT lda,
308                 float *x, CBLAS_INT incX);
309
310 void cblas_dtbmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
311                 enum CBLAS_DIAG diag, CBLAS_INT n, CBLAS_INDEX k, const double *A, CBLAS_INT lda,
312                 double *x, CBLAS_INT incX);
313
314 void cblas_ctbmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
315                 enum CBLAS_DIAG diag, CBLAS_INT n, CBLAS_INDEX k, const float *A, CBLAS_INT lda,
316                 float *x, CBLAS_INT incX);
317
318 void cblas_ztbmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
319                 enum CBLAS_DIAG diag, CBLAS_INT N, CBLAS_INDEX k, const double *A, CBLAS_INT lda,
```

```
320                 double *x, CBLAS_INT incX);
321
322 // trmv
323 void cblas_strmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
324                 enum CBLAS_DIAG diag, CBLAS_INT n, const float *A, CBLAS_INT lda, float *x,
325                 CBLAS_INT incX);
326
327 void cblas_dtrmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
328                 enum CBLAS_DIAG diag, CBLAS_INT n, const double *A, CBLAS_INT lda, double *x,
329                 CBLAS_INT incX);
330
331 void cblas_ctrmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
332                 enum CBLAS_DIAG diag, CBLAS_INT n, const float *A, CBLAS_INT lda, float *x,
333                 CBLAS_INT incX);
334
335 void cblas_ztrmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
336                 enum CBLAS_DIAG diag, CBLAS_INT N, const double *A, CBLAS_INT lda, double *x,
337                 CBLAS_INT incX);
338
339 // tpmv
340 void cblas_stpmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
341                 enum CBLAS_DIAG diag, CBLAS_INT n, const float *Ap, float *x, CBLAS_INT incX);
342
343 void cblas_dtpmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
344                 enum CBLAS_DIAG diag, CBLAS_INT n, const double *Ap, double *x, CBLAS_INT incX);
345
346 void cblas_ctpmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
347                 enum CBLAS_DIAG diag, CBLAS_INT n, const float *Ap, float *x, CBLAS_INT incX);
348
349 void cblas_ztpmv(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE transA,
350                 enum CBLAS_DIAG diag, CBLAS_INT N, const double *Ap, double *x, CBLAS_INT incX);
351
352 // ger
353 void cblas_sger(enum CBLAS_ORDER order, CBLAS_INT m, CBLAS_INT n, float alpha, const float *X,
354                CBLAS_INT incX, const float *Y, CBLAS_INT incY, float *A, CBLAS_INT lda);
355
356 void cblas_dger(enum CBLAS_ORDER order, CBLAS_INT m, CBLAS_INT n, double alpha, const double *X,
357                CBLAS_INT incX, const double *Y, CBLAS_INT incY, double *A, CBLAS_INT lda);
358
359 void cblas_cgeru(enum CBLAS_ORDER order, CBLAS_INT m, CBLAS_INT n, const float *alpha,
360                 const float *X, CBLAS_INT incX, const float *Y, CBLAS_INT incY, float *A,
361                 CBLAS_INT lda);
362
363 void cblas_cgerc(enum CBLAS_ORDER order, CBLAS_INT m, CBLAS_INT n, const float *alpha,
364                 const float *X, CBLAS_INT incX, const float *Y, CBLAS_INT incY, float *A,
365                 CBLAS_INT lda);
366
367 void cblas_zgeru(enum CBLAS_ORDER order, CBLAS_INT m, CBLAS_INT n, const double *alpha,
368                 const double *X, CBLAS_INT incX, const double *Y, CBLAS_INT incY, double *A,
369                 CBLAS_INT lda);
370
371 void cblas_zgerc(enum CBLAS_ORDER order, CBLAS_INT m, CBLAS_INT n, const double *alpha,
372                 const double *X, CBLAS_INT incX, const double *Y, CBLAS_INT incY, double *A,
373                 CBLAS_INT lda);
374
375 // syr
376 void cblas_ssyr(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, float alpha,
377                const float *X, CBLAS_INT incX, float *A, CBLAS_INT lda);
378
379 void cblas_dsyr(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, double alpha,
380                const double *X, CBLAS_INT incX, double *A, CBLAS_INT lda);
381
382 // spr
383 void cblas_sspr(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, float alpha,
384                const float *X, CBLAS_INT incX, float *A);
385
386 void cblas_dspr(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, double alpha,
387                const double *X, CBLAS_INT incX, double *A);
388
389 // her
390 void cblas_cher(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, float alpha,
391                const float *X, CBLAS_INT incX, float *A, CBLAS_INT lda);
392
393 void cblas_zher(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, double alpha,
394                const double *X, CBLAS_INT incX, double *A, CBLAS_INT lda);
395
396 // hpr
397 void cblas_chpr(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, float alpha,
398                const float *X, CBLAS_INT incX, float *A);
399
400 void cblas_zhpr(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, double alpha,
401                const double *X, CBLAS_INT incX, double *A);
402
403 // spr2
404 void cblas_sspr2(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, float alpha,
405                 const float *X, CBLAS_INT incX, const float *Y, CBLAS_INT incY, float *A);
406
```

```
407 void cblas_dspr2(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, double alpha,
408                  const double *X, CBLAS_INT incX, const double *Y, CBLAS_INT incY, double *A);
409
410 // syr2
411 void cblas_ssyr2(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, float alpha,
412                  const float *X, CBLAS_INT incX, const float *Y, CBLAS_INT incY, float *A,
413                  CBLAS_INT lda);
414
415 void cblas_dsyr2(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, double alpha,
416                  const double *X, CBLAS_INT incX, const double *Y, CBLAS_INT incY, double *A,
417                  CBLAS_INT lda);
418 // her2
419 void cblas_cher2(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, const float *alpha,
420                  const float *X, CBLAS_INT incX, const float *Y, CBLAS_INT incY, float *A,
421                  CBLAS_INT lda);
422
423 void cblas_zher2(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, const double *alpha,
424                  const double *X, CBLAS_INT incX, const double *Y, CBLAS_INT incY, double *A,
425                  CBLAS_INT lda);
426 // hpr2
427 void cblas_chpr2(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, const float *alpha,
428                  const float *X, CBLAS_INT incX, const float *Y, CBLAS_INT incY, float *A);
429
430 void cblas_zhpr2(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, CBLAS_INT n, const double *alpha,
431                  const double *X, CBLAS_INT incX, const double *Y, CBLAS_INT incY, double *A);
432
433 //-- LEVEL 3 ------------------------------------------------------------------
434
435 // gemm
436 void cblas_sgemm(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE transA, enum CBLAS_TRANSPOSE transB,
437                  CBLAS_INT m, CBLAS_INT n, CBLAS_INT k, float alpha, const float *A, CBLAS_INT ldA,
438                  const float *B, CBLAS_INT ldB, float beta, float *C, CBLAS_INT ldC);
439
440 void cblas_dgemm(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE transA, enum CBLAS_TRANSPOSE transB,
441                  CBLAS_INT m, CBLAS_INT n, CBLAS_INT k, double alpha, const double *A,
442                  CBLAS_INT ldA, const double *B, CBLAS_INT ldB, double beta, double *C,
443                  CBLAS_INT ldC);
444
445 void cblas_cgemm(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE transA, enum CBLAS_TRANSPOSE transB,
446                  CBLAS_INT m, CBLAS_INT n, CBLAS_INT k, const float *alpha, const float *A,
447                  CBLAS_INT ldA, const float *B, CBLAS_INT ldB, const float *beta, float *C,
448                  CBLAS_INT ldC);
449
450 void cblas_zgemm(enum CBLAS_ORDER order, enum CBLAS_TRANSPOSE transA, enum CBLAS_TRANSPOSE transB,
451                  CBLAS_INT m, CBLAS_INT n, CBLAS_INT k, const double *alpha, const double *A,
452                  CBLAS_INT ldA, const double *B, CBLAS_INT ldB, const double *beta, double *C,
453                  CBLAS_INT ldC);
454
455 // hemm
456 void cblas_chemm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo, CBLAS_INT m,
457                  CBLAS_INT n, const float *alpha, const float *A, CBLAS_INT ldA, const float *B,
458                  CBLAS_INT ldB, const float *beta, float *C, CBLAS_INT ldC);
459
460 void cblas_zhemm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo, CBLAS_INT m,
461                  CBLAS_INT n, const double *alpha, const double *A, CBLAS_INT ldA, const double *B,
462                  CBLAS_INT ldB, const double *beta, double *C, CBLAS_INT ldC);
463
464 // herk
465 void cblas_cherk(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
466                  CBLAS_INT n, CBLAS_INT k, float alpha, const float *A, CBLAS_INT ldA, float beta,
467                  float *C, CBLAS_INT ldC);
468
469 void cblas_zherk(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
470                  CBLAS_INT n, CBLAS_INT k, double alpha, const double *A, CBLAS_INT ldA,
471                  double beta, double *C, CBLAS_INT ldC);
472
473 // her2k
474 void cblas_cher2k(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
475                   CBLAS_INT n, CBLAS_INT k, const float *alpha, const float *A, CBLAS_INT ldA,
476                   const float *B, CBLAS_INT ldB, float beta, float *C, CBLAS_INT ldC);
477
478 void cblas_zher2k(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
479                   CBLAS_INT n, CBLAS_INT k, const double *alpha, const double *A, CBLAS_INT ldA,
480                   const double *B, CBLAS_INT ldB, double beta, double *C, CBLAS_INT ldC);
481
482 // symm
483 void cblas_ssymm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo, CBLAS_INT m,
484                  CBLAS_INT n, float alpha, const float *A, CBLAS_INT ldA, const float *B,
485                  CBLAS_INT ldB, float beta, float *C, CBLAS_INT ldC);
486
487 void cblas_dsymm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo, CBLAS_INT m,
488                  CBLAS_INT n, double alpha, const double *A, CBLAS_INT ldA, const double *B,
489                  CBLAS_INT ldB, double beta, double *C, CBLAS_INT ldC);
490
491 void cblas_csymm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo, CBLAS_INT m,
492                  CBLAS_INT n, const float *alpha, const float *A, CBLAS_INT ldA, const float *B,
493                  CBLAS_INT ldB, const float *beta, float *C, CBLAS_INT ldC);
```

```
494
495 void cblas_zsymm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo, CBLAS_INT m,
496                  CBLAS_INT n, const double *alpha, const double *A, CBLAS_INT ldA, const double *B,
497                  CBLAS_INT ldB, const double *beta, double *C, CBLAS_INT ldC);
498
499 // syrk
500 void cblas_ssyrk(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
501                  CBLAS_INT n, CBLAS_INT k, float alpha, const float *A, CBLAS_INT ldA, float beta,
502                  float *C, CBLAS_INT ldC);
503
504 void cblas_dsyrk(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
505                  CBLAS_INT n, CBLAS_INT k, double alpha, const double *A, CBLAS_INT ldA,
506                  double beta, double *C, CBLAS_INT ldC);
507
508 void cblas_csyrk(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
509                  CBLAS_INT n, CBLAS_INT k, const float *alpha, const float *A, CBLAS_INT ldA,
510                  const float *beta, float *C, CBLAS_INT ldC);
511
512 void cblas_zsyrk(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
513                  CBLAS_INT n, CBLAS_INT k, const double *alpha, const double *A, CBLAS_INT ldA,
514                  const double *beta, double *C, CBLAS_INT ldC);
515
516 // syr2k
517 void cblas_ssyr2k(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
518                   CBLAS_INT n, CBLAS_INT k, float alpha, const float *A, CBLAS_INT ldA,
519                   const float *B, CBLAS_INT ldB, float beta, float *C, CBLAS_INT ldC);
520
521 void cblas_dsyr2k(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
522                   CBLAS_INT n, CBLAS_INT k, double alpha, const double *A, CBLAS_INT ldA,
523                   const double *B, CBLAS_INT ldB, double beta, double *C, CBLAS_INT ldC);
524
525 void cblas_csyr2k(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
526                   CBLAS_INT n, CBLAS_INT k, const float *alpha, const float *A, CBLAS_INT ldA,
527                   const float *B, CBLAS_INT ldB, const float *beta, float *C, CBLAS_INT ldC);
528
529 void cblas_zsyr2k(enum CBLAS_ORDER order, enum CBLAS_UPLO upLo, enum CBLAS_TRANSPOSE trans,
530                   CBLAS_INT n, CBLAS_INT k, const double *alpha, const double *A, CBLAS_INT ldA,
531                   const double *B, CBLAS_INT ldB, const double *beta, double *C, CBLAS_INT ldC);
532
533 // trmm
534 void cblas_strmm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo,
535                  enum CBLAS_TRANSPOSE transA, enum CBLAS_DIAG diag, CBLAS_INT m, CBLAS_INT n,
536                  float alpha, const float *A, CBLAS_INT ldA, float *B, CBLAS_INT ldB);
537
538 void cblas_dtrmm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo,
539                  enum CBLAS_TRANSPOSE transA, enum CBLAS_DIAG diag, CBLAS_INT m, CBLAS_INT n,
540                  double alpha, const double *A, CBLAS_INT ldA, double *B, CBLAS_INT ldB);
541
542 void cblas_ctrmm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo,
543                  enum CBLAS_TRANSPOSE transA, enum CBLAS_DIAG diag, CBLAS_INT m, CBLAS_INT n,
544                  const float *alpha, const float *A, CBLAS_INT ldA, float *B, CBLAS_INT ldB);
545
546 void cblas_ztrmm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo,
547                  enum CBLAS_TRANSPOSE transA, enum CBLAS_DIAG diag, CBLAS_INT m, CBLAS_INT n,
548                  const double *alpha, const double *A, CBLAS_INT ldA, double *B, CBLAS_INT ldB);
549
550 // trsm
551 void cblas_strsm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo,
552                  enum CBLAS_TRANSPOSE transA, enum CBLAS_DIAG diag, CBLAS_INT m, CBLAS_INT n,
553                  float alpha, const float *A, CBLAS_INT ldA, float *B, CBLAS_INT ldB);
554
555 void cblas_dtrsm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo,
556                  enum CBLAS_TRANSPOSE transA, enum CBLAS_DIAG diag, CBLAS_INT m, CBLAS_INT n,
557                  double alpha, const double *A, CBLAS_INT ldA, double *B, CBLAS_INT ldB);
558
559 void cblas_ctrsm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo,
560                  enum CBLAS_TRANSPOSE transA, enum CBLAS_DIAG diag, CBLAS_INT m, CBLAS_INT n,
561                  const float *alpha, const float *A, CBLAS_INT ldA, float *B, CBLAS_INT ldB);
562
563 void cblas_ztrsm(enum CBLAS_ORDER order, enum CBLAS_SIDE side, enum CBLAS_UPLO upLo,
564                  enum CBLAS_TRANSPOSE transA, enum CBLAS_DIAG diag, CBLAS_INT m, CBLAS_INT n,
565                  const double *alpha, const double *A, CBLAS_INT ldA, double *B, CBLAS_INT ldB);
566
567 #ifdef __cplusplus
568 } // extern "C"
569 #endif
570
571 #endif // CXXBLAS_DRIVERS_CBLAS_H
```

7.14 drivers.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
```

```
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_DRIVERS_DRIVERS_H
34 #define CXXBLAS_DRIVERS_DRIVERS_H 1
35
36 #include "cxxblas/auxiliary/issame.h"
37 #include "cxxblas/auxiliary/restrictto.h"
38
39 // define implementation specific constants, macros, etc.
40 #if defined(LIBRAPID_BLAS_ATLAS)
41 #   include "cxxblas/drivers/atlas.h"
42 #elif defined(LIBRAPID_BLAS_GOTOBLAS)
43 #   include "cxxblas/drivers/gotoblas.h"
44 #elif defined(LIBRAPID_BLAS_OPENBLAS)
45 #   include "cxxblas/drivers/openblas.h"
46 #elif defined(LIBRAPID_BLAS_VECLIB)
47 #   include "cxxblas/drivers/veclib.h"
48 #elif defined(LIBRAPID_BLAS_MKLBLAS)
49 #   include "cxxblas/drivers/mklblas.h"
50 #elif defined(LIBRAPID_BLAS_REFBLAS)
51 #   include "cxxblas/drivers/refblas.h"
52 #endif
53
54 #ifdef HAVE_CBLAS
55 #   include "cxxblas/drivers/cblas.h"
56 #endif
57
58 #ifdef HAVE_SPARSEBLAS
59 #   include "cxxblas/drivers/sparseblas.h"
60 #endif
61
62 #include "cxxblas/typedefs.h"
63
64 namespace cxxblas {
65
66     template<typename CHAR>
67     const CHAR *blasImpl();
68
69     //--------------------------------------------------------------------------
70
71     template<typename Any>
72     struct If {};
73
74     template<>
75     struct If<int> {
76         typedef void isBlasCompatibleInteger;
77     };
78
79     template<>
80     struct If<long> {
81         typedef void isBlasCompatibleInteger;
82     };
83
84     //--------------------------------------------------------------------------
85     template<typename ENUM>
86     typename RestrictTo<IsSame<ENUM, Transpose>::value, char>::Type getF77BlasChar(ENUM trans);
87
88     template<typename ENUM>
89     typename RestrictTo<IsSame<ENUM, Diag>::value, char>::Type getF77BlasChar(ENUM diag);
```

```
90
91      template<typename ENUM>
92      typename RestrictTo<IsSame<ENUM, StorageUpLo>::value, char>::Type getF77BlasChar(ENUM upLo);
93
94      //--------------------------------------------------------------------------
95      template<typename ENUM>
96      typename RestrictTo<IsSame<ENUM, Transpose>::value, Transpose>::Type getCxxBlasEnum(char trans);
97
98      template<typename ENUM>
99      typename RestrictTo<IsSame<ENUM, Diag>::value, Diag>::Type getCxxBlasEnum(char diag);
100
101      template<typename ENUM>
102      typename RestrictTo<IsSame<ENUM, StorageUpLo>::value, StorageUpLo>::Type
103      getCxxBlasEnum(char upLo);
104
105 //--------------------------------------------------------------------------
106 #ifdef HAVE_CBLAS
107
108      namespace CBLAS {
109
110          // TODO: rename these to getCblasEnum
111
112          template<typename ENUM>
113          typename RestrictTo<IsSame<ENUM, StorageOrder>::value, CBLAS_ORDER>::Type
114          getCblasType(ENUM order);
115
116          template<typename ENUM>
117          typename RestrictTo<IsSame<ENUM, Transpose>::value, CBLAS_TRANSPOSE>::Type
118          getCblasType(ENUM trans);
119
120          template<typename ENUM>
121          typename RestrictTo<IsSame<ENUM, StorageUpLo>::value, CBLAS_UPLO>::Type
122          getCblasType(ENUM upLo);
123
124          template<typename ENUM>
125          typename RestrictTo<IsSame<ENUM, Side>::value, CBLAS_SIDE>::Type getCblasType(ENUM side);
126
127          template<typename ENUM>
128          typename RestrictTo<IsSame<ENUM, Diag>::value, CBLAS_DIAG>::Type getCblasType(ENUM diag);
129
130      } // namespace CBLAS
131
132 #endif // HAVE_CBLAS
133
134 } // namespace cxxblas
135
136 #endif // CXXBLAS_DRIVERS_DRIVERS_H
```

7.15 gotoblas.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_DRIVERS_GOTOBLAS_H
```

```
34 #define CXXBLAS_DRIVERS_GOTOBLAS_H 1
35
36 #define HAVE_CBLAS 1
37 #ifdef BLASINT
38 #   define CBLAS_INT BLASINT
39 #else
40 #   define CBLAS_INT int
41 #endif
42 #define BLAS_IMPL "GotoBLAS"
43 #ifndef CBLAS_INDEX
44 #   define CBLAS_INDEX size_t
45 #endif // CBLAS_INDEX
46
47 #endif // CXXBLAS_DRIVERS_GOTOBLAS_H
```

7.16 mklblas.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_DRIVERS_MKLBLAS_H
34 #define CXXBLAS_DRIVERS_MKLBLAS_H 1
35
36 #define HAVE_CBLAS      1
37 #define HAVE_SPARSEBLAS 1
38 #define WITH_MKLDSS     1
39 #ifdef MKL_ILP64
40 #   define CBLAS_INT    long
41 #   define CBLAS_INDEX long
42 #else
43 #   define CBLAS_INT    int
44 #   define CBLAS_INDEX int
45 #endif
46 #define BLAS_IMPL "MKLBLAS"
47
48 // BLAS extensions
49 #ifndef HAVE_CBLAS_AXPBY
50 #   define HAVE_CBLAS_AXPBY
51 #   define BLAS_EXT(x) cblas_##x
52 #endif
53
54 // MKL includes LAPACK
55 #ifndef USE_CXXLAPACK
56 #   define USE_CXXLAPACK 1
57 #endif
58 // MKL includes FFTW interface (float, double)
59 #ifndef HAVE_FFTW
60 #   define HAVE_FFTW 1
61 #endif
62 #ifndef HAVE_FFTW_FLOAT
63 #   define HAVE_FFTW_FLOAT 1
64 #endif
65 #ifndef HAVE_FFTW_DOUBLE
66 #   define HAVE_FFTW_DOUBLE 1
```

```
67 #endif
68
69 #endif // CXXBLAS_DRIVERS_MKLBLAS_H
```

7.17 openblas.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_DRIVERS_OPENBLAS_H
34 #define CXXBLAS_DRIVERS_OPENBLAS_H 1
35
36 #define HAVE_CBLAS 1
37 #ifdef BLASINT
38 #   define CBLAS_INT BLASINT
39 #else
40 #   define CBLAS_INT int
41 #endif
42 #define BLAS_IMPL "OpenBLAS"
43 #ifndef CBLAS_INDEX
44 #   define CBLAS_INDEX size_t
45 #endif // CBLAS_INDEX
46
47 // BLAS extensions
48 #ifndef HAVE_CBLAS_AXPBY
49 #   define HAVE_CBLAS_AXPBY
50 #   define BLAS_EXT(x) cblas_##x
51 #endif
52
53 extern "C" {
54 /* Assume C declarations for C++ */
55
56 /*Set the number of threads on runtime.*/
57 void openblas_set_num_threads(int num_threads);
58 void goto_set_num_threads(int num_threads);
59
60 /*Get the number of threads on runtime.*/
61 int openblas_get_num_threads(void);
62
63 /*Get the number of physical processors (cores).*/
64 int openblas_get_num_procs(void);
65
66 /*Get the build configure on runtime.*/
67 char *openblas_get_config(void);
68
69 /*Get the CPU corename on runtime.*/
70 char *openblas_get_corename(void);
71
72 /* Get the parallelization type which is used by OpenBLAS */
73 int openblas_get_parallel(void);
74 }
75
76 /* OpenBLAS is compiled for sequential use  */
77 #define OPENBLAS_SEQUENTIAL 0
```

```
78 /* OpenBLAS is compiled using normal threading model */
79 #define OPENBLAS_THREAD 1
80 /* OpenBLAS is compiled using OpenMP threading model */
81 #define OPENBLAS_OPENMP 2
82
83 #endif // CXXBLAS_DRIVERS_OPENBLAS_H
```

7.18 refblas.h

```
1 /*
2  *   Copyright (c) 2011, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_DRIVERS_REFBLAS_H
34 #define CXXBLAS_DRIVERS_REFBLAS_H 1
35
36 #define HAVE_CBLAS 1
37 #ifdef BLASINT
38 #   define CBLAS_INT BLASINT
39 #else
40 #   define CBLAS_INT int
41 #endif
42 #define BLAS_IMPL "RefBLAS"
43 #ifndef CBLAS_INDEX
44 #   define CBLAS_INDEX size_t
45 #endif // CBLAS_INDEX
46
47 #endif // CXXBLAS_DRIVERS_REFBLAS_H
```

7.19 sparseblas.h

```
1 /*
2  *   Modification of mkl_spblas.h
3  */
4
5 #ifndef CXXBLAS_DRIVERS_SPARSEBLAS_H
6 #define CXXBLAS_DRIVERS_SPARSEBLAS_H 1
7
8 #include "cxxblas/drivers/mklblas.h"
9
10 #ifndef SPARSEBLAS_INT
11 #   define SPARSEBLAS_INT int
12 #endif // CBLAS_INT
13
14 #ifndef SPARSEBLAS_INDEX
15 #   define SPARSEBLAS_INDEX int
16 #endif // CBLAS_INT
17
18 #ifdef __cplusplus
19 extern "C" {
20 #endif
```

```
21
22 //-- LEVEL 2 ------------------------------------------------------------------
23
24 // mv
25 void mkl_scscmv(const char *transa, const CBLAS_INT *m, const CBLAS_INT *k, const float *alpha,
26                  const char *matdescra, const float *val, const CBLAS_INT *indx,
27                  const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *x, const float *beta,
28                  float *y);
29
30 void mkl_scsrmv(const char *transa, const CBLAS_INT *m, const CBLAS_INT *k, const float *alpha,
31                  const char *matdescra, const float *val, const CBLAS_INT *indx,
32                  const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *x, const float *beta,
33                  float *y);
34
35 void mkl_dcscmv(const char *transa, const CBLAS_INT *m, const CBLAS_INT *k, const double *alpha,
36                  const char *matdescra, const double *val, const CBLAS_INT *indx,
37                  const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const double *x, const double *beta,
38                  double *y);
39
40 void mkl_dcsrmv(const char *transa, const CBLAS_INT *m, const CBLAS_INT *k, const double *alpha,
41                  const char *matdescra, const double *val, const CBLAS_INT *indx,
42                  const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const double *x, const double *beta,
43                  double *y);
44
45 void mkl_ccscmv(const char *transa, const CBLAS_INT *m, const CBLAS_INT *k, const float *alpha,
46                  const char *matdescra, const float *val, const CBLAS_INT *indx,
47                  const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *x, const float *beta,
48                  float *y);
49
50 void mkl_ccsrmv(const char *transa, const CBLAS_INT *m, const CBLAS_INT *k, const float *alpha,
51                  const char *matdescra, const float *val, const CBLAS_INT *indx,
52                  const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *x, const float *beta,
53                  float *y);
54
55 void mkl_zcscmv(const char *transa, const CBLAS_INT *m, const CBLAS_INT *k, const double *alpha,
56                  const char *matdescra, const double *val, const CBLAS_INT *indx,
57                  const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const double *x, const double *beta,
58                  double *y);
59
60 void mkl_zcsrmv(const char *transa, const CBLAS_INT *m, const CBLAS_INT *k, const double *alpha,
61                  const char *matdescra, const double *val, const CBLAS_INT *indx,
62                  const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const double *x, const double *beta,
63                  double *y);
64
65 // sv
66 void mkl_scsrsv(const char *transa, const CBLAS_INT *m, const float *alpha, const char *matdescra,
67                  const float *val, const CBLAS_INT *indx, const CBLAS_INT *pntrb,
68                  const CBLAS_INT *pntre, const float *x, float *y);
69
70 void mkl_scscsv(const char *transa, const CBLAS_INT *m, const float *alpha, const char *matdescra,
71                  const float *val, const CBLAS_INT *indx, const CBLAS_INT *pntrb,
72                  const CBLAS_INT *pntre, const float *x, float *y);
73
74 void mkl_dcsrsv(const char *transa, const CBLAS_INT *m, const double *alpha, const char *matdescra,
75                  const double *val, const CBLAS_INT *indx, const CBLAS_INT *pntrb,
76                  const CBLAS_INT *pntre, const double *x, double *y);
77
78 void mkl_dcscsv(const char *transa, const CBLAS_INT *m, const double *alpha, const char *matdescra,
79                  const double *val, const CBLAS_INT *indx, const CBLAS_INT *pntrb,
80                  const CBLAS_INT *pntre, const double *x, double *y);
81
82 void mkl_ccsrsv(const char *transa, const CBLAS_INT *m, const float *alpha, const char *matdescra,
83                  const float *val, const CBLAS_INT *indx, const CBLAS_INT *pntrb,
84                  const CBLAS_INT *pntre, const float *x, float *y);
85
86 void mkl_ccscsv(const char *transa, const CBLAS_INT *m, const float *alpha, const char *matdescra,
87                  const float *val, const CBLAS_INT *indx, const CBLAS_INT *pntrb,
88                  const CBLAS_INT *pntre, const float *x, float *y);
89
90 void mkl_zcsrsv(const char *transa, const CBLAS_INT *m, const double *alpha, const char *matdescra,
91                  const double *val, const CBLAS_INT *indx, const CBLAS_INT *pntrb,
92                  const CBLAS_INT *pntre, const double *x, double *y);
93
94 void mkl_zcscsv(const char *transa, const CBLAS_INT *m, const double *alpha, const char *matdescra,
95                  const double *val, const CBLAS_INT *indx, const CBLAS_INT *pntrb,
96                  const CBLAS_INT *pntre, const double *x, double *y);
97
98 // Level 3
99
100 // mm
101 void mkl_scscmm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const CBLAS_INT *k,
102                  const float *alpha, char *matdescra, const float *val, const CBLAS_INT *indx,
103                  const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *b, CBLAS_INT *ldb,
104                  const float *beta, const float *c, CBLAS_INT *ldc);
105
106 void mkl_scsrmm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const CBLAS_INT *k,
107                  const float *alpha, const char *matdescra, const float *val, const CBLAS_INT *indx,
```

```
108                 const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *b,
109                 const CBLAS_INT *ldb, const float *beta, float *c, const CBLAS_INT *ldc);
110
111 void mkl_dcscmm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const CBLAS_INT *k,
112                 const double *alpha, const char *matdescra, const double *val,
113                 const CBLAS_INT *indx, const CBLAS_INT *pntrb, const CBLAS_INT *pntre,
114                 const double *b, const CBLAS_INT *ldb, const double *beta, double *c,
115                 const CBLAS_INT *ldc);
116
117 void mkl_dcsrmm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const CBLAS_INT *k,
118                 const double *alpha, const char *matdescra, const double *val,
119                 const CBLAS_INT *indx, const CBLAS_INT *pntrb, const CBLAS_INT *pntre,
120                 const double *b, const CBLAS_INT *ldb, const double *beta, double *c,
121                 const CBLAS_INT *ldc);
122
123 void mkl_ccscmm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const CBLAS_INT *k,
124                 const float *alpha, const char *matdescra, const float *val, const CBLAS_INT *indx,
125                 const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *b,
126                 const CBLAS_INT *ldb, const float *beta, float *c, const CBLAS_INT *ldc);
127
128 void mkl_ccsrmm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const CBLAS_INT *k,
129                 const float *alpha, const char *matdescra, const float *val, const CBLAS_INT *indx,
130                 const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *b,
131                 const CBLAS_INT *ldb, const float *beta, float *c, const CBLAS_INT *ldc);
132
133 void mkl_zcscmm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const CBLAS_INT *k,
134                 const double *alpha, const char *matdescra, const double *val,
135                 const CBLAS_INT *indx, const CBLAS_INT *pntrb, const CBLAS_INT *pntre,
136                 const double *b, const CBLAS_INT *ldb, const double *beta, double *c,
137                 const CBLAS_INT *ldc);
138
139 void mkl_zcsrmm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const CBLAS_INT *k,
140                 const double *alpha, const char *matdescra, const double *val,
141                 const CBLAS_INT *indx, const CBLAS_INT *pntrb, const CBLAS_INT *pntre,
142                 const double *b, const CBLAS_INT *ldb, const double *beta, double *c,
143                 const CBLAS_INT *ldc);
144
145 // sm
146 void mkl_scsrsm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const float *alpha,
147                 const char *matdescra, const float *val, const CBLAS_INT *indx,
148                 const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *b,
149                 const CBLAS_INT *ldb, float *c, const CBLAS_INT *ldc);
150
151 void mkl_scscsm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const float *alpha,
152                 const char *matdescra, const float *val, const CBLAS_INT *indx,
153                 const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *b,
154                 const CBLAS_INT *ldb, float *c, const CBLAS_INT *ldc);
155
156 void mkl_dcsrsm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const double *alpha,
157                 const char *matdescra, const double *val, const CBLAS_INT *indx,
158                 const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const double *b,
159                 const CBLAS_INT *ldb, double *c, const CBLAS_INT *ldc);
160
161 void mkl_dcscsm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const double *alpha,
162                 const char *matdescra, const double *val, const CBLAS_INT *indx,
163                 const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const double *b,
164                 const CBLAS_INT *ldb, double *c, const CBLAS_INT *ldc);
165
166 void mkl_ccsrsm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const float *alpha,
167                 const char *matdescra, const float *val, const CBLAS_INT *indx,
168                 const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *b,
169                 const CBLAS_INT *ldb, float *c, const CBLAS_INT *ldc);
170
171 void mkl_ccscsm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const float *alpha,
172                 const char *matdescra, const float *val, const CBLAS_INT *indx,
173                 const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const float *b,
174                 const CBLAS_INT *ldb, float *c, const CBLAS_INT *ldc);
175
176 void mkl_zcsrsm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const double *alpha,
177                 const char *matdescra, const double *val, const CBLAS_INT *indx,
178                 const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const double *b,
179                 const CBLAS_INT *ldb, double *c, const CBLAS_INT *ldc);
180
181 void mkl_zcscsm(const char *transa, const CBLAS_INT *m, const CBLAS_INT *n, const double *alpha,
182                 const char *matdescra, const double *val, const CBLAS_INT *indx,
183                 const CBLAS_INT *pntrb, const CBLAS_INT *pntre, const double *b,
184                 const CBLAS_INT *ldb, double *c, const CBLAS_INT *ldc);
185
186 #ifdef __cplusplus
187 } // extern "C"
188 #endif
189
190 #endif // CXXBLAS_DRIVERS_SPARSEBLAS_H
```

7.20 veclib.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_DRIVERS_VECLIB_H
34 #define CXXBLAS_DRIVERS_VECLIB_H 1
35
36 #define HAVE_CBLAS 1
37 #define CBLAS_INT  int
38 #define BLAS_IMPL  "VecLib (ATLAS)"
39 #ifndef CBLAS_INDEX
40 #   define CBLAS_INDEX int
41 #endif // CBLAS_INDEX
42
43 // BLAS extensions
44 #ifndef HAVE_CBLAS_AXPBY
45 #   define HAVE_CBLAS_AXPBY
46 #   define BLAS_EXT(x) catlas_##x
47 #endif
48
49 // VECLIB includes LAPACK interface
50 #ifndef USE_CXXLAPACK
51 #   define USE_CXXLAPACK 1
52 #endif
53
54 #endif // CXXBLAS_DRIVERS_VECLIB_H
```

7.21 asum.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
```

```
 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_LEVEL1_ASUM_H
#define CXXBLAS_LEVEL1_ASUM_H 1

#include "cxxblas/typedefs.h"
#include "cxxblas/drivers/drivers.h"

#define HAVE_CXXBLAS_ASUM 1

namespace cxxblas {

    template<typename IndexType, typename X, typename T>
    void asum(IndexType n, const X *x, IndexType incX, T &absSum);

#ifdef HAVE_CBLAS
    // sasum
    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger asum(IndexType n, const float *x,
                                                         IndexType incX, float &absSum);

    // dasum
    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger asum(IndexType n, const double *x,
                                                         IndexType incX, double &absSum);

    // scasum
    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger asum(IndexType n, const ComplexFloat *x,
                                                         IndexType incX, float &absSum);

    // dzasum
    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger asum(IndexType n, const ComplexDouble *x,
                                                         IndexType incX, double &absSum);

#endif // HAVE_CBLAS

} // namespace cxxblas

#endif // CXXBLAS_LEVEL1_ASUM_H
```

7.22 axpy.h

```
/*
 *   Copyright (c) 2009, Michael Lehn
 *
 *   All rights reserved.
 *
 *   Redistribution and use in source and binary forms, with or without
 *   modification, are permitted provided that the following conditions
 *   are met:
 *
 *   1) Redistributions of source code must retain the above copyright
 *      notice, this list of conditions and the following disclaimer.
 *   2) Redistributions in binary form must reproduce the above copyright
 *      notice, this list of conditions and the following disclaimer in
 *      the documentation and/or other materials provided with the
 *      distribution.
 *   3) Neither the name of the FLENS development group nor the names of
 *      its contributors may be used to endorse or promote products derived
 *      from this software without specific prior written permission.
 *
 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_LEVEL1_AXPY_H
```

```
34 #define CXXBLAS_LEVEL1_AXPY_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_AXPY 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename X, typename Y>
44     void axpy(IndexType n, const ALPHA &alpha, const X *x, IndexType incX, Y *y, IndexType incY);
45
46 #ifdef HAVE_CBLAS
47     // saxpy
48     template<typename IndexType>
49     typename If<IndexType>::isBlasCompatibleInteger
50     axpy(IndexType n, const float &alpha, const float *x, IndexType incX, float *y, IndexType incY);
51
52     // daxpy
53     template<typename IndexType>
54     typename If<IndexType>::isBlasCompatibleInteger axpy(IndexType n, const double &alpha,
55                                                          const double *x, IndexType incX, double *y,
56                                                          IndexType incY);
57
58     // caxpy
59     template<typename IndexType>
60     typename If<IndexType>::isBlasCompatibleInteger axpy(IndexType n, const ComplexFloat &alpha,
61                                                          const ComplexFloat *x, IndexType incX,
62                                                          ComplexFloat *y, IndexType incY);
63
64     // zaxpy
65     template<typename IndexType>
66     typename If<IndexType>::isBlasCompatibleInteger axpy(IndexType n, const ComplexDouble &alpha,
67                                                          const ComplexDouble *x, IndexType incX,
68                                                          ComplexDouble *y, IndexType incY);
69
70 #endif // HAVE_CBLAS
71
72 } // namespace cxxblas
73
74 #endif // CXXBLAS_LEVEL1_AXPY_H
```

7.23 axpy.h

```
1 /*
2  *   Copyright (c) 2012, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_AXPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_AXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_AXPY 1
39
```

```
40 namespace cxxblas {
41
42 #ifdef HAVE_CBLAS
43
44     template<typename IndexType>
45     void axpy(IndexType n, const float &alpha, const float *x, IndexType incX,
46               std::complex<float> *y, IndexType incY);
47
48     template<typename IndexType>
49     void axpy(IndexType n, const std::complex<float> &alpha, const float *x, IndexType incX,
50               std::complex<float> *y, IndexType incY);
51
52     template<typename IndexType>
53     void axpy(IndexType n, const double &alpha, const double *x, IndexType incX,
54               std::complex<double> *y, IndexType incY);
55
56     template<typename IndexType>
57     void axpy(IndexType n, const std::complex<double> &alpha, const double *x, IndexType incX,
58               std::complex<double> *y, IndexType incY);
59
60 #endif // HAVE_CBLAS
61
62 } // namespace cxxblas
63
64 #endif // CXXBLAS_LEVEL1EXTENSIONS_AXPY_H
```

7.24 axpy.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_TINYLEVEL1_AXPY_H
34 #define CXXBLAS_TINYLEVEL1_AXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     template<int n, typename ALPHA, typename X, int incX, typename Y, int incY>
41     void axpy(const ALPHA &alpha, const X *x, Y *y);
42
43 } // namespace cxxblas
44
45 #endif // CXXBLAS_TINYLEVEL1_AXPY_H
```

7.25 copy.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
```

```
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1_COPY_H
34 #define CXXBLAS_LEVEL1_COPY_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_COPY 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename X, typename Y>
44     void copy(IndexType n, const X *x, IndexType incX, Y *y, IndexType incY);
45
46 #ifdef HAVE_CBLAS
47
48     // scopy
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger copy(IndexType n, const float *x,
51                                                          IndexType incX, float *y, IndexType incY);
52
53     // dcopy
54     template<typename IndexType>
55     typename If<IndexType>::isBlasCompatibleInteger copy(IndexType n, const double *x,
56                                                          IndexType incX, double *y, IndexType incY);
57
58     // ccopy
59     template<typename IndexType>
60     typename If<IndexType>::isBlasCompatibleInteger
61     copy(IndexType n, const ComplexFloat *x, IndexType incX, ComplexFloat *y, IndexType incY);
62
63     // zcopy
64     template<typename IndexType>
65     typename If<IndexType>::isBlasCompatibleInteger
66     copy(IndexType n, const ComplexDouble *x, IndexType incX, ComplexDouble *y, IndexType incY);
67
68 #endif // HAVE_CBLAS
69
70 } // namespace cxxblas
71
72 #endif // CXXBLAS_LEVEL1_COPY_H
```

7.26 copy.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
```

```
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_TINYLEVEL1_COPY_H
34 #define CXXBLAS_TINYLEVEL1_COPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     template<int n, typename X, int incX, typename Y, int incY>
41     void copy(const X *x, Y *y);
42
43 } // namespace cxxblas
44
45 #endif // CXXBLAS_TINYLEVEL1_COPY_H
```

7.27 dot.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1_DOT_H
34 #define CXXBLAS_LEVEL1_DOT_H 1
35
36 #include "cxxblas/cxxblas.h"
37
38 #define HAVE_CXXBLAS_DOT  1
39 #define HAVE_CXXBLAS_DOTU 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename X, typename Y, typename Result>
44     void dotu(IndexType n, const X *x, IndexType incX, const Y *y, IndexType incY, Result &result);
45
46     template<typename IndexType, typename X, typename Y, typename Result>
```

```
47      void dot(IndexType n, const X *x, IndexType incX, const Y *y, IndexType incY, Result &result);
48
49  #ifdef HAVE_CBLAS
50
51      // sdsdot
52      template<typename IndexType>
53      typename If<IndexType>::isBlasCompatibleInteger sdot(IndexType n, float alpha, const float *x,
54                                                           IndexType incX, const float *y,
55                                                           IndexType incY, float &result);
56
57      // dsdot
58      template<typename IndexType>
59      typename If<IndexType>::isBlasCompatibleInteger dot(IndexType n, const float *x, IndexType incX,
60                                                          const float *y, IndexType incY,
61                                                          double &result);
62
63      // sdot
64      template<typename IndexType>
65      typename If<IndexType>::isBlasCompatibleInteger
66      dot(IndexType n, const float *x, IndexType incX, const float *y, IndexType incY, float &result);
67
68      // ddot
69      template<typename IndexType>
70      typename If<IndexType>::isBlasCompatibleInteger dot(IndexType n, const double *x,
71                                                          IndexType incX, const double *y,
72                                                          IndexType incY, double &result);
73
74      // cdotu_sub
75      template<typename IndexType>
76      typename If<IndexType>::isBlasCompatibleInteger dotu(IndexType n, const ComplexFloat *x,
77                                                           IndexType incX, const ComplexFloat *y,
78                                                           IndexType incY, ComplexFloat &result);
79
80      // cdotc_sub
81      template<typename IndexType>
82      typename If<IndexType>::isBlasCompatibleInteger dot(IndexType n, const ComplexFloat *x,
83                                                          IndexType incX, const ComplexFloat *y,
84                                                          IndexType incY, ComplexFloat &result);
85
86      // zdotu_sub
87      template<typename IndexType>
88      typename If<IndexType>::isBlasCompatibleInteger dotu(IndexType n, const ComplexDouble *x,
89                                                           IndexType incX, const ComplexDouble *y,
90                                                           IndexType incY, ComplexDouble &result);
91
92      // zdotc_sub
93      template<typename IndexType>
94      typename If<IndexType>::isBlasCompatibleInteger dot(IndexType n, const ComplexDouble *x,
95                                                          IndexType incX, const ComplexDouble *y,
96                                                          IndexType incY, ComplexDouble &result);
97
98  #endif // HAVE_CBLAS
99
100 } // namespace cxxblas
101
102 #endif // CXXBLAS_LEVEL1_DOT_H
```

7.28 dot.h

```
1  /*
2   *   Copyright (c) 2012, Klaus Pototzky
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
```

```
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_DOT_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_DOT_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_AXPY 1
39
40 namespace cxxblas {
41
42 #ifdef HAVE_CBLAS
43
44     template<typename IndexType>
45     void dotu(IndexType n, const float *x, IndexType incX, const std::complex<float> *y,
46               IndexType incY, std::complex<float> &result);
47
48     template<typename IndexType>
49     void dotu(IndexType n, const std::complex<float> *x, IndexType incX, const float *y,
50               IndexType incY, std::complex<float> &result);
51
52     template<typename IndexType>
53     void dotu(IndexType n, const double *x, IndexType incX, const std::complex<double> *y,
54               IndexType incY, std::complex<double> &result);
55
56     template<typename IndexType>
57     void dotu(IndexType n, const std::complex<double> *x, IndexType incX, const double *y,
58               IndexType incY, std::complex<double> &result);
59
60     template<typename IndexType>
61     void dot(IndexType n, const float *x, IndexType incX, const std::complex<float> *y,
62              IndexType incY, std::complex<float> &result);
63
64     template<typename IndexType>
65     void dot(IndexType n, const std::complex<float> *x, IndexType incX, const float *y,
66              IndexType incY, std::complex<float> &result);
67
68     template<typename IndexType>
69     void dot(IndexType n, const double *x, IndexType incX, const std::complex<double> *y,
70              IndexType incY, std::complex<double> &result);
71
72     template<typename IndexType>
73     void dot(IndexType n, const std::complex<double> *x, IndexType incX, const double *y,
74              IndexType incY, std::complex<double> &result);
75
76 #endif
77
78 } // namespace cxxblas
79
80 #endif // CXXBLAS_LEVEL1EXTENSIONS_DOT_H
```

7.29 iamax.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
```

```
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1_IAMAX_H
34 #define CXXBLAS_LEVEL1_IAMAX_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_IAMAX 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename X>
44     void iamax(IndexType n, const X *x, IndexType incX, IndexType &i);
45
46     template<typename IndexType, typename X>
47     IndexType iamax(IndexType n, const X *x, IndexType incX);
48
49 #ifdef HAVE_CBLAS
50     // isamax
51     template<typename IndexType>
52     typename If<IndexType>::isBlasCompatibleInteger iamax(IndexType n, const float *x,
53                                                           IndexType incX, IndexType &i);
54
55     // idamax
56     template<typename IndexType>
57     typename If<IndexType>::isBlasCompatibleInteger iamax(IndexType n, const double *x,
58                                                           IndexType incX, IndexType &i);
59
60     // icamax
61     template<typename IndexType>
62     typename If<IndexType>::isBlasCompatibleInteger iamax(IndexType n, const ComplexFloat *x,
63                                                           IndexType incX, IndexType &i);
64
65     // izamax
66     template<typename IndexType>
67     typename If<IndexType>::isBlasCompatibleInteger iamax(IndexType n, const ComplexDouble *x,
68                                                           IndexType incX, IndexType &i);
69
70 #endif // HAVE_CBLAS
71
72 } // namespace cxxblas
73
74 #endif // CXXBLAS_LEVEL1_IAMAX_H
```

7.30 level1.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

```
31  */
32
33 #ifndef CXXBLAS_LEVEL1_LEVEL1_H
34 #define CXXBLAS_LEVEL1_LEVEL1_H 1
35
36 #include "cxxblas/level1/asum.h"
37 #include "cxxblas/level1/axpy.h"
38 #include "cxxblas/level1/copy.h"
39 #include "cxxblas/level1/dot.h"
40 #include "cxxblas/level1/iamax.h"
41 #include "cxxblas/level1/nrm2.h"
42 #include "cxxblas/level1/rot.h"
43 #include "cxxblas/level1/rotm.h"
44 #include "cxxblas/level1/scal.h"
45 #include "cxxblas/level1/swap.h"
46
47 #endif // CXXBLAS_LEVEL1_LEVEL1_H
```

7.31 nrm2.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1_NRM2_H
34 #define CXXBLAS_LEVEL1_NRM2_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_NRM2 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename X, typename T>
44     void nrm2(IndexType n, const X *x, IndexType incX, T &norm);
45
46 #ifdef HAVE_CBLAS
47     // snrm2
48     template<typename IndexType>
49     typename If<IndexType>::isBlasCompatibleInteger nrm2(IndexType n, const float *x,
50                                                          IndexType incX, float &norm);
51
52     // dnrm2
53     template<typename IndexType>
54     typename If<IndexType>::isBlasCompatibleInteger nrm2(IndexType n, const double *x,
55                                                          IndexType incX, double &norm);
56
57     // scnrm2
58     template<typename IndexType>
59     typename If<IndexType>::isBlasCompatibleInteger nrm2(IndexType n, const ComplexFloat *x,
60                                                          IndexType incX, float &norm);
61
62     // dznrm2
63     template<typename IndexType>
```

```
64      typename If<IndexType>::isBlasCompatibleInteger nrm2(IndexType n, const ComplexDouble *x,
65                                                           IndexType incX, double &norm);
66
67  #endif // HAVE_CBLAS
68
69  } // namespace cxxblas
70
71  #endif // CXXBLAS_LEVEL1_NRM2_H
```

7.32 rot.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_LEVEL1_ROT_H
34  #define CXXBLAS_LEVEL1_ROT_H 1
35
36  #include "cxxblas/drivers/drivers.h"
37  #include "cxxblas/typedefs.h"
38
39  #define HAVE_CXXBLAS_ROT  1
40  #define HAVE_CXXBLAS_ROTG 1
41
42  namespace cxxblas {
43
44      template<typename IndexType, typename X, typename Y, typename T>
45      void rot(IndexType n, X *x, IndexType incX, Y *y, IndexType incY, T c, T s);
46
47      template<typename A, typename B, typename T>
48      void rotg(A &a, B &b, T &c, T &s);
49
50      template<typename TA, typename TB, typename T>
51      void rotg(std::complex<TA> &a, std::complex<TB> &b, T &c, std::complex<T> &s);
52
53      /*
54       *  Note: The following variant of function rot is based on
55       *
56              SUBROUTINE ZROT( N, CX, INCX, CY, INCY, C, S )
57       *
58       *  -- LAPACK auxiliary routine (version 3.2) --
59       *  -- LAPACK is a software package provided by Univ. of Tennessee,    --
60       *  -- Univ. of California Berkeley, Univ. of Colorado Denver and NAG Ltd..--
61       *     November 2006
62       */
63      template<typename IndexType, typename X, typename Y, typename T>
64      void rot(IndexType n, std::complex<X> *x, IndexType incX, std::complex<Y> *y, IndexType incY,
65               T c, const std::complex<T> &s);
66
67  #ifdef HAVE_CBLAS
68      // srot
69      template<typename IndexType>
70      typename If<IndexType>::isBlasCompatibleInteger rot(IndexType n, float *x, IndexType incX,
71                                                          float *y, IndexType incY, float c, float s);
72
```

```
73     // drot
74     template<typename IndexType>
75     typename If<IndexType>::isBlasCompatibleInteger
76     rot(IndexType n, double *x, IndexType incX, double *y, IndexType incY, double c, double s);
77
78     // srotg
79     template<typename T>
80     typename RestrictTo<IsSame<T, float>::value, void>::Type rotg(T &a, T &b, T &c, T &s);
81
82     // drotg
83     template<typename T>
84     typename RestrictTo<IsSame<T, double>::value, void>::Type rotg(T &a, T &b, T &c, T &s);
85
86 #endif // HAVE_CBLAS
87
88 } // namespace cxxblas
89
90 #endif // CXXBLAS_LEVEL1_ROT_H
```

7.33 rotm.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1_ROTM_H
34 #define CXXBLAS_LEVEL1_ROTM_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 namespace cxxblas {
40
41 #ifdef HAVE_CBLAS
42
43 // TODO: provide generic implementation of rotm, rotmg
44 #   define HAVE_CXXBLAS_ROTMG 1
45 #   define HAVE_CXXBLAS_ROTM  1
46
47     // srotm
48     template<typename IndexType>
49     typename If<IndexType>::isBlasCompatibleInteger rotm(IndexType n, float *x, IndexType incX,
50                                                          float *y, IndexType incY, const float *p);
51
52     // drotm
53     template<typename IndexType>
54     typename If<IndexType>::isBlasCompatibleInteger
55     rotm(IndexType n, double *x, IndexType incX, double *y, IndexType incY, const double *p);
56
57     // srotmg
58     template<typename T>
59     typename RestrictTo<IsSame<T, float>::value, void>::Type rotmg(T &d1, T &d2, T &b1, T &b2,
60                                                                    T *p);
61
62     // drotmg
```

```
63      template<typename T>
64      typename RestrictTo<IsSame<T, double>::value, void>::Type rotmg(T &d1, T &d2, T &b1, T &b2,
65                                                                      T *p);
66
67  #endif // HAVE_CBLAS
68
69  } // namespace cxxblas
70
71  #endif // CXXBLAS_LEVEL1_ROTM_H
```

7.34 scal.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_LEVEL1_SCAL_H
34  #define CXXBLAS_LEVEL1_SCAL_H 1
35
36  #include "cxxblas/drivers/drivers.h"
37  #include "cxxblas/typedefs.h"
38
39  #define HAVE_CXXBLAS_SCAL 1
40
41  namespace cxxblas {
42
43      template<typename IndexType, typename ALPHA, typename Y>
44      void scal(IndexType n, const ALPHA &alpha, Y *y, IndexType incY);
45
46  #ifdef HAVE_CBLAS
47
48      // sscal
49      template<typename IndexType>
50      typename If<IndexType>::isBlasCompatibleInteger scal(IndexType n, float alpha, float *x,
51                                                           IndexType incX);
52
53      // dscal
54      template<typename IndexType>
55      typename If<IndexType>::isBlasCompatibleInteger scal(IndexType n, double alpha, double *x,
56                                                           IndexType incX);
57
58      // cscal
59      template<typename IndexType>
60      typename If<IndexType>::isBlasCompatibleInteger scal(IndexType n, const ComplexFloat &alpha,
61                                                           ComplexFloat *x, IndexType incX);
62
63      // zscal
64      template<typename IndexType>
65      typename If<IndexType>::isBlasCompatibleInteger scal(IndexType n, const ComplexDouble &alpha,
66                                                           ComplexDouble *x, IndexType incX);
67
68      // csscal
69      template<typename IndexType>
70      typename If<IndexType>::isBlasCompatibleInteger scal(IndexType n, float alpha, ComplexFloat *x,
71                                                           IndexType incX);
```

```
72
73      // zdscal
74      template<typename IndexType>
75      typename If<IndexType>::isBlasCompatibleInteger scal(IndexType n, double alpha,
76                                                          ComplexDouble *x, IndexType incX);
77
78  #endif // HAVE_CBLAS
79
80  } // namespace cxxblas
81
82  #endif // CXXBLAS_LEVEL1_SCAL_H
```

7.35 scal.h

```
1  /*
2   *   Copyright (c) 2012, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_TINYLEVEL1_SCAL_H
34  #define CXXBLAS_TINYLEVEL1_SCAL_H 1
35
36  #include "cxxblas/typedefs.h"
37
38  namespace cxxblas {
39
40      template<int n, typename ALPHA, typename Y, int incY>
41      void scal(const ALPHA &alpha, Y *y);
42
43  } // namespace cxxblas
44
45  #endif // CXXBLAS_TINYLEVEL1_SCAL_H
```

7.36 swap.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
```

```
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1_SWAP_H
34 #define CXXBLAS_LEVEL1_SWAP_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_SWAP 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename X, typename Y>
44     void swap(IndexType n, X *x, IndexType incX, Y *y, IndexType incY);
45
46 #ifdef HAVE_CBLAS
47     // sswap
48     template<typename IndexType>
49     typename If<IndexType>::isBlasCompatibleInteger swap(IndexType n, float *x, IndexType incX,
50                                                          float *y, IndexType incY);
51
52     // dswap
53     template<typename IndexType>
54     typename If<IndexType>::isBlasCompatibleInteger swap(IndexType n, double *x, IndexType incX,
55                                                          double *y, IndexType incY);
56
57     // cswap
58     template<typename IndexType>
59     typename If<IndexType>::isBlasCompatibleInteger
60     swap(IndexType n, ComplexFloat *x, IndexType incX, ComplexFloat *y, IndexType incY);
61
62     // zswap
63     template<typename IndexType>
64     typename If<IndexType>::isBlasCompatibleInteger
65     swap(IndexType n, ComplexDouble *x, IndexType incX, ComplexDouble *y, IndexType incY);
66
67 #endif // HAVE_CBLAS
68
69 } // namespace cxxblas
70
71 #endif // CXXBLAS_LEVEL1_SWAP_H
```

7.37 acxpby.h

```
1  /*
2   *   Copyright (c) 2013, Klaus Pototzky
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
```

```
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_ACXPBY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_ACXPBY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_ACXPBY 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename X, typename BETA, typename Y>
43     void acxpby(IndexType n, const ALPHA &alpha, const X *x, IndexType incX, const BETA &beta, Y *y,
44                 IndexType incY);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_ACXPBY_H
```

7.38 acxpby.h

```
1  /*
2   *   Copyright (c) 2012, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_TINYLEVEL1_ACXPBY_H
34 #define CXXBLAS_TINYLEVEL1_ACXPBY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     template<int n, typename ALPHA, typename X, int incX, typename BETA, typename Y, int incY>
41     void acxpby(const ALPHA &alpha, const X *x, const BETA &beta, Y *y);
42
43 } // namespace cxxblas
44
45 #endif // CXXBLAS_TINYLEVEL1_ACXPBY_H
```

7.39 acxpy.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
```

```
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_ACXPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_ACXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_ACXPY 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename X, typename Y>
43     void acxpy(IndexType n, const ALPHA &alpha, const X *x, IndexType incX, Y *y, IndexType incY);
44
45 #ifdef HAVE_CBLAS
46
47     template<typename IndexType>
48     void acxpy(IndexType n, const float &alpha, const std::complex<float> *x, IndexType incX,
49                std::complex<float> *y, IndexType incY);
50
51     template<typename IndexType>
52     void acxpy(IndexType n, const double &alpha, const std::complex<double> *x, IndexType incX,
53                std::complex<double> *y, IndexType incY);
54
55 #endif // HAVE_CBLAS
56
57 } // namespace cxxblas
58
59 #endif // CXXBLAS_LEVEL1EXTENSIONS_ACXPY_H
```

7.40 acxpy.h

```
1  /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
```

```
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_TINYLEVEL1_ACXPY_H
34 #define CXXBLAS_TINYLEVEL1_ACXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     template<int n, typename ALPHA, typename X, int incX, typename Y, int incY>
41     void acxpy(const ALPHA &alpha, const X *x, Y *y);
42
43 } // namespace cxxblas
44
45 #endif // CXXBLAS_TINYLEVEL1_ACXPY_H
```

7.41 asum1.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_ASUM_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_ASUM_H 1
35
36 #include "cxxblas/typedefs.h"
37 #include "cxxblas/drivers/drivers.h"
38
39 #define HAVE_CXXBLAS_ASUM1 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename X, typename T>
44     void asum1(IndexType n, const X *x, IndexType incX, T &absSum);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_ASUM_H
```

7.42 axpby.h

```
1 /*
2  *   Copyright (c) 2013, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
```

```
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1_AXPBY_H
34 #define CXXBLAS_LEVEL1_AXPBY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_AXPBY 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename X, typename BETA, typename Y>
43     void axpby(IndexType n, const ALPHA &alpha, const X *x, IndexType incX, const BETA &beta, Y *y,
44                IndexType incY);
45
46 #ifdef HAVE_CBLAS_AXPBY
47     // saxpy
48     template<typename IndexType>
49     typename If<IndexType>::isBlasCompatibleInteger
50     axpby(IndexType n, const float &alpha, const float *x, IndexType incX, const float &beta,
51           float *y, IndexType incY);
52
53     // daxpy
54     template<typename IndexType>
55     typename If<IndexType>::isBlasCompatibleInteger
56     axpby(IndexType n, const double &alpha, const double *x, IndexType incX, const double &beta,
57           double *y, IndexType incY);
58
59     // caxpy
60     template<typename IndexType>
61     typename If<IndexType>::isBlasCompatibleInteger
62     axpby(IndexType n, const ComplexFloat &alpha, const ComplexFloat *x, IndexType incX,
63           const ComplexFloat &beta, ComplexFloat *y, IndexType incY);
64
65     // zaxpy
66     template<typename IndexType>
67     typename If<IndexType>::isBlasCompatibleInteger
68     axpby(IndexType n, const ComplexDouble &alpha, const ComplexDouble *x, IndexType incX,
69           const ComplexDouble &beta, ComplexDouble *y, IndexType incY);
70
71 #endif // HAVE_CBLAS_AXPBY
72
73 } // namespace cxxblas
74
75 #endif // CXXBLAS_LEVEL1_AXPBY_H
```

7.43 axpby.h

```
1  /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
```

```
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_TINYLEVEL1_AXPBY_H
34 #define CXXBLAS_TINYLEVEL1_AXPBY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     template<int n, typename ALPHA, typename X, int incX, typename BETA, typename Y, int incY>
41     void axpby(const ALPHA &alpha, const X *x, const BETA &beta, Y *y);
42
43 } // namespace cxxblas
44
45 #endif // CXXBLAS_TINYLEVEL1_AXPBY_H
```

7.44 ccopy.h

```
1  /*
2   *   Copyright (c) 2012, Michael Lehn, Klaus Pototzky
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_CCOCPY_H
34 #   define CXXBLAS_LEVEL1EXTENSIONS_CCOPY_H 1
35
36 #   include "cxxblas/typedefs.h"
37
38 #   define HAVE_CXXBLAS_CCOPY 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename X, typename Y>
43     void ccopy(IndexType n, const X *x, IndexType incX, Y *y, IndexType incY);
44
45 } // namespace cxxblas
46
47 #endif // CXXBLAS_LEVEL1EXTENSIONS_CCOPY_H
```

7.45 ccopy.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_TINYLEVEL1_CCOPY_H
34 #define CXXBLAS_TINYLEVEL1_CCOPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     template<int n, typename X, int incX, typename Y, int incY>
41     void ccopy(const X *x, Y *y);
42
43 } // namespace cxxblas
44
45 #endif // CXXBLAS_TINYLEVEL1_CCOPY_H
```

7.46 gbaxpby.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GBAXPBY_H
```

```
34 #define CXXBLAS_LEVEL1EXTENSIONS_GBAXPBY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GBAXPBY 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename BETA, typename MB>
43     void gbaxpby(StorageOrder order, Transpose trans, IndexType m, IndexType n, IndexType kl,
44                  IndexType ku, const ALPHA &alpha, const MA *A, IndexType ldA, const BETA &beta,
45                  MB *B, IndexType ldB);
46
47 } // namespace cxxblas
48
49 #endif // CXXBLAS_LEVEL1EXTENSIONS_GBAXPBY_H
```

7.47 gbaxpy.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GBAXPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GBAXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GBAXPY 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename MB>
43     void gbaxpy(StorageOrder order, Transpose trans, IndexType m, IndexType n, IndexType kl,
44                 IndexType ku, const ALPHA &alpha, const MA *A, IndexType ldA, MB *B, IndexType ldB);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_GBAXPY_H
```

7.48 gbcopy.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
```

```
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GBCOPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GBCOPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GBCOPY 1
39
40 namespace cxxblas {
41
42     //
43     //  B = A  or B = A^T
44     //
45     template<typename IndexType, typename MA, typename MB>
46     void gbcopy(StorageOrder order, Transpose trans, IndexType m, IndexType n, IndexType kl,
47                 IndexType ku, const MA *A, IndexType ldA, MB *B, IndexType ldB);
48
49 } // namespace cxxblas
50
51 #endif // CXXBLAS_LEVEL1EXTENSIONS_GBCOPY_H
```

7.49 gbcotr.h

```
1  /*
2   *   Copyright (c) 2012, Michael Lehn, Klaus Pototzky
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GBCOTR_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GBCOTR_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GBCOTR 1
```

```
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename MA>
43     void gbcotr(StorageOrder order, Transpose trans, IndexType m, IndexType n, IndexType kl,
44                 IndexType ku, MA *A, IndexType ldA);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_GBCOTR_H
```

7.50 gbscal.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GBSCAL_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GBSCAL_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GBSCAL 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA>
43     void gbscal(StorageOrder order, IndexType m, IndexType n, IndexType kl, IndexType ku,
44                 const ALPHA &alpha, MA *A, IndexType ldA);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_GBSCAL_H
```

7.51 geaxpby.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
```

```
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GEAXPBY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GEAXPBY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GEAXPBY 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename BETA, typename MB>
43     void geaxpby(StorageOrder order, Transpose trans, IndexType m, IndexType n, const ALPHA &alpha,
44                  const MA *A, IndexType ldA, const BETA &beta, MB *B, IndexType ldB);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_GEAXPBY_H
```

7.52 geaxpy.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GEAXPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GEAXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GEAXPY 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename MB>
43     void geaxpy(StorageOrder order, Transpose trans, IndexType m, IndexType n, const ALPHA &alpha,
44                 const MA *A, IndexType ldA, MB *B, IndexType ldB);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_GEAXPY_H
```

7.53 geaxpy.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_TINYLEVEL1_GEAXPY_H
34 #define CXXBLAS_TINYLEVEL1_GEAXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GEAXPY 1
39
40 namespace cxxblas {
41
42     template<int m, int n, typename ALPHA, typename MA, int ldA, typename MB, int ldB>
43     void geaxpy(Transpose trans, const ALPHA &alpha, const MA *A, MB *B);
44
45 } // namespace cxxblas
46
47 #endif // CXXBLAS_TINYLEVEL1_GEAXPY_H
```

7.54 gecopy.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
```

```
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GECOPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GECOPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GECOPY 1
39
40 namespace cxxblas {
41
42     //
43     //  B = A  or B = A^T
44     //
45     template<typename IndexType, typename MA, typename MB>
46     void gecopy(StorageOrder order, Transpose trans, IndexType m, IndexType n, const MA *A,
47                 IndexType ldA, MB *B, IndexType ldB);
48
49 } // namespace cxxblas
50
51 #endif // CXXBLAS_LEVEL1EXTENSIONS_GECOPY_H
```

7.55 gecopy.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_TINYLEVEL1_GECOPY_H
34 #define CXXBLAS_TINYLEVEL1_GECOPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     //
41     //  B = A  or B = A^T
42     //
43     template<int m, int n, typename MA, int ldA, typename MB, int ldB>
44     void gecopy(Transpose trans, const MA *A, MB *B);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_TINYLEVEL1_GECOPY_H
```

7.56 gecotr.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
```

```
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GECOTR_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GECOTR_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GECOTR 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename MA>
43     void gecotr(StorageOrder order, Transpose trans, IndexType m, IndexType n, MA *A,
44                 IndexType ldA);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_GECOTR_H
```

7.57 geraxpy.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GERAXPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GERAXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
```

```
38 #define HAVE_CXXBLAS_GERAXPY 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename MB>
43     void geraxpy(StorageOrder order, Transpose trans, IndexType m, IndexType n, const ALPHA &alpha,
44                  const MA *A, IndexType ldA, MB *B, IndexType ldB);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_GERAXPY_H
```

7.58 gerscal.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GERSCAL_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GERSCAL_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GERSCAL 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA>
43     void gerscal(StorageOrder order, IndexType m, IndexType n, const ALPHA &alpha, MA *A,
44                  IndexType ldA);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_GERSCAL_H
```

7.59 gerscal.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
```

```
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_TINYLEVEL1_GERSCAL_H
34 #define CXXBLAS_TINYLEVEL1_GERSCAL_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     template<int m, int n, typename ALPHA, typename MA, int ldA>
41     void gerscal(const ALPHA &alpha, MA *A);
42
43 } // namespace cxxblas
44
45 #endif // CXXBLAS_TINYLEVEL1_GERSCAL_H
```

7.60 gescal.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GESCAL_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GESCAL_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GESCAL 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA>
43     void gescal_init(StorageOrder order, IndexType m, IndexType n, const ALPHA &alpha, MA *A,
44                      IndexType ldA);
45
46     template<typename IndexType, typename ALPHA, typename MA>
47     void gescal(StorageOrder order, IndexType m, IndexType n, const ALPHA &alpha, MA *A,
48                 IndexType ldA);
49
```

```
50 } // namespace cxxblas
51
52 #endif // CXXBLAS_LEVEL1EXTENSIONS_GESCAL_H
```

7.61 gescal.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_TINYLEVEL1_GESCAL_H
34 #define CXXBLAS_TINYLEVEL1_GESCAL_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     template<int m, int n, typename ALPHA, typename MA, int ldA>
41     void gescal(const ALPHA &alpha, MA *A);
42
43 } // namespace cxxblas
44
45 #endif // CXXBLAS_TINYLEVEL1_GESCAL_H
```

7.62 geswap.h

```
1 /*
2  *   Copyright (c) 2013, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
```

```
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_GESWAP_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_GESWAP_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GESWAP 1
39
40 namespace cxxblas {
41
42     //
43     //  swap A and B
44     //
45     template<typename IndexType, typename MA, typename MB>
46     void geswap(StorageOrder orderA, StorageOrder orderB, IndexType m, IndexType n, MA *A,
47                 IndexType ldA, MB *B, IndexType ldB);
48
49 } // namespace cxxblas
50
51 #endif // CXXBLAS_LEVEL1EXTENSIONS_GESWAP_H
```

7.63 hescal.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_HESCAL_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_HESCAL_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_HESCAL 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA>
43     void hescal(StorageOrder order, StorageUpLo upLoA, IndexType n, const ALPHA &alpha, MA *A,
44                 IndexType ldA);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_HESCAL_H
```

7.64 imax1.h

```
1  /*
```

```
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_IMAX_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_IMAX_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_IMAX1 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename X>
43     void imax1(IndexType n, const X *x, IndexType incX, IndexType &i);
44
45     template<typename IndexType, typename X>
46     IndexType imax1(IndexType n, const X *x, IndexType incX);
47
48 } // namespace cxxblas
49
50 #endif // CXXBLAS_LEVEL1EXTENSIONS_IMAX_H
```

7.65 level1extensions.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
```

```
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_LEVEL1EXTENSIONS_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_LEVEL1EXTENSIONS_H 1
35
36 #include "cxxblas/level1extensions/acxpby.h"
37 #include "cxxblas/level1extensions/acxpy.h"
38 #include "cxxblas/level1extensions/asum1.h"
39 #include "cxxblas/level1extensions/axpby.h"
40 #include "cxxblas/level1extensions/axpy.h"
41 #include "cxxblas/level1extensions/ccopy.h"
42 #include "cxxblas/level1extensions/dot.h"
43 #include "cxxblas/level1extensions/gbaxpby.h"
44 #include "cxxblas/level1extensions/gbaxpy.h"
45 #include "cxxblas/level1extensions/gbcopy.h"
46 #include "cxxblas/level1extensions/gbcotr.h"
47 #include "cxxblas/level1extensions/gbscal.h"
48 #include "cxxblas/level1extensions/geaxpby.h"
49 #include "cxxblas/level1extensions/geaxpy.h"
50 #include "cxxblas/level1extensions/gecopy.h"
51 #include "cxxblas/level1extensions/gecotr.h"
52 #include "cxxblas/level1extensions/geraxpy.h"
53 #include "cxxblas/level1extensions/gerscal.h"
54 #include "cxxblas/level1extensions/gescal.h"
55 #include "cxxblas/level1extensions/geswap.h"
56 #include "cxxblas/level1extensions/hescal.h"
57 #include "cxxblas/level1extensions/imax1.h"
58 #include "cxxblas/level1extensions/syscal.h"
59 #include "cxxblas/level1extensions/racxpy.h"
60 #include "cxxblas/level1extensions/raxpy.h"
61 #include "cxxblas/level1extensions/rscal.h"
62 #include "cxxblas/level1extensions/traxpby.h"
63 #include "cxxblas/level1extensions/traxpy.h"
64 #include "cxxblas/level1extensions/trcopy.h"
65 #include "cxxblas/level1extensions/tpaxpby.h"
66 #include "cxxblas/level1extensions/tpaxpy.h"
67 #include "cxxblas/level1extensions/tpcopy.h"
68 #include "cxxblas/level1extensions/tpscal.h"
69
70 #endif // CXXBLAS_LEVEL1EXTENSIONS_LEVEL1EXTENSIONS_H
```

7.66 racxpy.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_RACXPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_RACXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_RACXPY 1
39
40 namespace cxxblas {
41
```

```
42     template<typename IndexType, typename ALPHA, typename X, typename Y>
43     void racxpy(IndexType n, const ALPHA &alpha, const X *x, IndexType incX, Y *y, IndexType incY);
44
45 } // namespace cxxblas
46
47 #endif // CXXBLAS_LEVEL1EXTENSIONS_RACXPY_H
```

7.67 raxpy.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_RAXPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_RAXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_RAXPY 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename X, typename Y>
43     void raxpy(IndexType n, const ALPHA &alpha, const X *x, IndexType incX, Y *y, IndexType incY);
44
45 } // namespace cxxblas
46
47 #endif // CXXBLAS_LEVEL1EXTENSIONS_RAXPY_H
```

7.68 rscal.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
```

```
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_RSCAL_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_RSCAL_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_RSCAL 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename Y>
43     void rscal(IndexType n, const ALPHA &alpha, Y *y, IndexType incY);
44
45 } // namespace cxxblas
46
47 #endif // CXXBLAS_LEVEL1EXTENSIONS_RSCAL_H
```

7.69 rscal.h

```
1  /*
2   *   Copyright (c) 2012, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_TINYLEVEL1_RSCAL_H
34 #define CXXBLAS_TINYLEVEL1_RSCAL_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     template<int n, typename ALPHA, typename Y, int incY>
41     void rscal(const ALPHA &alpha, Y *y);
42
43 } // namespace cxxblas
44
45 #endif // CXXBLAS_TINYLEVEL1_RSCAL_H
```

7.70 syscal.h

```
1  /*
```

```
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_SYSCAL_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_SYSCAL_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_SYSCAL 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA>
43     void syscal(StorageOrder order, StorageUpLo upLoA, IndexType n, const ALPHA &alpha, MA *A,
44                 IndexType ldA);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL1EXTENSIONS_SYSCAL_H
```

7.71 tpaxpby.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_TPAXPBY_H
```

```
34 #define CXXBLAS_LEVEL1EXTENSIONS_TPAXPBY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_TPAXPBY 1
39
40 namespace cxxblas {
41
42     //
43     //  B = beta*B + alpha*op(A)
44     //
45     //  where B is a nxn triangular packed matrix as specified by upLo
46     //
47     template<typename IndexType, typename ALPHA, typename MA, typename BETA, typename MB>
48     void tpaxpby(StorageOrder order, StorageUpLo upLo, Transpose trans, Diag diag, IndexType n,
49                  const ALPHA &alpha, const MA *A, const BETA &beta, MB *B);
50
51 } // namespace cxxblas
52
53 #endif // CXXBLAS_LEVEL1EXTENSIONS_TPAXPBY_H
```

7.72 tpaxpy.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_TPAXPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_TPAXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_TPAXPY 1
39
40 namespace cxxblas {
41
42     //
43     //  B += alpha*A  or B += alpha*A^T
44     //
45     //  where B is a nxn triangular packed matrix as specified by upLo
46     //
47     template<typename IndexType, typename ALPHA, typename MA, typename MB>
48     void tpaxpy(StorageOrder order, StorageUpLo upLo, Transpose trans, Diag diag, IndexType n,
49                 const ALPHA &alpha, const MA *A, MB *B);
50
51 } // namespace cxxblas
52
53 #endif // CXXBLAS_LEVEL1EXTENSIONS_TPAXPY_H
```

7.73 tpcopy.h

```
1 /*
```

```
2  *   Copyright (c) 2012, Michael Lehn, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_TPCOPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_TPCOPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_TPCOPY 1
39
40 namespace cxxblas {
41
42     //
43     //  B = A  or B = A^T
44     //
45     //  where B is a nxn triangular packed matrix as specified by upLo
46     //
47     template<typename IndexType, typename MA, typename MB>
48     void tpcopy(StorageUpLo upLo, Transpose trans, Diag diag, IndexType n, const MA *A, MB *B);
49
50 } // namespace cxxblas
51
52 #endif // CXXBLAS_LEVEL1EXTENSIONS_TPCOPY_H
```

7.74 tpscal.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
```

```
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_TPSCAL_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_TPSCAL_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_TPSCAL 1
39
40 namespace cxxblas {
41
42     //
43     //  B = alpha*B
44     //
45     template<typename IndexType, typename ALPHA, typename MA>
46     void tpscal(Diag diag, IndexType n, const ALPHA &alpha, MA *A);
47
48 } // namespace cxxblas
49
50 #endif // CXXBLAS_LEVEL1EXTENSIONS_TPSCAL_H
```

7.75 traxpby.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_TRAXPBY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_TRAXPBY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_TRAXPBY 1
39
40 namespace cxxblas {
41
42     //
43     //  B = beta*B + alpha*op(A)
44     //
45     //  where B is a mxn triangular matrix as specified by upLo
46     //
47     template<typename IndexType, typename ALPHA, typename MA, typename BETA, typename MB>
48     void traxpby(StorageOrder order, StorageUpLo upLo, Transpose trans, Diag diag, IndexType m,
49                  IndexType n, const ALPHA &alpha, const MA *A, IndexType ldA, const BETA &beta,
50                  MB *B, IndexType ldB);
51
52 } // namespace cxxblas
53
54 #endif // CXXBLAS_LEVEL1EXTENSIONS_TRAXPBY_H
```

7.76 traxpy.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_TRAXPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_TRAXPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_TRAXPY 1
39
40 namespace cxxblas {
41
42     //
43     //  B += alpha*A  or B += alpha*A^T
44     //
45     //  where B is a mxn triangular matrix as specified by upLo
46     //
47     template<typename IndexType, typename ALPHA, typename MA, typename MB>
48     void traxpy(StorageOrder order, StorageUpLo upLo, Transpose trans, Diag diag, IndexType m,
49                 IndexType n, const ALPHA &alpha, const MA *A, IndexType ldA, MB *B, IndexType ldB);
50
51 } // namespace cxxblas
52
53 #endif // CXXBLAS_LEVEL1EXTENSIONS_TRAXPY_H
```

7.77 trcopy.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
```

```
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_TRCOPY_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_TRCOPY_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_TRCOPY 1
39
40 namespace cxxblas {
41
42     //
43     //  B = A  or B = A^T
44     //
45     //  where B is a mxn triangular matrix as specified by upLo
46     //
47     template<typename IndexType, typename MA, typename MB>
48     void trcopy(StorageOrder order, StorageUpLo upLo, Transpose trans, Diag diag, IndexType m,
49                 IndexType n, const MA *A, IndexType ldA, MB *B, IndexType ldB);
50
51 } // namespace cxxblas
52
53 #endif // CXXBLAS_LEVEL1EXTENSIONS_TRCOPY_H
```

7.78 trscal.h

```
1  /*
2   *   Copyright (c) 2012, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL1EXTENSIONS_TRSCAL_H
34 #define CXXBLAS_LEVEL1EXTENSIONS_TRSCAL_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_TRSCAL 1
39
40 namespace cxxblas {
41
42     //
43     //  B = alpha*A
44     //
45     template<typename IndexType, typename ALPHA, typename MA>
46     void trscal(StorageOrder order, StorageUpLo upLo, Diag diag, IndexType m, IndexType n,
47                 const ALPHA &alpha, MA *A, IndexType ldA);
48
49 } // namespace cxxblas
50
51 #endif // CXXBLAS_LEVEL1EXTENSIONS_TRSCAL_H
```

7.79 gbmv.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2_GBMV_H
34 #define CXXBLAS_LEVEL2_GBMV_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_GBMV 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
44              typename VY>
45     void gbmv(StorageOrder order, Transpose trans, IndexType m, IndexType n, IndexType kl,
46               IndexType ku, const ALPHA &alpha, const MA *A, IndexType ldA, const VX *x,
47               IndexType incX, const BETA &beta, VY *y, IndexType incY);
48
49 #ifdef HAVE_CBLAS
50
51     // sgbmv
52     template<typename IndexType>
53     typename If<IndexType>::isBlasCompatibleInteger
54     gbmv(StorageOrder order, Transpose trans, IndexType m, IndexType n, IndexType kl, IndexType ku,
55          float alpha, const float *A, IndexType ldA, const float *x, IndexType incX, float beta,
56          float *y, IndexType incY);
57
58     // dgbmv
59     template<typename IndexType>
60     typename If<IndexType>::isBlasCompatibleInteger
61     gbmv(StorageOrder order, Transpose trans, IndexType m, IndexType n, IndexType kl, IndexType ku,
62          double alpha, const double *A, IndexType ldA, const double *x, IndexType incX, double beta,
63          double *y, IndexType incY);
64
65     // cgbmv
66     template<typename IndexType>
67     typename If<IndexType>::isBlasCompatibleInteger
68     gbmv(StorageOrder order, Transpose trans, IndexType m, IndexType n, IndexType kl, IndexType ku,
69          const ComplexFloat &alpha, const ComplexFloat *A, IndexType ldA, const ComplexFloat *x,
70          IndexType incX, const ComplexFloat &beta, ComplexFloat *y, IndexType incY);
71
72     // zgbmv
73     template<typename IndexType>
74     typename If<IndexType>::isBlasCompatibleInteger
75     gbmv(StorageOrder order, Transpose trans, IndexType m, IndexType n, IndexType kl, IndexType ku,
76          const ComplexDouble &alpha, const ComplexDouble *A, IndexType ldA, const ComplexDouble *x,
77          IndexType incX, const ComplexDouble &beta, ComplexDouble *y, IndexType incY);
78
79 #endif // HAVE_CBLAS
80 } // namespace cxxblas
81
82 #endif // CXXBLAS_LEVEL2_GBMV_H
```

7.80 gbmv.h

```
1 /*
2  *   Copyright (c) 2012, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2EXTENSIONS_GBMV_H
34 #define CXXBLAS_LEVEL2EXTENSIONS_GBMV_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GBMV 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
43              typename VY>
44     void gbmv(StorageOrder order, Transpose trans, Transpose conjX, IndexType m, IndexType n,
45               IndexType ku, IndexType kl, const ALPHA &alpha, const MA *A, IndexType ldA,
46               const VX *x, IndexType incX, const BETA &beta, VY *y, IndexType incY);
47
48 } // namespace cxxblas
49
50 #endif // CXXBLAS_LEVEL2EXTENSIONS_GEMV_H
```

7.81 gemv.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
```

```
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL2_GEMV_H
34 #define CXXBLAS_LEVEL2_GEMV_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_GEMV 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
44              typename VY>
45     void gemv(StorageOrder order, Transpose trans, IndexType m, IndexType n, const ALPHA &alpha,
46               const MA *A, IndexType ldA, const VX *x, IndexType incX, const BETA &beta, VY *y,
47               IndexType incY);
48
49 #ifdef HAVE_CBLAS
50
51     // sgemv
52     template<typename IndexType>
53     typename If<IndexType>::isBlasCompatibleInteger
54     gemv(StorageOrder order, Transpose trans, IndexType m, IndexType n, float alpha, const float *A,
55          IndexType ldA, const float *x, IndexType incX, float beta, float *y, IndexType incY);
56
57     // dgemv
58     template<typename IndexType>
59     typename If<IndexType>::isBlasCompatibleInteger
60     gemv(StorageOrder order, Transpose trans, IndexType m, IndexType n, double alpha,
61          const double *A, IndexType ldA, const double *x, IndexType incX, double beta, double *y,
62          IndexType incY);
63
64     // cgemv
65     template<typename IndexType>
66     typename If<IndexType>::isBlasCompatibleInteger
67     gemv(StorageOrder order, Transpose trans, IndexType m, IndexType n, const ComplexFloat &alpha,
68          const ComplexFloat *A, IndexType ldA, const ComplexFloat *x, IndexType incX,
69          const ComplexFloat &beta, ComplexFloat *y, IndexType incY);
70
71     // zgemv
72     template<typename IndexType>
73     typename If<IndexType>::isBlasCompatibleInteger
74     gemv(StorageOrder order, Transpose trans, IndexType m, IndexType n, const ComplexDouble &alpha,
75          const ComplexDouble *A, IndexType ldA, const ComplexDouble *x, IndexType incX,
76          const ComplexDouble &beta, ComplexDouble *y, IndexType incY);
77
78 #endif // HAVE_CBLAS
79
80 } // namespace cxxblas
81
82 #endif // CXXBLAS_LEVEL2_GEMV_H
```

7.82 gemv.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
```

```
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL2EXTENSIONS_GEMV_H
34 #define CXXBLAS_LEVEL2EXTENSIONS_GEMV_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GEMV 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
43              typename VY>
44     void gemv(StorageOrder order, Transpose transA, Transpose conjX, IndexType m, IndexType n,
45               const ALPHA &alpha, const MA *A, IndexType ldA, const VX *x, IndexType incX,
46               const BETA &beta, VY *y, IndexType incY);
47
48 #ifdef HAVE_CBLAS
49
50     template<typename IndexType>
51     void gemv(StorageOrder order, Transpose transA, IndexType m, IndexType n, const float &alpha,
52               const float *A, IndexType ldA, const float *x, IndexType incX, const float &beta,
53               std::complex<float> *y, IndexType incY);
54
55     template<typename IndexType>
56     void gemv(StorageOrder order, Transpose transA, IndexType m, IndexType n, const float &alpha,
57               const float *A, IndexType ldA, const float *x, IndexType incX,
58               const std::complex<float> &beta, std::complex<float> *y, IndexType incY);
59
60     template<typename IndexType>
61     void gemv(StorageOrder order, Transpose transA, IndexType m, IndexType n, const float &alpha,
62               const float *A, IndexType ldA, const std::complex<float> *x, IndexType incX,
63               const float &beta, std::complex<float> *y, IndexType incY);
64
65     template<typename IndexType>
66     void gemv(StorageOrder order, Transpose transA, IndexType m, IndexType n, const float &alpha,
67               const float *A, IndexType ldA, const std::complex<float> *x, IndexType incX,
68               const std::complex<float> &beta, std::complex<float> *y, IndexType incY);
69
70     template<typename IndexType>
71     void gemv(StorageOrder order, Transpose transA, IndexType m, IndexType n, const double &alpha,
72               const double *A, IndexType ldA, const double *x, IndexType incX, const double &beta,
73               std::complex<double> *y, IndexType incY);
74
75     template<typename IndexType>
76     void gemv(StorageOrder order, Transpose transA, IndexType m, IndexType n, const double &alpha,
77               const double *A, IndexType ldA, const double *x, IndexType incX,
78               const std::complex<double> &beta, std::complex<double> *y, IndexType incY);
79
80     template<typename IndexType>
81     void gemv(StorageOrder order, Transpose transA, IndexType m, IndexType n, const double &alpha,
82               const double *A, IndexType ldA, const std::complex<double> *x, IndexType incX,
83               const double &beta, std::complex<double> *y, IndexType incY);
84
85     template<typename IndexType>
86     void gemv(StorageOrder order, Transpose transA, IndexType m, IndexType n, const double &alpha,
87               const double *A, IndexType ldA, const std::complex<double> *x, IndexType incX,
88               const std::complex<double> &beta, std::complex<double> *y, IndexType incY);
89
90 #endif
91 } // namespace cxxblas
92
93 #endif // CXXBLAS_LEVEL2EXTENSIONS_GEMV_H
```

7.83 gemv.h

```
1  /*
2   *   Copyright (c) 2012, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
```

```
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_TINYLEVEL2_GEMV_H
34 #define CXXBLAS_TINYLEVEL2_GEMV_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     //
41     //  B = A  or B = A^T
42     //
43     template<int m, int n, typename MA, int ldA, typename VX, int incX, typename VY, int incY>
44     void gemv(Transpose trans, MA alpha, const MA *A, const VX *x, VY beta, VY *y);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_TINYLEVEL2_GEMV_H
```

7.84 ger.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL2_GER_H
34 #define CXXBLAS_LEVEL2_GER_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_GER 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename VX, typename VY, typename MA>
44     void ger(StorageOrder order, IndexType m, IndexType n, const ALPHA &alpha, const VX *x,
45              IndexType incX, const VY *y, IndexType incY, MA *A, IndexType ldA);
```

```
46
47      template<typename IndexType, typename ALPHA, typename VX, typename VY, typename MA>
48      void geru(StorageOrder order, IndexType m, IndexType n, const ALPHA &alpha, const VX *x,
49                IndexType incX, const VY *y, IndexType incY, MA *A, IndexType ldA);
50
51      template<typename IndexType, typename ALPHA, typename VX, typename VY, typename MA>
52      void gerc(StorageOrder order, IndexType m, IndexType n, const ALPHA &alpha, const VX *x,
53                IndexType incX, const VY *y, IndexType incY, MA *A, IndexType ldA);
54
55  #ifdef HAVE_CBLAS
56
57      // sger
58      template<typename IndexType>
59      typename If<IndexType>::isBlasCompatibleInteger
60      ger(StorageOrder order, IndexType m, IndexType n, const float &alpha, const float *x,
61          IndexType incX, const float *y, IndexType incY, float *A, IndexType ldA);
62
63      // dger
64      template<typename IndexType>
65      typename If<IndexType>::isBlasCompatibleInteger
66      ger(StorageOrder order, IndexType m, IndexType n, const double &alpha, const double *x,
67          IndexType incX, const double *y, IndexType incY, double *A, IndexType ldA);
68
69      // cgeru
70      template<typename IndexType>
71      typename If<IndexType>::isBlasCompatibleInteger
72      geru(StorageOrder order, IndexType m, IndexType n, const ComplexFloat &alpha,
73           const ComplexFloat *x, IndexType incX, const ComplexFloat *y, IndexType incY,
74           ComplexFloat *A, IndexType ldA);
75
76      // zgeru
77      template<typename IndexType>
78      typename If<IndexType>::isBlasCompatibleInteger
79      geru(StorageOrder order, IndexType m, IndexType n, const ComplexDouble &alpha,
80           const ComplexDouble *x, IndexType incX, const ComplexDouble *y, IndexType incY,
81           ComplexDouble *A, IndexType ldA);
82
83      // cgerc
84      template<typename IndexType>
85      typename If<IndexType>::isBlasCompatibleInteger
86      gerc(StorageOrder order, IndexType m, IndexType n, const ComplexFloat &alpha,
87           const ComplexFloat *x, IndexType incX, const ComplexFloat *y, IndexType incY,
88           ComplexFloat *A, IndexType ldA);
89
90      // zgerc
91      template<typename IndexType>
92      typename If<IndexType>::isBlasCompatibleInteger
93      gerc(StorageOrder order, IndexType m, IndexType n, const ComplexDouble &alpha,
94           const ComplexDouble *x, IndexType incX, const ComplexDouble *y, IndexType incY,
95           ComplexDouble *A, IndexType ldA);
96
97  #endif // HAVE_CBLAS
98
99  } // namespace cxxblas
100
101 #endif // CXXBLAS_LEVEL2_GER_H
```

7.85 hbmv.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
```

```
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL2_HBMV_H
34 #define CXXBLAS_LEVEL2_HBMV_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_HBMV 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
44              typename VY>
45     void hbmv(StorageOrder order, StorageUpLo upLo, IndexType n, IndexType k, const ALPHA &alpha,
46               const MA *A, IndexType ldA, const VX *x, IndexType incX, const BETA &beta, VY *y,
47               IndexType incY);
48
49 #ifdef HAVE_CBLAS
50
51     // chbmv
52     template<typename IndexType>
53     typename If<IndexType>::isBlasCompatibleInteger
54     hbmv(StorageOrder order, Transpose trans, IndexType n, IndexType k, const ComplexFloat &alpha,
55          const ComplexFloat *A, IndexType ldA, const ComplexFloat *x, IndexType incX,
56          const ComplexFloat &beta, ComplexFloat *y, IndexType incY);
57
58     // zhbmv
59     template<typename IndexType>
60     typename If<IndexType>::isBlasCompatibleInteger
61     hbmv(StorageOrder order, Transpose trans, IndexType n, IndexType k, const ComplexDouble &alpha,
62          const ComplexDouble *A, IndexType ldA, const ComplexDouble *x, IndexType incX,
63          const ComplexDouble &beta, ComplexDouble *y, IndexType incY);
64
65 #endif // HAVE_CBLAS
66
67 } // namespace cxxblas
68
69 #endif // CXXBLAS_LEVEL2_HBMV_H
```

7.86 hemv.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL2_HEMV_H
34 #define CXXBLAS_LEVEL2_HEMV_H 1
35
```

```
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_HEMV 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
44              typename VY>
45     void hemv(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const MA *A,
46               IndexType ldA, const VX *x, IndexType incX, const BETA &beta, VY *y, IndexType incY);
47
48 #ifdef HAVE_CBLAS
49
50     // chemv
51     template<typename IndexType>
52     typename If<IndexType>::isBlasCompatibleInteger
53     hemv(StorageOrder order, StorageUpLo upLo, IndexType n, const ComplexFloat &alpha,
54          const ComplexFloat *A, IndexType ldA, const ComplexFloat *x, IndexType incX,
55          const ComplexFloat &beta, ComplexFloat *y, IndexType incY);
56
57     // zhemv
58     template<typename IndexType>
59     typename If<IndexType>::isBlasCompatibleInteger
60     hemv(StorageOrder order, StorageUpLo upLo, IndexType n, const ComplexDouble &alpha,
61          const ComplexDouble *A, IndexType ldA, const ComplexDouble *x, IndexType incX,
62          const ComplexDouble &beta, ComplexDouble *y, IndexType incY);
63
64 #endif // HAVE_CBLAS
65
66 } // namespace cxxblas
67
68 #endif // CXXBLAS_LEVEL2_HEMV_H
```

7.87 hemv.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2EXTENSIONS_HEMV_H
34 #define CXXBLAS_LEVEL2EXTENSIONS_HEMV_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
41              typename VY>
42     void hemv(StorageOrder order, StorageUpLo upLo, Transpose conjugateA, IndexType n,
43               const ALPHA &alpha, const MA *A, IndexType ldA, const VX *x, IndexType incX,
44               const BETA &beta, VY *y, IndexType incY);
45
46 } // namespace cxxblas
47
48 #endif // CXXBLAS_LEVEL2EXTENSIONS_HEMV_H
```

7.88 her.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2_HER_H
34 #define CXXBLAS_LEVEL2_HER_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_HER 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename VX, typename MA>
44     void her(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const VX *x,
45              IndexType incX, MA *A, IndexType ldA);
46
47 #ifdef HAVE_CBLAS
48     // cher
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger
51     her(StorageOrder order, StorageUpLo upLo, IndexType n, float alpha, const ComplexFloat *x,
52         IndexType incX, ComplexFloat *A, IndexType ldA);
53
54     // zher
55     template<typename IndexType>
56     typename If<IndexType>::isBlasCompatibleInteger
57     her(StorageOrder order, StorageUpLo upLo, IndexType n, double alpha, const ComplexDouble *x,
58         IndexType incX, ComplexDouble *A, IndexType ldA);
59
60 #endif // HAVE_CBLAS
61
62 } // namespace cxxblas
63
64 #endif // CXXBLAS_LEVEL2_HER_H
```

7.89 her.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
```

```
 *      distribution.
 *   3) Neither the name of the FLENS development group nor the names of
 *      its contributors may be used to endorse or promote products derived
 *      from this software without specific prior written permission.
 *
 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_LEVEL2EXTENSIONS_HER_H
#define CXXBLAS_LEVEL2EXTENSIONS_HER_H 1

#include "cxxblas/typedefs.h"

namespace cxxblas {

    template<typename IndexType, typename ALPHA, typename VX, typename MA>
    void her(StorageOrder order, StorageUpLo upLo, Transpose conjugateA, IndexType n,
             const ALPHA &alpha, const VX *x, IndexType incX, MA *A, IndexType ldA);

} // namespace cxxblas

#endif // CXXBLAS_LEVEL2EXTENSIONS_HER_H
```

7.90 her2.h

```
/*
 *   Copyright (c) 2009, Michael Lehn
 *
 *   All rights reserved.
 *
 *   Redistribution and use in source and binary forms, with or without
 *   modification, are permitted provided that the following conditions
 *   are met:
 *
 *   1) Redistributions of source code must retain the above copyright
 *      notice, this list of conditions and the following disclaimer.
 *   2) Redistributions in binary form must reproduce the above copyright
 *      notice, this list of conditions and the following disclaimer in
 *      the documentation and/or other materials provided with the
 *      distribution.
 *   3) Neither the name of the FLENS development group nor the names of
 *      its contributors may be used to endorse or promote products derived
 *      from this software without specific prior written permission.
 *
 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_LEVEL2_HER2_H
#define CXXBLAS_LEVEL2_HER2_H 1

#include "cxxblas/drivers/drivers.h"
#include "cxxblas/typedefs.h"

#define HAVE_CXXBLAS_HER2 1

namespace cxxblas {

    template<typename IndexType, typename ALPHA, typename VX, typename VY, typename MA>
    void her2(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const VX *x,
              IndexType incX, const VY *y, IndexType incY, MA *A, IndexType ldA);

#ifdef HAVE_CBLAS
    // cher
```

```
49      template<typename IndexType>
50      typename If<IndexType>::isBlasCompatibleInteger
51      her2(StorageOrder order, StorageUpLo upLo, IndexType n, const ComplexFloat &alpha,
52           const ComplexFloat *x, IndexType incX, const ComplexFloat *y, IndexType incY,
53           ComplexFloat *A, IndexType ldA);
54
55      // zher
56      template<typename IndexType>
57      typename If<IndexType>::isBlasCompatibleInteger
58      her2(StorageOrder order, StorageUpLo upLo, IndexType n, const ComplexDouble &alpha,
59           const ComplexDouble *x, IndexType incX, const ComplexDouble *y, IndexType incY,
60           ComplexDouble *A, IndexType ldA);
61
62  #endif // HAVE_CBLAS
63
64  } // namespace cxxblas
65
66  #endif // CXXBLAS_LEVEL2_HER2_H
```

7.91 her2.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_LEVEL2EXTENSIONS_HER_H
34  #define CXXBLAS_LEVEL2EXTENSIONS_HER_H 1
35
36  #include "cxxblas/typedefs.h"
37
38  #define HAVE_CXXBLAS_HER2 1
39
40  namespace cxxblas {
41
42      template<typename IndexType, typename ALPHA, typename VX, typename VY, typename MA>
43      void her2(StorageOrder order, StorageUpLo upLo, Transpose conjugateA, IndexType n,
44                const ALPHA &alpha, const VX *x, IndexType incX, const VY *y, IndexType incY, MA *A,
45                IndexType ldA);
46
47  } // namespace cxxblas
48
49  #endif // CXXBLAS_LEVEL2EXTENSIONS_HER_H
```

7.92 hpmv.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
```

```
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2_HPMV_H
34 #define CXXBLAS_LEVEL2_HPMV_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_HPMV 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
44              typename VY>
45     void hpmv(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const MA *A,
46               const VX *x, IndexType incX, const BETA &beta, VY *y, IndexType incY);
47
48 #ifdef HAVE_CBLAS
49
50     // chpmv
51     template<typename IndexType>
52     typename If<IndexType>::isBlasCompatibleInteger
53     hpmv(StorageOrder order, StorageUpLo upLo, IndexType n, const ComplexFloat &alpha,
54          const ComplexFloat *A, const ComplexFloat *x, IndexType incX, const ComplexFloat &beta,
55          ComplexFloat *y, IndexType incY);
56
57     // zhpmv
58     template<typename IndexType>
59     typename If<IndexType>::isBlasCompatibleInteger
60     hpmv(StorageOrder order, StorageUpLo upLo, IndexType n, const ComplexDouble &alpha,
61          const ComplexDouble *A, const ComplexDouble *x, IndexType incX, const ComplexDouble &beta,
62          ComplexDouble *y, IndexType incY);
63
64 #endif // HAVE_CBLAS
65
66 } // namespace cxxblas
67
68 #endif // CXXBLAS_LEVEL2_HPMV_H
```

7.93 hpr.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
```

```
 *
 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_LEVEL2_HPR_H
#define CXXBLAS_LEVEL2_HPR_H 1

#include "cxxblas/drivers/drivers.h"
#include "cxxblas/typedefs.h"

#define HAVE_CXXBLAS_SPR 1

namespace cxxblas {

    template<typename IndexType, typename ALPHA, typename VX, typename MA>
    void hpr(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const VX *x,
             IndexType incX, MA *A);

#ifdef HAVE_CBLAS
    // chpr
    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    hpr(StorageOrder order, StorageUpLo upLo, IndexType n, float alpha, const ComplexFloat *x,
        IndexType incX, ComplexFloat *A);

    // zhpr
    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    hpr(StorageOrder order, StorageUpLo upLo, IndexType n, double alpha, const ComplexDouble *x,
        IndexType incX, ComplexDouble *A);

#endif // HAVE_CBLAS

} // namespace cxxblas

#endif // CXXBLAS_LEVEL2_HPR_H
```

7.94 hpr2.h

```
/*
 *   Copyright (c) 2009, Michael Lehn
 *
 *   All rights reserved.
 *
 *   Redistribution and use in source and binary forms, with or without
 *   modification, are permitted provided that the following conditions
 *   are met:
 *
 *   1) Redistributions of source code must retain the above copyright
 *      notice, this list of conditions and the following disclaimer.
 *   2) Redistributions in binary form must reproduce the above copyright
 *      notice, this list of conditions and the following disclaimer in
 *      the documentation and/or other materials provided with the
 *      distribution.
 *   3) Neither the name of the FLENS development group nor the names of
 *      its contributors may be used to endorse or promote products derived
 *      from this software without specific prior written permission.
 *
 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_LEVEL2_HPR2_H
#define CXXBLAS_LEVEL2_HPR2_H 1
```

```
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 namespace cxxblas {
40
41     template<typename IndexType, typename ALPHA, typename VX, typename VY, typename MA>
42     void hpr2(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const VX *x,
43               IndexType incX, const VY *y, IndexType incY, MA *A);
44
45 #ifdef HAVE_CBLAS
46     // cher
47     template<typename IndexType>
48     typename If<IndexType>::isBlasCompatibleInteger
49     hpr2(StorageOrder order, StorageUpLo upLo, IndexType n, float alpha, const ComplexFloat *x,
50          IndexType incX, const ComplexFloat *y, IndexType incY, ComplexFloat *A);
51
52     // zher
53     template<typename IndexType>
54     typename If<IndexType>::isBlasCompatibleInteger
55     hpr2(StorageOrder order, StorageUpLo upLo, IndexType n, double alpha, const ComplexDouble *x,
56          IndexType incX, const ComplexDouble *y, IndexType incY, ComplexDouble *A);
57
58 #endif // HAVE_CBLAS
59
60 } // namespace cxxblas
61
62 #endif // CXXBLAS_LEVEL2_HPR2_H
```

7.95 level2.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2_LEVEL2_H
34 #define CXXBLAS_LEVEL2_LEVEL2_H 1
35
36 #include "cxxblas/level2/gbmv.h"
37 #include "cxxblas/level2/gemv.h"
38 #include "cxxblas/level2/ger.h"
39 #include "cxxblas/level2/hbmv.h"
40 #include "cxxblas/level2/hemv.h"
41 #include "cxxblas/level2/her.h"
42 #include "cxxblas/level2/her2.h"
43 #include "cxxblas/level2/hpmv.h"
44 #include "cxxblas/level2/hpr.h"
45 #include "cxxblas/level2/hpr2.h"
46 #include "cxxblas/level2/sbmv.h"
47 #include "cxxblas/level2/spmv.h"
48 #include "cxxblas/level2/spr.h"
49 #include "cxxblas/level2/spr2.h"
50 #include "cxxblas/level2/symv.h"
51 #include "cxxblas/level2/syr.h"
52 #include "cxxblas/level2/syr2.h"
```

```
53 #include "cxxblas/level2/tbmv.h"
54 #include "cxxblas/level2/tbsv.h"
55 #include "cxxblas/level2/tpmv.h"
56 #include "cxxblas/level2/tpsv.h"
57 #include "cxxblas/level2/trmv.h"
58 #include "cxxblas/level2/trsv.h"
59
60 #endif // CXXBLAS_LEVEL2_LEVEL2_H
```

7.96 sbmv.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2_SBMV_H
34 #define CXXBLAS_LEVEL2_SBMV_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_SBMV 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
44              typename VY>
45     void sbmv(StorageOrder order, StorageUpLo upLo, IndexType n, IndexType k, const ALPHA &alpha,
46               const MA *A, IndexType ldA, const VX *x, IndexType incX, const BETA &beta, VY *y,
47               IndexType incY);
48
49 #ifdef HAVE_CBLAS
50
51     // ssbmv
52     template<typename IndexType>
53     typename If<IndexType>::isBlasCompatibleInteger
54     sbmv(StorageOrder order, StorageUpLo upLo, IndexType n, IndexType k, float alpha,
55          const float *A, IndexType ldA, const float *x, IndexType incX, float beta, float *y,
56          IndexType incY);
57
58     // dsbmv
59     template<typename IndexType>
60     typename If<IndexType>::isBlasCompatibleInteger
61     sbmv(StorageOrder order, StorageUpLo upLo, IndexType n, IndexType k, double alpha,
62          const double *A, IndexType ldA, const double *x, IndexType incX, double &beta, double *y,
63          IndexType incY);
64
65 #endif // HAVE_CBLAS
66
67 } // namespace cxxblas
68
69 #endif // CXXBLAS_LEVEL2_SBMV_H
```

7.97 spmv.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2_SPMV_H
34 #define CXXBLAS_LEVEL2_SPMV_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_SPMV 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
44              typename VY>
45     void spmv(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const MA *A,
46               const VX *x, IndexType incX, const BETA &beta, VY *y, IndexType incY);
47
48 #ifdef HAVE_CBLAS
49
50     // sspmv
51     template<typename IndexType>
52     typename If<IndexType>::isBlasCompatibleInteger
53     spmv(StorageOrder order, StorageUpLo upLo, IndexType n, float alpha, const float *A,
54          const float *x, IndexType incX, float beta, float *y, IndexType incY);
55
56     // dspmv
57     template<typename IndexType>
58     typename If<IndexType>::isBlasCompatibleInteger
59     spmv(StorageOrder order, StorageUpLo upLo, IndexType n, double alpha, const double *A,
60          const double *x, IndexType incX, double beta, double *y, IndexType incY);
61
62 #endif // HAVE_CBLAS
63
64 } // namespace cxxblas
65
66 #endif // CXXBLAS_LEVEL2_SPMV_H
```

7.98 spr.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
```

```
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL2_SPR_H
34 #define CXXBLAS_LEVEL2_SPR_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_SPR 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename VX, typename MA>
44     void spr(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const VX *x,
45              IndexType incX, MA *A);
46
47 #ifdef HAVE_CBLAS
48     // sspr
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger spr(StorageOrder order, StorageUpLo upLo,
51                                                         IndexType n, float alpha, const float *x,
52                                                         IndexType incX, float *A);
53
54     // dspr
55     template<typename IndexType>
56     typename If<IndexType>::isBlasCompatibleInteger spr(StorageOrder order, StorageUpLo upLo,
57                                                         IndexType n, double alpha, const double *x,
58                                                         IndexType incX, double *A);
59
60 #endif // HAVE_CBLAS
61
62 } // namespace cxxblas
63
64 #endif // CXXBLAS_LEVEL2_SPR_H
```

7.99 spr2.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
```

```
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_LEVEL2_SPR2_H
#define CXXBLAS_LEVEL2_SPR2_H 1

#include "cxxblas/drivers/drivers.h"
#include "cxxblas/typedefs.h"

namespace cxxblas {

    template<typename IndexType, typename ALPHA, typename VX, typename VY, typename MA>
    void spr2(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const VX *x,
              IndexType incX, const VY *y, IndexType incY, MA *A);

#ifdef HAVE_CBLAS
    // sspr2
    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    spr2(StorageOrder order, StorageUpLo upLo, IndexType n, float alpha, const float *x,
         IndexType incX, const float *y, IndexType incY, float *A);

    // dspr2
    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    spr2(StorageOrder order, StorageUpLo upLo, IndexType n, double alpha, const double *x,
         IndexType incX, const double *y, IndexType incY, double *A);

#endif // HAVE_CBLAS

} // namespace cxxblas

#endif // CXXBLAS_LEVEL2_SPR2_H
```

7.100 symv.h

```
/*
 *   Copyright (c) 2010, Michael Lehn
 *
 *   All rights reserved.
 *
 *   Redistribution and use in source and binary forms, with or without
 *   modification, are permitted provided that the following conditions
 *   are met:
 *
 *   1) Redistributions of source code must retain the above copyright
 *      notice, this list of conditions and the following disclaimer.
 *   2) Redistributions in binary form must reproduce the above copyright
 *      notice, this list of conditions and the following disclaimer in
 *      the documentation and/or other materials provided with the
 *      distribution.
 *   3) Neither the name of the FLENS development group nor the names of
 *      its contributors may be used to endorse or promote products derived
 *      from this software without specific prior written permission.
 *
 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_LEVEL2_SYMV_H
#define CXXBLAS_LEVEL2_SYMV_H 1

#include "cxxblas/drivers/drivers.h"
#include "cxxblas/typedefs.h"

#define HAVE_CXXBLAS_SYMV 1

namespace cxxblas {

    template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
             typename VY>
    void symv(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const MA *A,
              IndexType ldA, const VX *x, IndexType incX, const BETA &beta, VY *y, IndexType incY);
```

```
47
48 #ifdef HAVE_CBLAS
49
50     // ssymv
51     template<typename IndexType>
52     typename If<IndexType>::isBlasCompatibleInteger
53     symv(StorageOrder order, StorageUpLo upLo, IndexType n, float alpha, const float *A,
54          IndexType ldA, const float *x, IndexType incX, float beta, float *y, IndexType incY);
55
56     // dsymv
57     template<typename IndexType>
58     typename If<IndexType>::isBlasCompatibleInteger
59     symv(StorageOrder order, StorageUpLo upLo, IndexType n, double alpha, const double *A,
60          IndexType ldA, const double *x, IndexType incX, double beta, double *y, IndexType incY);
61
62 // Complex functions symv provided by lapack
63 #   ifdef USE_CXXLAPACK
64
65     // csymv
66     template<typename IndexType>
67     typename If<IndexType>::isBlasCompatibleInteger
68     symv(StorageOrder order, StorageUpLo upLo, IndexType n, const ComplexFloat &alpha,
69          const ComplexFloat *A, IndexType ldA, const ComplexFloat *x, IndexType incX,
70          const ComplexFloat &beta, ComplexFloat *y, IndexType incY);
71
72     // zsymv
73     template<typename IndexType>
74     typename If<IndexType>::isBlasCompatibleInteger
75     symv(StorageOrder order, StorageUpLo upLo, IndexType n, const ComplexDouble &alpha,
76          const ComplexDouble *A, IndexType ldA, const ComplexDouble *x, IndexType incX,
77          const ComplexDouble &beta, ComplexDouble *y, IndexType incY);
78
79 #   endif // USE_CXXLAPACK
80
81 #endif // HAVE_CBLAS
82
83 } // namespace cxxblas
84
85 #endif // CXXBLAS_LEVEL2_SYMV_H
```

7.101 symv.h

```
1 /*
2  *   Copyright (c) 2012, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2EXTENSIONS_SYMV_H
34 #define CXXBLAS_LEVEL2EXTENSIONS_SYMV_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_SYMV 1
40
41 namespace cxxblas {
```

```
42
43 #ifdef HAVE_CBLAS
44
45     template<typename IndexType>
46     typename If<IndexType>::isBlasCompatibleInteger
47     symv(StorageOrder order, StorageUpLo upLo, IndexType n, float alpha, const float *A,
48          IndexType ldA, const float *x, IndexType incX, float beta, std::complex<float> *y,
49          IndexType incY);
50
51     template<typename IndexType>
52     typename If<IndexType>::isBlasCompatibleInteger
53     symv(StorageOrder order, StorageUpLo upLo, IndexType n, float alpha, const float *A,
54          IndexType ldA, const float *x, IndexType incX, std::complex<float> beta,
55          std::complex<float> *y, IndexType incY);
56
57     template<typename IndexType>
58     typename If<IndexType>::isBlasCompatibleInteger
59     symv(StorageOrder order, StorageUpLo upLo, IndexType n, float alpha, const float *A,
60          IndexType ldA, const std::complex<float> *x, IndexType incX, float beta,
61          std::complex<float> *y, IndexType incY);
62
63     template<typename IndexType>
64     typename If<IndexType>::isBlasCompatibleInteger
65     symv(StorageOrder order, StorageUpLo upLo, IndexType n, float alpha, const float *A,
66          IndexType ldA, const std::complex<float> *x, IndexType incX, std::complex<float> beta,
67          std::complex<float> *y, IndexType incY);
68
69     template<typename IndexType>
70     typename If<IndexType>::isBlasCompatibleInteger
71     symv(StorageOrder order, StorageUpLo upLo, IndexType n, double alpha, const double *A,
72          IndexType ldA, const double *x, IndexType incX, double beta, std::complex<double> *y,
73          IndexType incY);
74
75     template<typename IndexType>
76     typename If<IndexType>::isBlasCompatibleInteger
77     symv(StorageOrder order, StorageUpLo upLo, IndexType n, double alpha, const double *A,
78          IndexType ldA, const double *x, IndexType incX, std::complex<double> beta,
79          std::complex<double> *y, IndexType incY);
80
81     template<typename IndexType>
82     typename If<IndexType>::isBlasCompatibleInteger
83     symv(StorageOrder order, StorageUpLo upLo, IndexType n, double alpha, const double *A,
84          IndexType ldA, const std::complex<double> *x, IndexType incX, double beta,
85          std::complex<double> *y, IndexType incY);
86
87     template<typename IndexType>
88     typename If<IndexType>::isBlasCompatibleInteger
89     symv(StorageOrder order, StorageUpLo upLo, IndexType n, double alpha, const double *A,
90          IndexType ldA, const std::complex<double> *x, IndexType incX, std::complex<double> beta,
91          std::complex<double> *y, IndexType incY);
92
93 #endif // HAVE_CBLAS
94
95 } // namespace cxxblas
96
97 #endif // CXXBLAS_LEVEL2EXTENSIONS_SYMV_H
```

7.102 syr.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
```

```
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL2_SYR_H
34 #define CXXBLAS_LEVEL2_SYR_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_SYR 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename VX, typename MA>
44     void syr(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const VX *x,
45              IndexType incX, MA *A, IndexType ldA);
46
47 #ifdef HAVE_CBLAS
48     // ssyr
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger syr(StorageOrder order, StorageUpLo upLo,
51                                                         IndexType n, float alpha, const float *x,
52                                                         IndexType incX, float *A, IndexType ldA);
53
54     // dsyr
55     template<typename IndexType>
56     typename If<IndexType>::isBlasCompatibleInteger syr(StorageOrder order, StorageUpLo upLo,
57                                                         IndexType n, double alpha, const double *x,
58                                                         IndexType incX, double *A, IndexType ldA);
59
60 #endif // HAVE_CBLAS
61
62 } // namespace cxxblas
63
64 #endif // CXXBLAS_LEVEL2_SYR_H
```

7.103 syr2.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL2_SYR2_H
34 #define CXXBLAS_LEVEL2_SYR2_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_SYR2 1
40
```

```
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename VX, typename VY, typename MA>
44     void syr2(StorageOrder order, StorageUpLo upLo, IndexType n, const ALPHA &alpha, const VX *x,
45               IndexType incX, const VY *y, IndexType incY, MA *A, IndexType ldA);
46
47 #ifdef HAVE_CBLAS
48     // ssyr2
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger
51     syr2(StorageOrder order, StorageUpLo upLo, IndexType n, float alpha, const float *x,
52          IndexType incX, const float *y, IndexType incY, float *A, IndexType ldA);
53
54     // dsyr2
55     template<typename IndexType>
56     typename If<IndexType>::isBlasCompatibleInteger
57     syr2(StorageOrder order, StorageUpLo upLo, IndexType n, double alpha, const double *x,
58          IndexType incX, const double *y, IndexType incY, double *A, IndexType ldA);
59
60 #endif // HAVE_CBLAS
61
62 } // namespace cxxblas
63
64 #endif // CXXBLAS_LEVEL2_SYR2_H
```

7.104 tbmv.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2_TBMV_H
34 #define CXXBLAS_LEVEL2_TBMV_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_TBMV 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename MA, typename VX>
44     void tbmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
45               IndexType k, const MA *A, IndexType ldA, VX *x, IndexType incX);
46
47 #ifdef HAVE_CBLAS
48
49     // stbmv
50     template<typename IndexType>
51     typename If<IndexType>::isBlasCompatibleInteger
52     tbmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
53          IndexType k, const float *A, IndexType ldA, float *x, IndexType incX);
54
55     // dtbmv
56     template<typename IndexType>
```

```
57      typename If<IndexType>::isBlasCompatibleInteger
58      tbmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
59           IndexType k, const double *A, IndexType ldA, double *x, IndexType incX);
60
61      // ctbmv
62      template<typename IndexType>
63      typename If<IndexType>::isBlasCompatibleInteger
64      tbmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
65           IndexType k, const ComplexFloat *A, IndexType ldA, ComplexFloat *x, IndexType incX);
66
67      // ztbmv
68      template<typename IndexType>
69      typename If<IndexType>::isBlasCompatibleInteger
70      tbmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
71           IndexType k, const ComplexDouble *A, IndexType ldA, ComplexDouble *x, IndexType incX);
72
73  #endif // HAVE_CBLAS
74
75  } // namespace cxxblas
76
77  #endif // CXXBLAS_LEVEL2_TBMV_H
```

7.105 tbsv.h

```
1  /*
2   *   Copyright (c) 2009, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_LEVEL2_TBSV_H
34  #define CXXBLAS_LEVEL2_TBSV_H 1
35
36  #include "cxxblas/drivers/drivers.h"
37  #include "cxxblas/typedefs.h"
38
39  #define HAVE_CXXBLAS_TBSV 1
40
41  namespace cxxblas {
42
43      template<typename IndexType, typename MA, typename VX>
44      void tbsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
45                IndexType k, const MA *A, IndexType ldA, VX *x, IndexType incX);
46
47  #ifdef HAVE_CBLAS
48
49      // stbsv
50      template<typename IndexType>
51      typename If<IndexType>::isBlasCompatibleInteger
52      tbsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
53           IndexType k, const float *A, IndexType ldA, float *x, IndexType incX);
54
55      // dtbsv
56      template<typename IndexType>
57      typename If<IndexType>::isBlasCompatibleInteger
58      tbsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
59           IndexType k, const double *A, IndexType ldA, double *x, IndexType incX);
```

```
60
61      // ctbsv
62      template<typename IndexType>
63      typename If<IndexType>::isBlasCompatibleInteger
64      tbsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
65           IndexType k, const ComplexFloat *A, IndexType ldA, ComplexFloat *x, IndexType incX);
66
67      // ztbsv
68      template<typename IndexType>
69      typename If<IndexType>::isBlasCompatibleInteger
70      tbsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
71           IndexType k, const ComplexDouble *A, IndexType ldA, ComplexDouble *x, IndexType incX);
72
73  #endif // HAVE_CBLAS
74
75  } // namespace cxxblas
76
77  #endif // CXXBLAS_LEVEL2_TBSV_H
```

7.106 tpmv.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_LEVEL2_TPMV_H
34  #define CXXBLAS_LEVEL2_TPMV_H 1
35
36  #include "cxxblas/drivers/drivers.h"
37  #include "cxxblas/typedefs.h"
38
39  #define HAVE_CXXBLAS_TPMV 1
40
41  namespace cxxblas {
42
43      template<typename IndexType, typename MA, typename VX>
44      void tpmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
45                const MA *A, VX *x, IndexType incX);
46
47  #ifdef HAVE_CBLAS
48
49      // stpmv
50      template<typename IndexType>
51      typename If<IndexType>::isBlasCompatibleInteger tpmv(StorageOrder order, StorageUpLo upLo,
52                                                           Transpose transA, Diag diag, IndexType n,
53                                                           const float *A, float *x, IndexType incX);
54
55      // dtpmv
56      template<typename IndexType>
57      typename If<IndexType>::isBlasCompatibleInteger
58      tpmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
59           const double *A, double *x, IndexType incX);
60
61      // ctpmv
62      template<typename IndexType>
```

```
63      typename If<IndexType>::isBlasCompatibleInteger
64      tpmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
65           const ComplexFloat *A, ComplexFloat *x, IndexType incX);
66
67      // ztpmv
68      template<typename IndexType>
69      typename If<IndexType>::isBlasCompatibleInteger
70      tpmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
71           const ComplexDouble *A, ComplexDouble *x, IndexType incX);
72
73  #endif // HAVE_CBLAS
74
75  } // namespace cxxblas
76
77  #endif // CXXBLAS_LEVEL2_TPMV_H
```

7.107 tpsv.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_LEVEL2_TPSV_H
34  #define CXXBLAS_LEVEL2_TPSV_H 1
35
36  #include "cxxblas/drivers/drivers.h"
37  #include "cxxblas/typedefs.h"
38
39  #define HAVE_CXXBLAS_TPSV 1
40
41  namespace cxxblas {
42
43      template<typename IndexType, typename MA, typename VX>
44      void tpsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
45                const MA *A, VX *x, IndexType incX);
46
47  #ifdef HAVE_CBLAS
48
49      // stpsv
50      template<typename IndexType>
51      typename If<IndexType>::isBlasCompatibleInteger tpsv(StorageOrder order, StorageUpLo upLo,
52                                                           Transpose transA, Diag diag, IndexType n,
53                                                           const float *A, float *x, IndexType incX);
54
55      // dtpsv
56      template<typename IndexType>
57      typename If<IndexType>::isBlasCompatibleInteger
58      tpsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
59           const double *A, double *x, IndexType incX);
60
61      // ctpsv
62      template<typename IndexType>
63      typename If<IndexType>::isBlasCompatibleInteger
64      tpsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
65           const ComplexFloat *A, ComplexFloat *x, IndexType incX);
```

```
66
67      // ztpsv
68      template<typename IndexType>
69      typename If<IndexType>::isBlasCompatibleInteger
70      tpsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
71           const ComplexDouble *A, ComplexDouble *x, IndexType incX);
72
73  #endif // HAVE_CBLAS
74
75  } // namespace cxxblas
76
77  #endif // CXXBLAS_LEVEL2_TPSV_H
```

7.108 trmv.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_LEVEL2_TRMV_H
34  #define CXXBLAS_LEVEL2_TRMV_H 1
35
36  #include "cxxblas/drivers/drivers.h"
37  #include "cxxblas/typedefs.h"
38
39  #define HAVE_CXXBLAS_TRMV 1
40
41  namespace cxxblas {
42
43      template<typename IndexType, typename MA, typename VX>
44      void trmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
45                const MA *A, IndexType ldA, VX *x, IndexType incX);
46
47  #ifdef HAVE_CBLAS
48
49      // strmv
50      template<typename IndexType>
51      typename If<IndexType>::isBlasCompatibleInteger
52      trmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
53           const float *A, IndexType ldA, float *x, IndexType incX);
54
55      // dtrmv
56      template<typename IndexType>
57      typename If<IndexType>::isBlasCompatibleInteger
58      trmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
59           const double *A, IndexType ldA, double *x, IndexType incX);
60
61      // ctrmv
62      template<typename IndexType>
63      typename If<IndexType>::isBlasCompatibleInteger
64      trmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
65           const ComplexFloat *A, IndexType ldA, ComplexFloat *x, IndexType incX);
66
67      // ztrmv
68      template<typename IndexType>
```

```
69     typename If<IndexType>::isBlasCompatibleInteger
70     trmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
71          const ComplexDouble *A, IndexType ldA, ComplexDouble *x, IndexType incX);
72
73 #endif // HAVE_CBLAS
74
75 } // namespace cxxblas
76
77 #endif // CXXBLAS_LEVEL2_TRMV_H
```

7.109 trmv.h

```
1 /*
2  *   Copyright (c) 2012, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2EXTENSIONS_TRMV_H
34 #define CXXBLAS_LEVEL2EXTENSIONS_TRMV_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_TRMV 1
40
41 namespace cxxblas {
42
43 #ifdef HAVE_CBLAS
44
45     template<typename IndexType>
46     typename If<IndexType>::isBlasCompatibleInteger
47     trmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
48          const float *A, IndexType ldA, ComplexFloat *x, IndexType incX);
49
50     template<typename IndexType>
51     typename If<IndexType>::isBlasCompatibleInteger
52     trmv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
53          const double *A, IndexType ldA, ComplexDouble *x, IndexType incX);
54
55 #endif
56
57 } // namespace cxxblas
58
59 #endif // CXXBLAS_LEVEL2EXTENSIONS_TRMV_H
```

7.110 trsv.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
```

```
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL2_TRSV_H
34 #define CXXBLAS_LEVEL2_TRSV_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_TRSV 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename MA, typename VX>
44     void trsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
45               const MA *A, IndexType ldA, VX *x, IndexType incX);
46
47 #ifdef HAVE_CBLAS
48
49     // strsv
50     template<typename IndexType>
51     typename If<IndexType>::isBlasCompatibleInteger
52     trsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
53          const float *A, IndexType ldA, float *x, IndexType incX);
54
55     // dtrsv
56     template<typename IndexType>
57     typename If<IndexType>::isBlasCompatibleInteger
58     trsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
59          const double *A, IndexType ldA, double *x, IndexType incX);
60
61     // ctrsv
62     template<typename IndexType>
63     typename If<IndexType>::isBlasCompatibleInteger
64     trsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
65          const ComplexFloat *A, IndexType ldA, ComplexFloat *x, IndexType incX);
66
67     // ztrsv
68     template<typename IndexType>
69     typename If<IndexType>::isBlasCompatibleInteger
70     trsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
71          const ComplexDouble *A, IndexType ldA, ComplexDouble *x, IndexType incX);
72
73 #endif // HAVE_CBLAS
74
75 } // namespace cxxblas
76
77 #endif // CXXBLAS_LEVEL2_TRSV_H
```

7.111 trsv.h

```
1 /*
2  *   Copyright (c) 2012, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
```

```
 *
 *   1) Redistributions of source code must retain the above copyright
 *      notice, this list of conditions and the following disclaimer.
 *   2) Redistributions in binary form must reproduce the above copyright
 *      notice, this list of conditions and the following disclaimer in
 *      the documentation and/or other materials provided with the
 *      distribution.
 *   3) Neither the name of the FLENS development group nor the names of
 *      its contributors may be used to endorse or promote products derived
 *      from this software without specific prior written permission.
 *
 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_LEVEL2EXTENSIONS_TRSV_H
#define CXXBLAS_LEVEL2EXTENSIONS_TRSV_H 1

#include "cxxblas/drivers/drivers.h"
#include "cxxblas/typedefs.h"

#define HAVE_CXXBLAS_TRSV 1

namespace cxxblas {

#ifdef HAVE_CBLAS

    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    trsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
         const float *A, IndexType ldA, ComplexFloat *x, IndexType incX);

    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    trsv(StorageOrder order, StorageUpLo upLo, Transpose transA, Diag diag, IndexType n,
         const double *A, IndexType ldA, ComplexDouble *x, IndexType incX);

#endif

} // namespace cxxblas

#endif // CXXBLAS_LEVEL2EXTENSIONS_TRSV_H
```

7.112 level2extensions.h

```
/*
 *   Copyright (c) 2010, Michael Lehn
 *
 *   All rights reserved.
 *
 *   Redistribution and use in source and binary forms, with or without
 *   modification, are permitted provided that the following conditions
 *   are met:
 *
 *   1) Redistributions of source code must retain the above copyright
 *      notice, this list of conditions and the following disclaimer.
 *   2) Redistributions in binary form must reproduce the above copyright
 *      notice, this list of conditions and the following disclaimer in
 *      the documentation and/or other materials provided with the
 *      distribution.
 *   3) Neither the name of the FLENS development group nor the names of
 *      its contributors may be used to endorse or promote products derived
 *      from this software without specific prior written permission.
 *
 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
```

```
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_LEVEL2EXTENSIONS_LEVEL2EXTENSIONS_H
#define CXXBLAS_LEVEL2EXTENSIONS_LEVEL2EXTENSIONS_H 1

#include "cxxblas/level2extensions/gbmv.h"
#include "cxxblas/level2extensions/gemv.h"
#include "cxxblas/level2extensions/hemv.h"
#include "cxxblas/level2extensions/her.h"
#include "cxxblas/level2extensions/her2.h"
#include "cxxblas/level2extensions/symv.h"
#include "cxxblas/level2extensions/trmv.h"
#include "cxxblas/level2extensions/trsv.h"

#endif // CXXBLAS_LEVEL2EXTENSIONS_LEVEL2EXTENSIONS_H
```

7.113 gemm.h

```
/*
 *   Copyright (c) 2010, Michael Lehn
 *
 *   All rights reserved.
 *
 *   Redistribution and use in source and binary forms, with or without
 *   modification, are permitted provided that the following conditions
 *   are met:
 *
 *   1) Redistributions of source code must retain the above copyright
 *      notice, this list of conditions and the following disclaimer.
 *   2) Redistributions in binary form must reproduce the above copyright
 *      notice, this list of conditions and the following disclaimer in
 *      the documentation and/or other materials provided with the
 *      distribution.
 *   3) Neither the name of the FLENS development group nor the names of
 *      its contributors may be used to endorse or promote products derived
 *      from this software without specific prior written permission.
 *
 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_LEVEL3_GEMM_H
#define CXXBLAS_LEVEL3_GEMM_H 1

#include "cxxblas/drivers/drivers.h"
#include "cxxblas/typedefs.h"

#define HAVE_CXXBLAS_GEMM 1

namespace cxxblas {

    template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
             typename MC>
    void gemm(StorageOrder order, Transpose transA, Transpose transB, IndexType m, IndexType n,
              IndexType k, const ALPHA &alpha, const MA *A, IndexType ldA, const MB *B,
              IndexType ldB, const BETA &beta, MC *C, IndexType ldC);

#ifdef HAVE_CBLAS

    // sgemm
    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    gemm(StorageOrder order, Transpose transA, Transpose transB, IndexType m, IndexType n,
         IndexType k, float alpha, const float *A, IndexType ldA, const float *B, IndexType ldB,
         float beta, float *C, IndexType ldC);

    // dgemm
    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    gemm(StorageOrder order, Transpose transA, Transpose transB, IndexType m, IndexType n,
         IndexType k, double alpha, const double *A, IndexType ldA, const double *B, IndexType ldB,
         double beta, double *C, IndexType ldC);

```

```
65      // cgemm
66      template<typename IndexType>
67      typename If<IndexType>::isBlasCompatibleInteger
68      gemm(StorageOrder order, Transpose transA, Transpose transB, IndexType m, IndexType n,
69           IndexType k, const ComplexFloat &alpha, const ComplexFloat *A, IndexType ldA,
70           const ComplexFloat *B, IndexType ldB, const ComplexFloat &beta, ComplexFloat *C,
71           IndexType ldC);
72
73      // zgemm
74      template<typename IndexType>
75      typename If<IndexType>::isBlasCompatibleInteger
76      gemm(StorageOrder order, Transpose transA, Transpose transB, IndexType m, IndexType n,
77           IndexType k, const ComplexDouble &alpha, const ComplexDouble *A, IndexType ldA,
78           const ComplexDouble *B, IndexType ldB, const ComplexDouble &beta, ComplexDouble *C,
79           IndexType ldC);
80
81  #endif // HAVE_CBLAS
82
83  } // namespace cxxblas
84
85  #endif // CXXBLAS_LEVEL3_GEMM_H
```

7.114 hemm.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_LEVEL3_HEMM_H
34  #define CXXBLAS_LEVEL3_HEMM_H 1
35
36  #include "cxxblas/drivers/drivers.h"
37  #include "cxxblas/typedefs.h"
38
39  #define HAVE_CXXBLAS_HEMM 1
40
41  namespace cxxblas {
42
43      template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
44               typename MC>
45      void hemm(StorageOrder order, Side side, StorageUpLo upLo, IndexType m, IndexType n,
46                const ALPHA &alpha, const MA *A, IndexType ldA, const MB *B, IndexType ldB,
47                const BETA &beta, MC *C, IndexType ldC);
48
49  #ifdef HAVE_CBLAS
50
51      template<typename IndexType>
52      typename If<IndexType>::isBlasCompatibleInteger
53      hemm(StorageOrder order, Side side, StorageUpLo upLo, IndexType m, IndexType n,
54           const ComplexFloat &alpha, const ComplexFloat *A, IndexType ldA, const ComplexFloat *B,
55           IndexType ldB, const ComplexFloat &beta, ComplexFloat *C, IndexType ldC);
56
57      template<typename IndexType>
58      typename If<IndexType>::isBlasCompatibleInteger
59      hemm(StorageOrder order, Side side, StorageUpLo upLo, IndexType m, IndexType n,
```

```
60          const ComplexDouble &alpha, const ComplexDouble *A, IndexType ldA, const ComplexDouble *B,
61          IndexType ldB, const ComplexDouble &beta, ComplexDouble *C, IndexType ldC);
62
63 #endif // HAVE_CBLAS
64
65 } // namespace cxxblas
66
67 #endif // CXXBLAS_LEVEL3_HEMM_H
```

7.115 her2k.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL3_HER2K_H
34 #define CXXBLAS_LEVEL3_HER2K_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_HER2K 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
44              typename MC>
45     void her2k(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
46                const ALPHA &alpha, const MA *A, IndexType ldA, const MB *B, IndexType ldB,
47                const BETA &beta, MC *C, IndexType ldC);
48
49 #ifdef HAVE_CBLAS
50
51     // cher2k
52     template<typename IndexType>
53     typename If<IndexType>::isBlasCompatibleInteger
54     her2k(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
55           const ComplexFloat &alpha, const ComplexFloat *A, IndexType ldA, const ComplexFloat *B,
56           IndexType ldB, float beta, ComplexFloat *C, IndexType ldC);
57
58     // zher2k
59     template<typename IndexType>
60     typename If<IndexType>::isBlasCompatibleInteger
61     her2k(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
62           const ComplexDouble &alpha, const ComplexDouble *A, IndexType ldA, const ComplexDouble *B,
63           IndexType ldB, double beta, ComplexDouble *C, IndexType ldC);
64
65 #endif // HAVE_CBLAS
66
67 } // namespace cxxblas
68
69 #endif // CXXBLAS_LEVEL3_HER2K_H
```

7.116 herk.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL3_HERK_H
34 #define CXXBLAS_LEVEL3_HERK_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_HERK 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename BETA, typename MC>
44     void herk(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
45               const ALPHA &alpha, const MA *A, IndexType ldA, const BETA &beta, MC *C,
46               IndexType ldC);
47
48 #ifdef HAVE_CBLAS
49
50     // cherk
51     template<typename IndexType>
52     typename If<IndexType>::isBlasCompatibleInteger
53     herk(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
54          float alpha, const ComplexFloat *A, IndexType ldA, float beta, ComplexFloat *C,
55          IndexType ldC);
56
57     // zherk
58     template<typename IndexType>
59     typename If<IndexType>::isBlasCompatibleInteger
60     herk(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
61          double alpha, const ComplexDouble *A, IndexType ldA, double beta, ComplexDouble *C,
62          IndexType ldC);
63
64 #endif // HAVE_CBLAS
65
66 } // namespace cxxblas
67
68 #endif // CXXBLAS_LEVEL3_HERK_H
```

7.117 level3.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
```

```
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL3_LEVEL3_H
34 #define CXXBLAS_LEVEL3_LEVEL3_H 1
35
36 #include "cxxblas/level3/gemm.h"
37 #include "cxxblas/level3/hemm.h"
38 #include "cxxblas/level3/herk.h"
39 #include "cxxblas/level3/her2k.h"
40 #include "cxxblas/level3/symm.h"
41 #include "cxxblas/level3/syrk.h"
42 #include "cxxblas/level3/syr2k.h"
43 #include "cxxblas/level3/trmm.h"
44 #include "cxxblas/level3/trsm.h"
45
46 #endif // CXXBLAS_LEVEL3_LEVEL3_H
```

7.118 symm.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL3_SYMM_H
34 #define CXXBLAS_LEVEL3_SYMM_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_SYMM 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
44              typename MC>
```

```
45      void symm(StorageOrder order, Side side, StorageUpLo upLo, IndexType m, IndexType n,
46                const ALPHA &alpha, const MA *A, IndexType ldA, const MB *B, IndexType ldB,
47                const BETA &beta, MC *C, IndexType ldC);
48
49  #ifdef HAVE_CBLAS
50
51      // ssymm
52      template<typename IndexType>
53      typename If<IndexType>::isBlasCompatibleInteger
54      symm(StorageOrder order, Side side, StorageUpLo upLo, IndexType m, IndexType n, float alpha,
55           const float *A, IndexType ldA, const float *B, IndexType ldB, float beta, float *C,
56           IndexType ldC);
57
58      // dsymm
59      template<typename IndexType>
60      typename If<IndexType>::isBlasCompatibleInteger
61      symm(StorageOrder order, Side side, StorageUpLo upLo, IndexType m, IndexType n, double alpha,
62           const double *A, IndexType ldA, const double *B, IndexType ldB, double beta, double *C,
63           IndexType ldC);
64
65      // csymm
66      template<typename IndexType>
67      typename If<IndexType>::isBlasCompatibleInteger
68      symm(StorageOrder order, Side side, StorageUpLo upLo, IndexType m, IndexType n,
69           const ComplexFloat &alpha, const ComplexFloat *A, IndexType ldA, const ComplexFloat *B,
70           IndexType ldB, const ComplexFloat &beta, ComplexFloat *C, IndexType ldC);
71
72      // zsymm
73      template<typename IndexType>
74      typename If<IndexType>::isBlasCompatibleInteger
75      symm(StorageOrder order, Side side, StorageUpLo upLo, IndexType m, IndexType n,
76           const ComplexDouble &alpha, const ComplexDouble *A, IndexType ldA, const ComplexDouble *B,
77           IndexType ldB, const ComplexDouble &beta, ComplexDouble *C, IndexType ldC);
78
79  #endif // HAVE_CBLAS
80
81  } // namespace cxxblas
82
83  #endif // CXXBLAS_LEVEL3_SYMM_H
```

7.119 syr2k.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_LEVEL3_SYR2K_H
34  #define CXXBLAS_LEVEL3_SYR2K_H 1
35
36  #include "cxxblas/drivers/drivers.h"
37  #include "cxxblas/typedefs.h"
38
39  #define HAVE_CXXBLAS_SYR2K 1
40
41  namespace cxxblas {
```

```
42
43      template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
44               typename MC>
45      void syr2k(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
46                 const ALPHA &alpha, const MA *A, IndexType ldA, const MB *B, IndexType ldB,
47                 const BETA &beta, MC *C, IndexType ldC);
48
49  #ifdef HAVE_CBLAS
50
51      // ssyr2k
52      template<typename IndexType>
53      typename If<IndexType>::isBlasCompatibleInteger
54      syr2k(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
55            float alpha, const float *A, IndexType ldA, const float *B, IndexType ldB, float beta,
56            float *C, IndexType ldC);
57
58      // dsyr2k
59      template<typename IndexType>
60      typename If<IndexType>::isBlasCompatibleInteger
61      syr2k(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
62            double alpha, const double *A, IndexType ldA, const double *B, IndexType ldB, double beta,
63            double *C, IndexType ldC);
64
65      // csyr2k
66      template<typename IndexType>
67      typename If<IndexType>::isBlasCompatibleInteger
68      syr2k(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
69            const ComplexFloat &alpha, const ComplexFloat *A, IndexType ldA, const ComplexFloat *B,
70            IndexType ldB, const ComplexFloat &beta, ComplexFloat *C, IndexType ldC);
71
72      // zsyr2k
73      template<typename IndexType>
74      typename If<IndexType>::isBlasCompatibleInteger
75      syr2k(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
76            const ComplexDouble &alpha, const ComplexDouble *A, IndexType ldA, const ComplexDouble *B,
77            IndexType ldB, const ComplexDouble &beta, ComplexDouble *C, IndexType ldC);
78
79  #endif // HAVE_CBLAS
80
81  } // namespace cxxblas
82
83  #endif // CXXBLAS_LEVEL3_SYR2K_H
```

7.120 syrk.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_LEVEL3_SYRK_H
34  #define CXXBLAS_LEVEL3_SYRK_H 1
35
36  #include "cxxblas/drivers/drivers.h"
37  #include "cxxblas/typedefs.h"
38
```

```
39 #define HAVE_CXXBLAS_SYRK 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename BETA, typename MC>
44     void syrk(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
45               const ALPHA &alpha, const MA *A, IndexType ldA, const BETA &beta, MC *C,
46               IndexType ldC);
47
48 #ifdef HAVE_CBLAS
49
50     // ssyrk
51     template<typename IndexType>
52     typename If<IndexType>::isBlasCompatibleInteger
53     syrk(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
54          float alpha, const float *A, IndexType ldA, float beta, float *C, IndexType ldC);
55
56     // dsyrk
57     template<typename IndexType>
58     typename If<IndexType>::isBlasCompatibleInteger
59     syrk(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
60          double alpha, const double *A, IndexType ldA, double beta, double *C, IndexType ldC);
61
62     // csyrk
63     template<typename IndexType>
64     typename If<IndexType>::isBlasCompatibleInteger
65     syrk(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
66          const ComplexFloat &alpha, const ComplexFloat *A, IndexType ldA, const ComplexFloat &beta,
67          ComplexFloat *C, IndexType ldC);
68
69     // zsyrk
70     template<typename IndexType>
71     typename If<IndexType>::isBlasCompatibleInteger
72     syrk(StorageOrder order, StorageUpLo upLo, Transpose trans, IndexType n, IndexType k,
73          const ComplexDouble &alpha, const ComplexDouble *A, IndexType ldA,
74          const ComplexDouble &beta, ComplexDouble *C, IndexType ldC);
75
76 #endif // HAVE_CBLAS
77
78 } // namespace cxxblas
79
80 #endif // CXXBLAS_LEVEL3_SYRK_H
```

7.121 trmm.h

```
1  /*
2   *   Copyright (c) 2010, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL3_TRMM_H
34 #define CXXBLAS_LEVEL3_TRMM_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
```

```
39 #define HAVE_CXXBLAS_TRMM 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename MB>
44     void trmm(StorageOrder order, Side side, StorageUpLo upLo, Transpose transA, Diag diag,
45               IndexType m, IndexType n, const ALPHA &alpha, const MA *A, IndexType ldA, MB *B,
46               IndexType ldB);
47
48 #ifdef HAVE_CBLAS
49
50     // strmm
51     template<typename IndexType>
52     typename If<IndexType>::isBlasCompatibleInteger
53     trmm(StorageOrder order, Side side, StorageUpLo upLo, Transpose transA, Diag diag, IndexType m,
54          IndexType n, float alpha, const float *A, IndexType ldA, float *B, IndexType ldB);
55
56     // dtrmm
57     template<typename IndexType>
58     typename If<IndexType>::isBlasCompatibleInteger
59     trmm(StorageOrder order, Side side, StorageUpLo upLo, Transpose transA, Diag diag, IndexType m,
60          IndexType n, double alpha, const double *A, IndexType ldA, double *B, IndexType ldB);
61
62     // ctrmm
63     template<typename IndexType>
64     typename If<IndexType>::isBlasCompatibleInteger
65     trmm(StorageOrder order, Side side, StorageUpLo upLo, Transpose transA, Diag diag, IndexType m,
66          IndexType n, const ComplexFloat &alpha, const ComplexFloat *A, IndexType ldA,
67          ComplexFloat *B, IndexType ldB);
68
69     // ztrmm
70     template<typename IndexType>
71     typename If<IndexType>::isBlasCompatibleInteger
72     trmm(StorageOrder order, Side side, StorageUpLo upLo, Transpose transA, Diag diag, IndexType m,
73          IndexType n, const ComplexDouble &alpha, const ComplexDouble *A, IndexType ldA,
74          ComplexDouble *B, IndexType ldB);
75
76 #endif // HAVE_CBLAS
77
78 } // namespace cxxblas
79
80 #endif // CXXBLAS_LEVEL3_TRMM_H
```

7.122 trsm.h

```
1 /*
2  *   Copyright (c) 2010, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL3_TRSM_H
34 #define CXXBLAS_LEVEL3_TRSM_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
```

```
39 #define HAVE_CXXBLAS_TRSM 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename MB>
44     void trsm(StorageOrder order, Side side, StorageUpLo upLo, Transpose transA, Diag diag,
45               IndexType m, IndexType n, const ALPHA &alpha, const MA *A, IndexType ldA, MB *B,
46               IndexType ldB);
47
48 #ifdef HAVE_CBLAS
49
50     // strsm
51     template<typename IndexType>
52     typename If<IndexType>::isBlasCompatibleInteger
53     trsm(StorageOrder order, Side side, StorageUpLo upLo, Transpose transA, Diag diag, IndexType m,
54          IndexType n, float alpha, const float *A, IndexType ldA, float *B, IndexType ldB);
55
56     // dtrsm
57     template<typename IndexType>
58     typename If<IndexType>::isBlasCompatibleInteger
59     trsm(StorageOrder order, Side side, StorageUpLo upLo, Transpose transA, Diag diag, IndexType m,
60          IndexType n, double alpha, const double *A, IndexType ldA, double *B, IndexType ldB);
61
62     // ctrsm
63     template<typename IndexType>
64     typename If<IndexType>::isBlasCompatibleInteger
65     trsm(StorageOrder order, Side side, StorageUpLo upLo, Transpose transA, Diag diag, IndexType m,
66          IndexType n, const ComplexFloat &alpha, const ComplexFloat *A, IndexType ldA,
67          ComplexFloat *B, IndexType ldB);
68
69     // ztrsm
70     template<typename IndexType>
71     typename If<IndexType>::isBlasCompatibleInteger
72     trsm(StorageOrder order, Side side, StorageUpLo upLo, Transpose transA, Diag diag, IndexType m,
73          IndexType n, const ComplexDouble &alpha, const ComplexDouble *A, IndexType ldA,
74          ComplexDouble *B, IndexType ldB);
75
76 #endif // HAVE_CBLAS
77
78 } // namespace cxxblas
79
80 #endif // CXXBLAS_LEVEL3_TRSM_H
```

7.123 gbmm.h

```
1 /*
2  *   Copyright (c) 2012, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL3EXTENSION_GBMM_H
34 #define CXXBLAS_LEVEL3EXTENSION_GBMM_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
```

```
39 #define HAVE_CXXBLAS_GBMM 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
44              typename VC>
45     void gbmm(StorageOrder order, Side side, Transpose transA, Transpose transB, IndexType m,
46               IndexType n, IndexType kl, IndexType ku, IndexType l, const ALPHA &alpha, const MA *A,
47               IndexType ldA, const MB *B, IndexType ldB, const BETA &beta, VC *C, IndexType ldC);
48
49 } // namespace cxxblas
50
51 #endif // CXXBLAS_LEVEL3EXTENSION_GBMM_H
```

7.124 hbmm.h

```
1 /*
2  *   Copyright (c) 2012, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL3EXTENSION_HBMM_H
34 #define CXXBLAS_LEVEL3EXTENSION_HBMM_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_HBMM 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
44              typename VC>
45     void hbmm(StorageOrder order, Side side, StorageUpLo upLo, IndexType m, IndexType k,
46               IndexType l, const ALPHA &alpha, const MA *A, IndexType ldA, const MB *B,
47               IndexType ldB, const BETA &beta, VC *C, IndexType ldC);
48
49 } // namespace cxxblas
50
51 #endif // CXXBLAS_LEVEL3EXTENSION_HBMM_H
```

7.125 level3extensions.h

```
1 /*
2  *   Copyright (c) 2012, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
```

```
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL3EXTENSIONS_LEVEL3EXTENSIONS_H
34 #define CXXBLAS_LEVEL3EXTENSIONS_LEVEL3EXTENSIONS_H 1
35
36 #include "cxxblas/level3extensions/hbmm.h"
37 #include "cxxblas/level3extensions/gbmm.h"
38 #include "cxxblas/level3extensions/sbmm.h"
39 #include "cxxblas/level3extensions/tbmm.h"
40
41 #endif // CXXBLAS_LEVEL3EXTENSIONS_LEVEL3EXTENSIONS_H
```

7.126 sbmm.h

```
1 /*
2  *   Copyright (c) 2012, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_LEVEL3EXTENSION_SBMM_H
34 #define CXXBLAS_LEVEL3EXTENSION_SBMM_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_SBMM 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
44              typename VC>
45     void sbmm(StorageOrder order, Side side, StorageUpLo upLo, IndexType m, IndexType k,
46               IndexType l, const ALPHA &alpha, const MA *A, IndexType ldA, const MB *B,
47               IndexType ldB, const BETA &beta, VC *C, IndexType ldC);
48
```

```
49 } // namespace cxxblas
50
51 #endif // CXXBLAS_LEVEL3EXTENSION_SBMM_H
```

7.127 tbmm.h

```
1 /*
2  *   Copyright (c) 2012, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_LEVEL3EXTENSION_TBMM_H
34 #define CXXBLAS_LEVEL3EXTENSION_TBMM_H 1
35
36 #include "cxxblas/drivers/drivers.h"
37 #include "cxxblas/typedefs.h"
38
39 #define HAVE_CXXBLAS_TBMM 1
40
41 namespace cxxblas {
42
43     template<typename IndexType, typename ALPHA, typename MA, typename MB>
44     void tbmm(StorageOrder order, Side side, StorageUpLo upLo, Transpose transA, Diag diag,
45               IndexType m, IndexType n, IndexType k, const ALPHA &alpha, const MA *A, IndexType ldA,
46               MB *B, IndexType ldB);
47
48 } // namespace cxxblas
49
50 #endif // CXXBLAS_LEVEL3EXTENSION_TBMM_H
```

7.128 gecrsmv.h

```
1 /*
2  *   Copyright (c) 2007-2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
```

```
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_SPARSELEVEL2_GECRSMV_H
34 #define CXXBLAS_SPARSELEVEL2_GECRSMV_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GECRSMV 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
43              typename VY>
44     void gecrsmv(Transpose trans, IndexType m, IndexType n, const ALPHA &alpha, const MA *A,
45                  const IndexType *ia, const IndexType *ja, const VX *x, const BETA &beta, VY *y);
46
47 #ifdef HAVE_SPARSEBLAS
48
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger
51     gecrsmv(Transpose trans, IndexType m, IndexType n, const float &alpha, const float *A,
52             const IndexType *ia, const IndexType *ja, const float *x, const float &beta, float *y);
53
54     template<typename IndexType>
55     typename If<IndexType>::isBlasCompatibleInteger
56     gecrsmv(Transpose trans, IndexType m, IndexType n, const double &alpha, const double *A,
57             const IndexType *ia, const IndexType *ja, const double *x, const double &beta,
58             double *y);
59
60     template<typename IndexType>
61     typename If<IndexType>::isBlasCompatibleInteger
62     gecrsmv(Transpose trans, IndexType m, IndexType n, const ComplexFloat &alpha,
63             const ComplexFloat *A, const IndexType *ia, const IndexType *ja, const ComplexFloat *x,
64             const ComplexFloat &beta, ComplexFloat *y);
65
66     template<typename IndexType>
67     typename If<IndexType>::isBlasCompatibleInteger
68     gecrsmv(Transpose trans, IndexType m, IndexType n, const ComplexDouble &alpha,
69             const ComplexDouble *A, const IndexType *ia, const IndexType *ja,
70             const ComplexDouble *x, const ComplexDouble &beta, ComplexDouble *y);
71
72 #endif
73
74 } // namespace cxxblas
75
76 #endif // CXXBLAS_SPARSELEVEL2_GECRSMV_H
```

7.129 heccsmv.h

```
1  /*
2   *   Copyright (c) 2007-2012, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
```

```
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_SPARSELEVEL2_HECCSMV_H
34 #define CXXBLAS_SPARSELEVEL2_HECCSMV_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_HECCSMV 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
43              typename VY>
44     void heccsmv(StorageUpLo upLo, IndexType n, const ALPHA &alpha, const MA *A,
45                  const IndexType *ia, const IndexType *ja, const VX *x, const BETA &beta, VY *y);
46
47 #ifdef HAVE_SPARSEBLAS
48
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger
51     heccsmv(StorageUpLo upLo, IndexType n, const ComplexFloat &alpha, const ComplexFloat *A,
52             const IndexType *ia, const IndexType *ja, const ComplexFloat *x,
53             const ComplexFloat &beta, ComplexFloat *y);
54
55     template<typename IndexType>
56     typename If<IndexType>::isBlasCompatibleInteger
57     heccsmv(StorageUpLo upLo, IndexType n, const ComplexDouble &alpha, const ComplexDouble *A,
58             const IndexType *ia, const IndexType *ja, const ComplexDouble *x,
59             const ComplexDouble &beta, ComplexDouble *y);
60
61 #endif // HAVE_SPARSEBLAS
62
63 } // namespace cxxblas
64
65 #endif // CXXBLAS_SPARSELEVEL2_HECCSMV_H
```

7.130 hecrsmv.h

```
1  /*
2   *   Copyright (c) 2007-2012, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_SPARSELEVEL2_HECRSMV_H
34 #define CXXBLAS_SPARSELEVEL2_HECRSMV_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_HECRSMV 1
39
```

```
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
43              typename VY>
44     void hecrsmv(StorageUpLo upLo, IndexType n, const ALPHA &alpha, const MA *A,
45                  const IndexType *ia, const IndexType *ja, const VX *x, const BETA &beta, VY *y);
46
47 #ifdef HAVE_SPARSEBLAS
48
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger
51     hecrsmv(StorageUpLo upLo, IndexType n, const ComplexFloat &alpha, const ComplexFloat *A,
52             const IndexType *ia, const IndexType *ja, const ComplexFloat *x,
53             const ComplexFloat &beta, ComplexFloat *y);
54
55     template<typename IndexType>
56     typename If<IndexType>::isBlasCompatibleInteger
57     hecrsmv(StorageUpLo upLo, IndexType n, const ComplexDouble &alpha, const ComplexDouble *A,
58             const IndexType *ia, const IndexType *ja, const ComplexDouble *x,
59             const ComplexDouble &beta, ComplexDouble *y);
60
61 #endif // HAVE_SPARSEBLAS
62
63 } // namespace cxxblas
64
65 #endif // CXXBLAS_SPARSELEVEL2_HECRSMV_H
```

7.131 sparselevel2.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_SPARSELEVEL2_SPARSELEVEL2_H
34 #define CXXBLAS_SPARSELEVEL2_SPARSELEVEL2_H 1
35
36 namespace cxxblas {
37
38     template<typename IndexType>
39     char getIndexBaseChar(IndexType x);
40
41 } // namespace cxxblas
42
43 #include "cxxblas/sparselevel2/gecrsmv.h"
44 #include "cxxblas/sparselevel2/heccsmv.h"
45 #include "cxxblas/sparselevel2/hecrsmv.h"
46 #include "cxxblas/sparselevel2/syccsmv.h"
47 #include "cxxblas/sparselevel2/sycrsmv.h"
48 #include "cxxblas/sparselevel2/trccssv.h"
49 #include "cxxblas/sparselevel2/trcrssv.h"
50
51 #endif // CXXBLAS_SPARSELEVEL2_SPARSELEVEL2_H
```

7.132 syccsmv.h

```
1 /*
2  *   Copyright (c) 2007-2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_SPARSELEVEL2_SYCCSMV_H
34 #define CXXBLAS_SPARSELEVEL2_SYCCSMV_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_SYCCSMV 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
43              typename VY>
44     void syccsmv(StorageUpLo upLo, IndexType n, const ALPHA &alpha, const MA *A,
45                  const IndexType *ia, const IndexType *ja, const VX *x, const BETA &beta, VY *y);
46
47 #ifdef HAVE_SPARSEBLAS
48
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger
51     syccsmv(StorageUpLo upLo, IndexType n, const float &alpha, const float *A, const IndexType *ia,
52             const IndexType *ja, const float *x, const float &beta, float *y);
53
54     template<typename IndexType>
55     typename If<IndexType>::isBlasCompatibleInteger
56     syccsmv(StorageUpLo upLo, IndexType n, const double &alpha, const double *A,
57             const IndexType *ia, const IndexType *ja, const double *x, const double &beta,
58             double *y);
59
60     template<typename IndexType>
61     typename If<IndexType>::isBlasCompatibleInteger
62     syccsmv(StorageUpLo upLo, IndexType n, const ComplexFloat &alpha, const ComplexFloat *A,
63             const IndexType *ia, const IndexType *ja, const ComplexFloat *x,
64             const ComplexFloat &beta, ComplexFloat *y);
65
66     template<typename IndexType>
67     typename If<IndexType>::isBlasCompatibleInteger
68     syccsmv(StorageUpLo upLo, IndexType n, const ComplexDouble &alpha, const ComplexDouble *A,
69             const IndexType *ia, const IndexType *ja, const ComplexDouble *x,
70             const ComplexDouble &beta, ComplexDouble *y);
71
72 #endif // HAVE_SPARSEBLAS
73
74 } // namespace cxxblas
75
76 #endif // CXXBLAS_SPARSELEVEL2_SYCCSMV_H
```

7.133 sycrsmv.h

```
1 /*
2  *   Copyright (c) 2007-2012, Michael Lehn
```

```
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_SPARSELEVEL2_SYCRSMV_H
34 #define CXXBLAS_SPARSELEVEL2_SYCRSMV_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_SYCRSMV 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename VX, typename BETA,
43              typename VY>
44     void sycrsmv(StorageUpLo upLo, IndexType n, const ALPHA &alpha, const MA *A,
45                  const IndexType *ia, const IndexType *ja, const VX *x, const BETA &beta, VY *y);
46
47 #ifdef HAVE_SPARSEBLAS
48
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger
51     sycrsmv(StorageUpLo upLo, IndexType n, const float &alpha, const float *A, const IndexType *ia,
52             const IndexType *ja, const float *x, const float &beta, float *y);
53
54     template<typename IndexType>
55     typename If<IndexType>::isBlasCompatibleInteger
56     sycrsmv(StorageUpLo upLo, IndexType n, const double &alpha, const double *A,
57             const IndexType *ia, const IndexType *ja, const double *x, const double &beta,
58             double *y);
59
60     template<typename IndexType>
61     typename If<IndexType>::isBlasCompatibleInteger
62     sycrsmv(StorageUpLo upLo, IndexType n, const ComplexFloat &alpha, const ComplexFloat *A,
63             const IndexType *ia, const IndexType *ja, const ComplexFloat *x,
64             const ComplexFloat &beta, ComplexFloat *y);
65
66     template<typename IndexType>
67     typename If<IndexType>::isBlasCompatibleInteger
68     sycrsmv(StorageUpLo upLo, IndexType n, const ComplexDouble &alpha, const ComplexDouble *A,
69             const IndexType *ia, const IndexType *ja, const ComplexDouble *x,
70             const ComplexDouble &beta, ComplexDouble *y);
71
72 #endif // HAVE_SPARSEBLAS
73
74 } // namespace cxxblas
75
76 #endif // CXXBLAS_SPARSELEVEL2_SYCRSMV_H
```

7.134 trccssv.h

```
1 /*
2  *   Copyright (c) 2013, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
```

```
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_SPARSELEVEL2_TRCCSSV_H
34 #define CXXBLAS_SPARSELEVEL2_TRCCSSV_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40 #ifdef HAVE_SPARSEBLAS
41
42 #   define HAVE_CXXBLAS_TRCCSSV 1
43
44     template<typename IndexType>
45     typename If<IndexType>::isBlasCompatibleInteger
46     trccssv(StorageUpLo upLo, Transpose trans, IndexType n, const float &alpha, const float *A,
47             const IndexType *ia, const IndexType *ja, const float *x, float *y);
48
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger
51     trccssv(StorageUpLo upLo, Transpose trans, IndexType n, const double &alpha, const double *A,
52             const IndexType *ia, const IndexType *ja, const double *x, double *y);
53
54     template<typename IndexType>
55     typename If<IndexType>::isBlasCompatibleInteger
56     trccssv(StorageUpLo upLo, Transpose trans, IndexType n, const ComplexFloat &alpha,
57             const ComplexFloat *A, const IndexType *ia, const IndexType *ja, const ComplexFloat *x,
58             ComplexFloat *y);
59
60     template<typename IndexType>
61     typename If<IndexType>::isBlasCompatibleInteger
62     trccssv(StorageUpLo upLo, Transpose trans, IndexType n, const ComplexDouble &alpha,
63             const ComplexDouble *A, const IndexType *ia, const IndexType *ja,
64             const ComplexDouble *x, ComplexDouble *y);
65
66 #endif // HAVE_SPARSEBLAS
67
68 } // namespace cxxblas
69
70 #endif // CXXBLAS_SPARSELEVEL2_TRCCSMV_H
```

7.135 trcrssv.h

```
1  /*
2  *   Copyright (c) 2013, Klaus Pototzky
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
```

```
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_SPARSELEVEL2_TRCRSSV_H
34 #define CXXBLAS_SPARSELEVEL2_TRCRSSV_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40 #ifdef HAVE_SPARSEBLAS
41
42 #   define HAVE_CXXBLAS_TRCRSSV 1
43
44     template<typename IndexType>
45     typename If<IndexType>::isBlasCompatibleInteger
46     trcrssv(StorageUpLo upLo, Transpose trans, IndexType n, const float &alpha, const float *A,
47             const IndexType *ia, const IndexType *ja, const float *x, float *y);
48
49     template<typename IndexType>
50     typename If<IndexType>::isBlasCompatibleInteger
51     trcrssv(StorageUpLo upLo, Transpose trans, IndexType n, const double &alpha, const double *A,
52             const IndexType *ia, const IndexType *ja, const double *x, double *y);
53
54     template<typename IndexType>
55     typename If<IndexType>::isBlasCompatibleInteger
56     trcrssv(StorageUpLo upLo, Transpose trans, IndexType n, const ComplexFloat &alpha,
57             const ComplexFloat *A, const IndexType *ia, const IndexType *ja, const ComplexFloat *x,
58             ComplexFloat *y);
59
60     template<typename IndexType>
61     typename If<IndexType>::isBlasCompatibleInteger
62     trcrssv(StorageUpLo upLo, Transpose trans, IndexType n, const ComplexDouble &alpha,
63             const ComplexDouble *A, const IndexType *ia, const IndexType *ja,
64             const ComplexDouble *x, ComplexDouble *y);
65
66 #endif
67
68 } // namespace cxxblas
69
70 #endif // CXXBLAS_SPARSELEVEL2_TRCRSSV_H
```

7.136 gecrsmm.h

```
1  /*
2   *   Copyright (c) 2013, Klaus Pototzky
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
```

```
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_SPARSELEVEL3_GECRSMM_H
34 #define CXXBLAS_SPARSELEVEL3_GECRSMM_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_GECRSMM 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
43              typename MC>
44     void gecrsmm(Transpose transA, IndexType m, IndexType n, IndexType k, const ALPHA &alpha,
45                  const MA *A, const IndexType *ia, const IndexType *ja, const MB *B, IndexType ldB,
46                  const BETA &beta, MC *C, IndexType ldC);
47
48 #ifdef HAVE_SPARSEBLAS
49
50     template<typename IndexType>
51     typename If<IndexType>::isBlasCompatibleInteger
52     gecrsmm(Transpose transA, IndexType m, IndexType n, IndexType k, const float &alpha,
53             const float *A, const IndexType *ia, const IndexType *ja, const float *B, IndexType ldB,
54             const float &beta, float *C, IndexType ldC);
55
56     template<typename IndexType>
57     typename If<IndexType>::isBlasCompatibleInteger
58     gecrsmm(Transpose transA, IndexType m, IndexType n, IndexType k, const double &alpha,
59             const double *A, const IndexType *ia, const IndexType *ja, const double *B,
60             IndexType ldB, const double &beta, double *C, IndexType ldC);
61
62     template<typename IndexType>
63     typename If<IndexType>::isBlasCompatibleInteger
64     gecrsmm(Transpose transA, IndexType m, IndexType n, IndexType k, const ComplexFloat &alpha,
65             const ComplexFloat *A, const IndexType *ia, const IndexType *ja, const ComplexFloat *B,
66             IndexType ldB, const ComplexFloat &beta, ComplexFloat *C, IndexType ldC);
67
68     template<typename IndexType>
69     typename If<IndexType>::isBlasCompatibleInteger
70     gecrsmm(Transpose transA, IndexType m, IndexType n, IndexType k, const ComplexDouble &alpha,
71             const ComplexDouble *A, const IndexType *ia, const IndexType *ja,
72             const ComplexDouble *B, IndexType ldB, const ComplexDouble &beta, ComplexDouble *C,
73             IndexType ldC);
74
75 #endif
76
77 } // namespace cxxblas
78
79 #endif // CXXBLAS_SPARSELEVEL3_GECRSMM_H
```

7.137 heccsmm.h

```
1  /*
2   *   Copyright (c) 2013, Klaus Pototzky
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *    1) Redistributions of source code must retain the above copyright
11  *       notice, this list of conditions and the following disclaimer.
12  *    2) Redistributions in binary form must reproduce the above copyright
13  *       notice, this list of conditions and the following disclaimer in
14  *       the documentation and/or other materials provided with the
15  *       distribution.
16  *    3) Neither the name of the FLENS development group nor the names of
17  *       its contributors may be used to endorse or promote products derived
18  *       from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
```

```
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_SPARSELEVEL3_HECCSMM_H
34 #define CXXBLAS_SPARSELEVEL3_HECCSMM_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_HECCSMM 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
43              typename MC>
44     void heccsmm(StorageUpLo upLo, IndexType m, IndexType n, const ALPHA &alpha, const MA *A,
45                  const IndexType *ia, const IndexType *ja, const MB *B, IndexType ldB,
46                  const BETA &beta, MC *C, IndexType ldC);
47
48 #ifdef HAVE_SPARSEBLAS
49
50     template<typename IndexType>
51     typename If<IndexType>::isBlasCompatibleInteger
52     heccsmm(StorageUpLo upLo, IndexType m, IndexType n, const ComplexFloat &alpha,
53             const ComplexFloat *A, const IndexType *ia, const IndexType *ja, const ComplexFloat *B,
54             IndexType ldB, const ComplexFloat &beta, ComplexFloat *C, IndexType ldC);
55
56     template<typename IndexType>
57     typename If<IndexType>::isBlasCompatibleInteger
58     heccsmm(StorageUpLo upLo, IndexType m, IndexType n, const ComplexDouble &alpha,
59             const ComplexDouble *A, const IndexType *ia, const IndexType *ja,
60             const ComplexDouble *B, IndexType ldB, const ComplexDouble &beta, ComplexDouble *C,
61             IndexType ldC);
62
63 #endif // HAVE_SPARSEBLAS
64
65 } // namespace cxxblas
66
67 #endif // CXXBLAS_SPARSELEVEL3_HECCSMM_H
```

7.138 hecrsmm.h

```
1  /*
2   *   Copyright (c) 2007-2012, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_SPARSELEVEL3_HECRSMM_H
34 #define CXXBLAS_SPARSELEVEL3_HECRSMM_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_HECRSMM 1
39
40 namespace cxxblas {
```

```
41
42      template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
43               typename MC>
44      void hecrsmm(StorageUpLo upLo, IndexType m, IndexType n, const ALPHA &alpha, const MA *A,
45                   const IndexType *ia, const IndexType *ja, const MB *B, IndexType ldB,
46                   const BETA &beta, MC *C, IndexType ldC);
47
48  #ifdef HAVE_SPARSEBLAS
49
50      template<typename IndexType>
51      typename If<IndexType>::isBlasCompatibleInteger
52      hecrsmm(StorageUpLo upLo, IndexType m, IndexType n, const ComplexFloat &alpha,
53              const ComplexFloat *A, const IndexType *ia, const IndexType *ja, const ComplexFloat *B,
54              IndexType ldB, const ComplexFloat &beta, ComplexFloat *C, IndexType ldC);
55
56      template<typename IndexType>
57      typename If<IndexType>::isBlasCompatibleInteger
58      hecrsmm(StorageUpLo upLo, IndexType m, IndexType n, const ComplexDouble &alpha,
59              const ComplexDouble *A, const IndexType *ia, const IndexType *ja,
60              const ComplexDouble *B, IndexType ldB, const ComplexDouble &beta, ComplexDouble *C,
61              IndexType ldC);
62
63  #endif // HAVE_SPARSEBLAS
64
65  } // namespace cxxblas
66
67  #endif // CXXBLAS_SPARSELEVEL3_HECRSMM_H
```

7.139 sparselevel3.h

```
1  /*
2   *   Copyright (c) 2012, Michael Lehn
3   *
4   *   All rights reserved.
5   *
6   *   Redistribution and use in source and binary forms, with or without
7   *   modification, are permitted provided that the following conditions
8   *   are met:
9   *
10  *   1) Redistributions of source code must retain the above copyright
11  *      notice, this list of conditions and the following disclaimer.
12  *   2) Redistributions in binary form must reproduce the above copyright
13  *      notice, this list of conditions and the following disclaimer in
14  *      the documentation and/or other materials provided with the
15  *      distribution.
16  *   3) Neither the name of the FLENS development group nor the names of
17  *      its contributors may be used to endorse or promote products derived
18  *      from this software without specific prior written permission.
19  *
20  *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33  #ifndef CXXBLAS_SPARSELEVEL3_SPARSELEVEL3_H
34  #define CXXBLAS_SPARSELEVEL3_SPARSELEVEL3_H 1
35
36  #include "cxxblas/sparselevel3/gecrsmm.h"
37  #include "cxxblas/sparselevel3/heccsmm.h"
38  #include "cxxblas/sparselevel3/hecrsmm.h"
39  #include "cxxblas/sparselevel3/syccsmm.h"
40  #include "cxxblas/sparselevel3/sycrsmm.h"
41  #include "cxxblas/sparselevel3/trccssm.h"
42  #include "cxxblas/sparselevel3/trcrssm.h"
43
44  #endif // CXXBLAS_SPARSELEVEL2_SPARSELEVEL2_H
```

7.140 syccsmm.h

```
1  /*
2   *   Copyright (c) 2013, Klaus Pototzky
```

```
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_SPARSELEVEL3_SYCCSMM_H
34 #define CXXBLAS_SPARSELEVEL3_SYCCSMM_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_SYCCSMM 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
43              typename MC>
44     void syccsmm(StorageUpLo upLo, IndexType m, IndexType n, const ALPHA &alpha, const MA *A,
45                  const IndexType *ia, const IndexType *ja, const MB *B, IndexType ldB,
46                  const BETA &beta, MC *C, IndexType ldC);
47
48 #ifdef HAVE_SPARSEBLAS
49
50     template<typename IndexType>
51     typename If<IndexType>::isBlasCompatibleInteger
52     syccsmm(StorageUpLo upLo, IndexType m, IndexType n, const float &alpha, const float *A,
53             const IndexType *ia, const IndexType *ja, const float *B, IndexType ldB,
54             const float &beta, float *C, IndexType ldC);
55
56     template<typename IndexType>
57     typename If<IndexType>::isBlasCompatibleInteger
58     syccsmm(StorageUpLo upLo, IndexType m, IndexType n, const double &alpha, const double *A,
59             const IndexType *ia, const IndexType *ja, const double *B, IndexType ldB,
60             const double &beta, double *C, IndexType ldC);
61
62     template<typename IndexType>
63     typename If<IndexType>::isBlasCompatibleInteger
64     syccsmm(StorageUpLo upLo, IndexType m, IndexType n, const ComplexFloat &alpha,
65             const ComplexFloat *A, const IndexType *ia, const IndexType *ja, const ComplexFloat *B,
66             IndexType ldB, const ComplexFloat &beta, ComplexFloat *C, IndexType ldC);
67
68     template<typename IndexType>
69     typename If<IndexType>::isBlasCompatibleInteger
70     syccsmm(StorageUpLo upLo, IndexType m, IndexType n, const ComplexDouble &alpha,
71             const ComplexDouble *A, const IndexType *ia, const IndexType *ja,
72             const ComplexDouble *B, IndexType ldB, const ComplexDouble &beta, ComplexDouble *C,
73             IndexType ldC);
74
75 #endif // HAVE_SPARSEBLAS
76
77 } // namespace cxxblas
78
79 #endif // CXXBLAS_SPARSELEVEL3_SYCCSMM_H
```

7.141 sycrsmm.h

```
1 /*
2  *   Copyright (c) 2007-2012, Michael Lehn
3  *
```

```
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_SPARSELEVEL3_SYCRSMM_H
34 #define CXXBLAS_SPARSELEVEL3_SYCRSMM_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 #define HAVE_CXXBLAS_SYCRSMM 1
39
40 namespace cxxblas {
41
42     template<typename IndexType, typename ALPHA, typename MA, typename MB, typename BETA,
43              typename MC>
44     void sycrsmm(StorageUpLo upLo, IndexType m, IndexType n, const ALPHA &alpha, const MA *A,
45                  const IndexType *ia, const IndexType *ja, const MB *B, IndexType ldB,
46                  const BETA &beta, MC *C, IndexType ldC);
47
48 #ifdef HAVE_SPARSEBLAS
49
50     template<typename IndexType>
51     typename If<IndexType>::isBlasCompatibleInteger
52     sycrsmm(StorageUpLo upLo, IndexType m, IndexType n, const float &alpha, const float *A,
53             const IndexType *ia, const IndexType *ja, const float *B, IndexType ldB,
54             const float &beta, float *C, IndexType ldC);
55
56     template<typename IndexType>
57     typename If<IndexType>::isBlasCompatibleInteger
58     sycrsmm(StorageUpLo upLo, IndexType m, IndexType n, const double &alpha, const double *A,
59             const IndexType *ia, const IndexType *ja, const double *B, IndexType ldB,
60             const double &beta, double *C, IndexType ldC);
61
62     template<typename IndexType>
63     typename If<IndexType>::isBlasCompatibleInteger
64     sycrsmm(StorageUpLo upLo, IndexType m, IndexType n, const ComplexFloat &alpha,
65             const ComplexFloat *A, const IndexType *ia, const IndexType *ja, const ComplexFloat *B,
66             IndexType ldB, const ComplexFloat &beta, ComplexFloat *C, IndexType ldC);
67
68     template<typename IndexType>
69     typename If<IndexType>::isBlasCompatibleInteger
70     sycrsmm(StorageUpLo upLo, IndexType m, IndexType n, const ComplexDouble &alpha,
71             const ComplexDouble *A, const IndexType *ia, const IndexType *ja,
72             const ComplexDouble *B, IndexType ldB, const ComplexDouble &beta, ComplexDouble *C,
73             IndexType ldC);
74
75 #endif // HAVE_SPARSEBLAS
76
77 } // namespace cxxblas
78
79 #endif // CXXBLAS_SPARSELEVEL3_SYCRSMM_H
```

7.142 trccssm.h

```
1 /*
2  *   Copyright (c) 2007-2012, Michael Lehn
3  *
4  *   All rights reserved.
```

```
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_SPARSELEVEL3_TRCCSSM_H
34 #define CXXBLAS_SPARSELEVEL3_TRCCSSM_H 1
35
36 #include "cxxblas/typedefs.h"
37
38 namespace cxxblas {
39
40 #ifdef HAVE_SPARSEBLAS
41
42 #   define HAVE_CXXBLAS_TRCCSSM 1
43
44     template<typename IndexType>
45     typename If<IndexType>::isBlasCompatibleInteger
46     trccssm(StorageUpLo upLo, Transpose trans, IndexType m, IndexType n, const float &alpha,
47             const float *A, const IndexType *ia, const IndexType *ja, const float *B, IndexType ldB,
48             float *C, IndexType ldC);
49
50     template<typename IndexType>
51     typename If<IndexType>::isBlasCompatibleInteger
52     trccssm(StorageUpLo upLo, Transpose trans, IndexType m, IndexType n, const double &alpha,
53             const double *A, const IndexType *ia, const IndexType *ja, const double *B,
54             IndexType ldB, double *C, IndexType ldC);
55
56     template<typename IndexType>
57     typename If<IndexType>::isBlasCompatibleInteger
58     trccssm(StorageUpLo upLo, Transpose trans, IndexType m, IndexType n, const ComplexFloat &alpha,
59             const ComplexFloat *A, const IndexType *ia, const IndexType *ja, const ComplexFloat *B,
60             IndexType ldB, ComplexFloat *C, IndexType ldC);
61
62     template<typename IndexType>
63     typename If<IndexType>::isBlasCompatibleInteger
64     trccssm(StorageUpLo upLo, Transpose trans, IndexType m, IndexType n, const ComplexDouble &alpha,
65             const ComplexDouble *A, const IndexType *ia, const IndexType *ja,
66             const ComplexDouble *B, IndexType ldB, ComplexDouble *C, IndexType ldC);
67
68 #endif // HAVE_SPARSEBLAS
69
70 } // namespace cxxblas
71
72 #endif // CXXBLAS_SPARSELEVEL3_TRCCSSM_H
```

7.143 trcrssm.h

```
1  /*
2  *   Copyright (c) 2007-2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
```

```
 *      notice, this list of conditions and the following disclaimer in
 *      the documentation and/or other materials provided with the
 *      distribution.
 *   3) Neither the name of the FLENS development group nor the names of
 *      its contributors may be used to endorse or promote products derived
 *      from this software without specific prior written permission.
 *
 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

#ifndef CXXBLAS_SPARSELEVEL3_TRCRSSM_H
#define CXXBLAS_SPARSELEVEL3_TRCRSSM_H 1

#include "cxxblas/typedefs.h"

namespace cxxblas {

#ifdef HAVE_SPARSEBLAS

#   define HAVE_CXXBLAS_TRCRSSM 1

    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    trcrssm(StorageUpLo upLo, Transpose trans, IndexType m, IndexType n, const float &alpha,
            const float *A, const IndexType *ia, const IndexType *ja, const float *B, IndexType ldB,
            float *C, IndexType ldC);

    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    trcrssm(StorageUpLo upLo, Transpose trans, IndexType m, IndexType n, const double &alpha,
            const double *A, const IndexType *ia, const IndexType *ja, const double *B,
            IndexType ldB, double *C, IndexType ldC);

    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    trcrssm(StorageUpLo upLo, Transpose trans, IndexType m, IndexType n, const ComplexFloat &alpha,
            const ComplexFloat *A, const IndexType *ia, const IndexType *ja, const ComplexFloat *B,
            IndexType ldB, ComplexFloat *C, IndexType ldC);

    template<typename IndexType>
    typename If<IndexType>::isBlasCompatibleInteger
    trcrssm(StorageUpLo upLo, Transpose trans, IndexType m, IndexType n, const ComplexDouble &alpha,
            const ComplexDouble *A, const IndexType *ia, const IndexType *ja,
            const ComplexDouble *B, IndexType ldB, ComplexDouble *C, IndexType ldC);

#endif // HAVE_SPARSEBLAS

} // namespace cxxblas

#endif // CXXBLAS_SPARSELEVEL3_TRCRSSM_H
```

7.144 tinylevel1.h

```
/*
 *   Copyright (c) 2012, Michael Lehn
 *
 *   All rights reserved.
 *
 *   Redistribution and use in source and binary forms, with or without
 *   modification, are permitted provided that the following conditions
 *   are met:
 *
 *   1) Redistributions of source code must retain the above copyright
 *      notice, this list of conditions and the following disclaimer.
 *   2) Redistributions in binary form must reproduce the above copyright
 *      notice, this list of conditions and the following disclaimer in
 *      the documentation and/or other materials provided with the
 *      distribution.
 *   3) Neither the name of the FLENS development group nor the names of
 *      its contributors may be used to endorse or promote products derived
 *      from this software without specific prior written permission.
 *
 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
```

```
21  *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22  *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23  *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24  *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25  *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26  *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27  *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28  *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29  *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30  *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31  */
32
33 #ifndef CXXBLAS_TINYLEVEL1_TINYLEVEL1_H
34 #define CXXBLAS_TINYLEVEL1_TINYLEVEL1_H 1
35
36 #include "cxxblas/tinylevel1/acxpby.h"
37 #include "cxxblas/tinylevel1/acxpy.h"
38 #include "cxxblas/tinylevel1/axpby.h"
39 #include "cxxblas/tinylevel1/axpy.h"
40 #include "cxxblas/tinylevel1/copy.h"
41 #include "cxxblas/tinylevel1/ccopy.h"
42 #include "cxxblas/tinylevel1/geaxpy.h"
43 #include "cxxblas/tinylevel1/gecopy.h"
44 #include "cxxblas/tinylevel1/gerscal.h"
45 #include "cxxblas/tinylevel1/gescal.h"
46 #include "cxxblas/tinylevel1/rscal.h"
47 #include "cxxblas/tinylevel1/scal.h"
48
49 #endif // CXXBLAS_TINYLEVEL1_TINYLEVEL1_H
```

7.145 tinylevel2.h

```
1 /*
2  *   Copyright (c) 2012, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *    1) Redistributions of source code must retain the above copyright
11 *       notice, this list of conditions and the following disclaimer.
12 *    2) Redistributions in binary form must reproduce the above copyright
13 *       notice, this list of conditions and the following disclaimer in
14 *       the documentation and/or other materials provided with the
15 *       distribution.
16 *    3) Neither the name of the FLENS development group nor the names of
17 *       its contributors may be used to endorse or promote products derived
18 *       from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_TINYLEVEL2_TINYLEVEL2_H
34 #define CXXBLAS_TINYLEVEL2_TINYLEVEL2_H 1
35
36 #include "cxxblas/tinylevel2/gemv.h"
37
38 #endif // CXXBLAS_TINYLEVEL2_TINYLEVEL2_H
```

7.146 typedefs.h

```
1 /*
2  *   Copyright (c) 2009, Michael Lehn
3  *
4  *   All rights reserved.
5  *
6  *   Redistribution and use in source and binary forms, with or without
```

```
7  *   modification, are permitted provided that the following conditions
8  *   are met:
9  *
10 *   1) Redistributions of source code must retain the above copyright
11 *      notice, this list of conditions and the following disclaimer.
12 *   2) Redistributions in binary form must reproduce the above copyright
13 *      notice, this list of conditions and the following disclaimer in
14 *      the documentation and/or other materials provided with the
15 *      distribution.
16 *   3) Neither the name of the FLENS development group nor the names of
17 *      its contributors may be used to endorse or promote products derived
18 *      from this software without specific prior written permission.
19 *
20 *   THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
21 *   "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
22 *   LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR
23 *   A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT
24 *   OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
25 *   SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT
26 *   LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,
27 *   DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY
28 *   THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
29 *   (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
30 *   OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
31 */
32
33 #ifndef CXXBLAS_TYPEDEFS_H
34 #define CXXBLAS_TYPEDEFS_H 1
35
36 // #include <complex>
37
38 namespace cxxblas {
39
40 #ifndef CXXBLAS_COMPLEX_TYPES
41 #   define CXXBLAS_COMPLEX_TYPES 1
42     typedef std::complex<float> ComplexFloat;
43     typedef std::complex<double> ComplexDouble;
44 #endif // CXXBLAS_COMPLEX_TYPES
45
46     enum StorageOrder { RowMajor, ColMajor };
47
48     enum StorageUpLo { Upper = 'U', Lower = 'L' };
49
50     enum Diag { Unit = 'U', NonUnit = 'N' };
51
52     enum Side { Left = 'L', Right = 'R' };
53
54     enum Transpose { NoTrans = 0, Conj = 1, Trans = 2, ConjTrans = 3 };
55
56 } // namespace cxxblas
57
58 #endif // CXXBLAS_TYPEDEFS_H
```

7.147 array.hpp

```
1 #ifndef LIBRAPID_ARRAY
2 #define LIBRAPID_ARRAY
3
4 #include "sizetype.hpp"
5 #include "storage.hpp"
6 #include "arrayContainer.hpp"
7 #include "operations.hpp"
8 #include "function.hpp"
9 #include "assignOps.hpp"
10
11 #endif // LIBRAPID_ARRAY
```

7.148 arrayContainer.hpp

```
1 #ifndef LIBRAPID_ARRAY_ARRAY_CONTAINER_HPP
2 #define LIBRAPID_ARRAY_ARRAY_CONTAINER_HPP
3
4 namespace librapid {
5     namespace typetraits {
6         template<typename ShapeType_, typename StorageType_>
7         struct TypeInfo<ArrayContainer<ShapeType_, StorageType_» {
8             static constexpr bool isLibRapidType = true;
9             using Scalar                         = typename TypeInfo<StorageType_>::Scalar;
10         };
```

```
11      } // namespace typetraits
12
13      template<typename ShapeType_, typename StorageType_>
14      class ArrayContainer {
15      public:
16          using StorageType = StorageType_;
17          using ShapeType   = ShapeType_;
18          using SizeType    = typename ShapeType::SizeType;
19          using Scalar      = typename StorageType::Scalar;
20          using Packet      = typename typetraits::TypeInfo<Scalar>::Packet;
21
23          ArrayContainer() = default;
24
27          LIBRAPID_ALWAYS_INLINE explicit ArrayContainer(const ShapeType &shape);
28
33          LIBRAPID_ALWAYS_INLINE ArrayContainer(const ShapeType &shape, const Scalar &value);
34
37          LIBRAPID_ALWAYS_INLINE explicit ArrayContainer(ShapeType &&shape);
38
41          LIBRAPID_ALWAYS_INLINE ArrayContainer(const ArrayContainer &other) = default;
42
45          LIBRAPID_ALWAYS_INLINE ArrayContainer(ArrayContainer &&other) noexcept = default;
46
52          template<typename Functor_, typename... Args>
53          LIBRAPID_ALWAYS_INLINE explicit ArrayContainer(
54            const detail::Function<Functor_, Args...> &function) LIBRAPID_RELEASE_NOEXCEPT;
55
59          LIBRAPID_ALWAYS_INLINE ArrayContainer &operator=(const ArrayContainer &other) = default;
60
64          LIBRAPID_ALWAYS_INLINE ArrayContainer &operator=(ArrayContainer &&other) noexcept = default;
65
72          template<typename Functor_, typename... Args>
73          LIBRAPID_ALWAYS_INLINE ArrayContainer &
74          operator=(const detail::Function<Functor_, Args...> &function);
75
78          LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE const ShapeType &shape() const noexcept;
79
83          LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Packet packet(size_t index) const;
84
88          LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Scalar scalar(size_t index) const;
89
93          LIBRAPID_ALWAYS_INLINE void writePacket(size_t index, const Packet &value);
94
98          LIBRAPID_ALWAYS_INLINE void write(size_t index, const Scalar &value);
99
100     public:
101         ShapeType m_shape;
102         StorageType m_storage;
103     };
104
105     template<typename ShapeType_, typename StorageType_>
106     ArrayContainer<ShapeType_, StorageType_>::ArrayContainer(const ShapeType &shape) :
107             m_shape(shape), m_storage(shape.size()) {}
108
109     template<typename ShapeType_, typename StorageType_>
110     ArrayContainer<ShapeType_, StorageType_>::ArrayContainer(const ShapeType &shape,
111                                                              const Scalar &value) :
112             m_shape(shape),
113             m_storage(shape.size(), value) {}
114
115     template<typename ShapeType_, typename StorageType_>
116     ArrayContainer<ShapeType_, StorageType_>::ArrayContainer(ShapeType &&shape) :
117             m_shape(std::move(shape)), m_storage(m_shape.size()) {}
118
119     template<typename ShapeType_, typename StorageType_>
120     template<typename Functor_, typename... Args>
121     ArrayContainer<ShapeType_, StorageType_>::ArrayContainer(
122       const detail::Function<Functor_, Args...> &function) LIBRAPID_RELEASE_NOEXCEPT
123             : m_shape(function.shape()),
124               m_storage(m_shape.size()) {
125         detail::assign(*this, function);
126     }
127
128     template<typename ShapeType_, typename StorageType_>
129     template<typename Functor_, typename... Args>
130     ArrayContainer<ShapeType_, StorageType_> &ArrayContainer<ShapeType_, StorageType_>::operator=(
131       const detail::Function<Functor_, Args...> &function) {
132         m_storage.resize(function.shape().size(), 0);
133         detail::assign(*this, function);
134         return *this;
135     }
136
137     template<typename ShapeType_, typename StorageType_>
138     auto ArrayContainer<ShapeType_, StorageType_>::shape() const noexcept -> const ShapeType & {
139         return m_shape;
140     }
141
```

```
142     template<typename ShapeType_, typename StorageType_>
143     auto ArrayContainer<ShapeType_, StorageType_>::packet(size_t index) const -> Packet {
144         Packet res;
145         res.load(m_storage.begin() + index);
146         return res;
147     }
148
149     template<typename ShapeType_, typename StorageType_>
150     auto ArrayContainer<ShapeType_, StorageType_>::scalar(size_t index) const -> Scalar {
151         return m_storage[index];
152     }
153
154     template<typename ShapeType_, typename StorageType_>
155     void ArrayContainer<ShapeType_, StorageType_>::writePacket(size_t index, const Packet &value) {
156         value.store(m_storage.begin() + index);
157     }
158
159     template<typename ShapeType_, typename StorageType_>
160     void ArrayContainer<ShapeType_, StorageType_>::write(size_t index, const Scalar &value) {
161         m_storage[index] = value;
162     }
163 } // namespace librapid
164
165 #endif // LIBRAPID_ARRAY_ARRAY_CONTAINER_HPP
```

7.149 assignOps.hpp

```
1 #ifndef LIBRAPID_ARRAY_ASSIGN_OPS_HPP
2 #define LIBRAPID_ARRAY_ASSIGN_OPS_HPP
3
4 namespace librapid::detail {
5     // All assignment operators are forward declared in "forward.hpp" so they can be used
6     // elsewhere. They are defined here.
7
15     template<typename ShapeType_, typename StorageType_, typename Functor_, typename... Args>
16     LIBRAPID_ALWAYS_INLINE void assign(ArrayContainer<ShapeType_, StorageType_> &lhs,
17                                        const detail::Function<Functor_, Args...> &function) {
18         using Scalar                 = typename ArrayContainer<ShapeType_, StorageType_>::Scalar;
19         constexpr int64_t packetWidth = typetraits::TypeInfo<Scalar>::packetWidth;
20
21         const int64_t size       = function.shape().size();
22         const int64_t vectorSize = size - (size % packetWidth);
23
24         // Ensure the function can actually be assigned to the array container
25         static_assert(typetraits::IsSame<Scalar, typename Function<Functor_, Args...>::Scalar>,
26                       "Function return type must be the same as the array container's scalar type");
27         LIBRAPID_ASSERT(lhs.shape() == function.shape(), "Shapes must be equal");
28
29         for (int64_t index = 0; index < vectorSize; index += packetWidth) {
30             lhs.writePacket(index, function.packet(index));
31         }
32
33         // Assign the remaining elements
34         for (int64_t index = vectorSize; index < size; ++index) {
35             lhs.write(index, function.scalar(index));
36         }
37     }
38 } // namespace librapid::detail
39
40 #endif // LIBRAPID_ARRAY_ASSIGN_OPS_HPP
```

7.150 function.hpp

```
1 #ifndef LIBRAPID_ARRAY_FUNCTION_HPP
2 #define LIBRAPID_ARRAY_FUNCTION_HPP
3
4 namespace librapid {
5     namespace typetraits {
6         template<typename Functor_, typename... Args>
7         struct TypeInfo<::librapid::detail::Function<Functor_, Args...>> {
8             static constexpr bool isLibRapidType     = true;
9             using Scalar                             = decltype(std::declval<Functor_>()(
10                std::declval<typename TypeInfo<std::decay_t<Args>>::Scalar>()...));
11             static constexpr bool supportsArithmetic = TypeInfo<Scalar>::supportsArithmetic;
12             static constexpr bool supportsLogical    = TypeInfo<Scalar>::supportsLogical;
13             static constexpr bool supportsBinary     = TypeInfo<Scalar>::supportsBinary;
14         };
15     } // namespace typetraits
16
```

```
17      namespace detail {
18          template<typename Functor_, typename... Args>
19          class Function {
20          public:
21              using Type    = Function<Functor_, Args...>;
22              using Functor = Functor_;
23              using Scalar  = typename typetraits::TypeInfo<Type>::Scalar;
24
25              using Packet = typename typetraits::TypeInfo<Scalar>::Packet;
26
27              Function() = default;
28
32              LIBRAPID_ALWAYS_INLINE explicit Function(Functor &&functor, Args &&...args);
33
36              LIBRAPID_ALWAYS_INLINE Function(const Function &other) = default;
37
40              LIBRAPID_ALWAYS_INLINE Function(Function &&other) noexcept = default;
41
45              LIBRAPID_ALWAYS_INLINE Function &operator=(const Function &other) = default;
46
50              LIBRAPID_ALWAYS_INLINE Function &operator=(Function &&other) noexcept = default;
51
52              LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE auto shape() const {
53                  return std::get<0>(m_args).shape();
54              }
55
59              LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Packet packet(size_t index) const;
60
64              LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Scalar scalar(size_t index) const;
65
66          private:
72              template<size_t... I>
73              LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Packet packetImpl(std::index_sequence<I...>,
74                                                                          size_t index) const;
75
81              template<size_t... I>
82              LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Scalar scalarImpl(std::index_sequence<I...>,
83                                                                          size_t index) const;
84
85              Functor m_functor;
86              std::tuple<Args...> m_args;
87          };
88
89          template<typename Functor, typename... Args>
90          Function<Functor, Args...>::Function(Functor &&functor, Args &&...args) :
91                  m_functor(std::forward<Functor>(functor)), m_args(std::forward<Args>(args)...) {}
92
93          template<typename Functor, typename... Args>
94          typename Function<Functor, Args...>::Packet
95          Function<Functor, Args...>::packet(size_t index) const {
96              return packetImpl(std::make_index_sequence<sizeof...(Args)>(), index);
97          }
98
99          template<typename Functor, typename... Args>
100         template<size_t... I>
101         typename Function<Functor, Args...>::Packet
102         Function<Functor, Args...>::packetImpl(std::index_sequence<I...>, size_t index) const {
103             return m_functor.packet((std::get<I>(m_args).packet(index))...);
104         }
105
106         template<typename Functor, typename... Args>
107         auto Function<Functor, Args...>::scalar(size_t index) const -> Scalar {
108             return scalarImpl(std::make_index_sequence<sizeof...(Args)>(), index);
109         }
110
111         template<typename Functor, typename... Args>
112         template<size_t... I>
113         auto Function<Functor, Args...>::scalarImpl(std::index_sequence<I...>, size_t index) const
114           -> Scalar {
115             return m_functor((std::get<I>(m_args).scalar(index))...);
116         }
117     } // namespace detail
118 } // namespace librapid
119
120 #endif // LIBRAPID_ARRAY_FUNCTION_HPP
```

7.151 operations.hpp

```
1 #ifndef LIBRAPID_ARRAY_OPERATIONS_HPP
2 #define LIBRAPID_ARRAY_OPERATIONS_HPP
3
4 #define LIBRAPID_BINARY_FUNCTOR(NAME_, OP_)                                                \
5     struct NAME_ {                                                                         \
```

```
6         template<typename T, typename V>                                                    \
7         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE auto operator()(const T &lhs,             \
8                                                                   const V &rhs) const {     \
9             return lhs OP_ rhs;                                                             \
10        }                                                                                   \
11                                                                                            \
12        template<typename Packet>                                                           \
13        LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE auto packet(const Packet &lhs,            \
14                                                              const Packet &rhs) const {    \
15            return lhs OP_ rhs;                                                             \
16        }                                                                                   \
17    }
18
19 #define LIBRAPID_BINARY_OPERATION(NAME_, OP_)                                              \
20    template<class LHS, class RHS>                                                          \
21    LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE auto operator OP_(LHS &&lhs, RHS &&rhs)       \
22      LIBRAPID_RELEASE_NOEXCEPT->detail::Function<detail::NAME_, LHS, RHS> {               \
23        LIBRAPID_ASSERT(lhs.shape() == rhs.shape(), "Shapes must be equal");              \
24        return detail::makeFunction<detail::NAME_>(std::forward<LHS>(lhs),               \
25                                                   std::forward<RHS>(rhs));               \
26    }
27
28 namespace librapid {
29    namespace detail {
30        template<typename Functor, typename... Args>
31        auto makeFunction(Args &&...args) {
32            using OperationType = Function<Functor, Args...>;
33            return OperationType(Functor(), std::forward<Args>(args)...);
34        }
35
36        LIBRAPID_BINARY_FUNCTOR(Plus, +);     // a + b
37        LIBRAPID_BINARY_FUNCTOR(Minus, -);    // a - b
38        LIBRAPID_BINARY_FUNCTOR(Multiply, *); // a * b
39        LIBRAPID_BINARY_FUNCTOR(Divide, /);   // a / b
40
41    } // namespace detail
42
43    LIBRAPID_BINARY_OPERATION(Plus, +)
44    LIBRAPID_BINARY_OPERATION(Minus, -)
45    LIBRAPID_BINARY_OPERATION(Multiply, *)
46    LIBRAPID_BINARY_OPERATION(Divide, /)
47 } // namespace librapid
48
49 #endif // LIBRAPID_ARRAY_OPERATIONS_HPP
```

7.152 sizetype.hpp

```
1 #ifndef LIBRAPID_ARRAY_SIZETYPE_HPP
2 #define LIBRAPID_ARRAY_SIZETYPE_HPP
3
4 /*
5  * This file defines the Shape class and some helper functions,
6  * including stride operations.
7  */
8
9 namespace librapid {
10    template<typename T = size_t, size_t N = 32>
11    class Shape {
12    public:
13        using SizeType                        = T;
14        static constexpr size_t MaxDimensions = N;
15
17        Shape() = default;
18
22        template<typename V, typename typetraits::EnableIf<typetraits::CanCast<V, T>::value> = 0>
23        Shape(const std::initializer_list<V> &vals);
24
28        template<typename V, typename typetraits::EnableIf<typetraits::CanCast<V, T>::value> = 0>
29        explicit Shape(const std::vector<V> &vals);
30
33        Shape(const Shape &other) = default;
34
37        Shape(Shape &&other) noexcept = default;
38
43        template<typename V, size_t Dim>
44        Shape(const Shape<V, Dim> &other);
45
51        template<typename V, size_t Dim>
52        Shape(Shape<V, Dim> &&other) noexcept;
53
58        template<typename V, typename typetraits::EnableIf<typetraits::CanCast<V, T>::value> = 0>
59        Shape &operator=(const std::initializer_list<V> &vals);
60
```

```
65          template<typename V, typename typetraits::EnableIf<typetraits::CanCast<V, T>::value> = 0>
66          Shape &operator=(const std::vector<V> &vals);
67
71          Shape &operator=(Shape &&other) noexcept = default;
72
76          static Shape zeros(size_t dims);
77
81          static Shape ones(size_t dims);
82
87          template<typename Index>
88          LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE const T &operator[](Index index) const;
89
94          template<typename Index>
95          LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE T &operator[](Index index);
96
100          LIBRAPID_ALWAYS_INLINE bool operator==(const Shape &other) const;
101
105          LIBRAPID_ALWAYS_INLINE bool operator!=(const Shape &other) const;
106
109          LIBRAPID_NODISCARD T ndim() const;
110
113          LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE T size() const;
114
117          LIBRAPID_NODISCARD std::string str() const;
118
119      private:
120          T m_dims;
121          std::array<T, N> m_data;
122      };
123
124      template<typename T, size_t N>
125      template<typename V, typename typetraits::EnableIf<typetraits::CanCast<V, T>::value»
126      Shape<T, N>::Shape(const std::initializer_list<V> &vals) : m_dims(vals.size()) {
127          LIBRAPID_ASSERT(vals.size() <= N,
128                          "Shape object is limited to {} dimensions. Cannot initialize with {}",
129                          N,
130                          vals.size());
131          for (size_t i = 0; i < vals.size(); ++i) { m_data[i] = *(vals.begin() + i); }
132      }
133
134      template<typename T, size_t N>
135      template<typename V, typename typetraits::EnableIf<typetraits::CanCast<V, T>::value»
136      Shape<T, N>::Shape(const std::vector<V> &vals) : m_dims(vals.size()) {
137          LIBRAPID_ASSERT(vals.size() <= N,
138                          "Shape object is limited to {} dimensions. Cannot initialize with {}",
139                          N,
140                          vals.size());
141          for (size_t i = 0; i < vals.size(); ++i) { m_data[i] = vals[i]; }
142      }
143
144      template<typename T, size_t N>
145      template<typename V, size_t Dim>
146      Shape<T, N>::Shape(const Shape<V, Dim> &other) : m_dims(other.ndim()) {
147          LIBRAPID_ASSERT(other.ndim() <= N,
148                          "Shape object is limited to {} dimensions. Cannot initialize with {}",
149                          N,
150                          other.ndim());
151          for (size_t i = 0; i < m_dims; ++i) { m_data[i] = other[i]; }
152      }
153
154      template<typename T, size_t N>
155      template<typename V, size_t Dim>
156      Shape<T, N>::Shape(Shape<V, Dim> &&other) noexcept : m_dims(other.ndim()) {
157          LIBRAPID_ASSERT(other.ndim() <= N,
158                          "Shape object is limited to {} dimensions. Cannot initialize with {}",
159                          N,
160                          other.ndim());
161          for (size_t i = 0; i < m_dims; ++i) { m_data[i] = other[i]; }
162      }
163
164      template<typename T, size_t N>
165      template<typename V, typename typetraits::EnableIf<typetraits::CanCast<V, T>::value»
166      Shape<T, N> &Shape<T, N>::operator=(const std::initializer_list<V> &vals) {
167          LIBRAPID_ASSERT(vals.size() <= N,
168                          "Shape object is limited to {} dimensions. Cannot initialize with {}",
169                          N,
170                          vals.size());
171          m_dims = vals.size();
172          for (int64_t i = 0; i < vals.size(); ++i) { m_data[i] = *(vals.begin() + i); }
173          return *this;
174      }
175
176      template<typename T, size_t N>
177      template<typename V, typename typetraits::EnableIf<typetraits::CanCast<V, T>::value»
178      Shape<T, N> &Shape<T, N>::operator=(const std::vector<V> &vals) {
179          LIBRAPID_ASSERT(vals.size() <= N,
180                          "Shape object is limited to {} dimensions. Cannot initialize with {}",
```

```
181                         N,
182                         vals.size());
183         m_dims = vals.size();
184         for (int64_t i = 0; i < vals.size(); ++i) { m_data[i] = vals[i]; }
185         return *this;
186     }
187
188     template<typename T, size_t N>
189     Shape<T, N> Shape<T, N>::zeros(size_t dims) {
190         Shape res;
191         res.m_dims = dims;
192         for (size_t i = 0; i < dims; ++i) res.m_data[i] = 0;
193         return res;
194     }
195
196     template<typename T, size_t N>
197     Shape<T, N> Shape<T, N>::ones(size_t dims) {
198         Shape res;
199         res.m_dims = dims;
200         for (size_t i = 0; i < dims; ++i) res.m_data[i] = 1;
201         return res;
202     }
203
204     template<typename T, size_t N>
205     template<typename Index>
206     LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE const T &Shape<T, N>::operator[](Index index) const {
207         return m_data[index];
208     }
209
210     template<typename T, size_t N>
211     template<typename Index>
212     LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE T &Shape<T, N>::operator[](Index index) {
213         return m_data[index];
214     }
215
216     template<typename T, size_t N>
217     LIBRAPID_ALWAYS_INLINE bool Shape<T, N>::operator==(const Shape &other) const {
218         if (m_dims != other.m_dims) return false;
219         for (size_t i = 0; i < m_dims; ++i) {
220             if (m_data[i] != other.m_data[i]) return false;
221         }
222         return true;
223     }
224
225     template<typename T, size_t N>
226     LIBRAPID_ALWAYS_INLINE bool Shape<T, N>::operator!=(const Shape &other) const {
227         return !(*this == other);
228     }
229
230     template<typename T, size_t N>
231     LIBRAPID_NODISCARD T Shape<T, N>::ndim() const {
232         return m_dims;
233     }
234
235     template<typename T, size_t N>
236     LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE T Shape<T, N>::size() const {
237         T res = 1;
238         for (size_t i = 0; i < m_dims; ++i) res *= m_data[i];
239         return res;
240     }
241
242     template<typename T, size_t N>
243     std::string Shape<T, N>::str() const {
244         std::string result("(");
245         for (size_t i = 0; i < m_dims; ++i) {
246             result += fmt::format("{}", m_data[i]);
247             if (i < m_dims - 1) result += std::string(", ");
248         }
249         return std::operator+(result, std::string(")"));
250     }
251
263     template<typename T1, size_t N1, typename T2, size_t N2, typename... Tn, size_t... Nn>
264     LIBRAPID_NODISCARD LIBRAPID_INLINE bool shapesMatch(const Shape<T1, N1> &first,
265                                                         const Shape<T2, N2> &second,
266                                                         const Shape<Tn, Nn> &...shapes) {
267         if constexpr (sizeof...(Tn) == 0) {
268             return first == second;
269         } else {
270             return first == second && shapesMatch(first, shapes...);
271         }
272     }
273
275     template<typename T1, size_t N1, typename T2, size_t N2, typename... Tn, size_t... Nn>
276     LIBRAPID_NODISCARD LIBRAPID_INLINE bool
277     shapesMatch(const std::tuple<Shape<T1, N1>, Shape<T2, N2>, Shape<Tn, Nn>...> &shapes) {
278         if constexpr (sizeof...(Tn) == 0) {
279             return std::get<0>(shapes) == std::get<1>(shapes);
```

```
280         } else {
281             return std::get<0>(shapes) == std::get<1>(shapes) &&
282                    shapesMatch(std::apply(
283                      [](auto, auto, auto... rest) { return std::make_tuple(rest...); }, shapes));
284         }
285     }
286 } // namespace librapid
287
288 // Support FMT printing
289 #ifdef FMT_API
290 LIBRAPID_SIMPLE_IO_IMPL(typename T COMMA size_t N, librapid::Shape<T COMMA N>)
291 #endif // FMT_API
292
293 #endif // LIBRAPID_ARRAY_SIZETYPE_HPP
```

7.153 storage.hpp

```
1 #ifndef LIBRAPID_ARRAY_DENSE_STORAGE_HPP
2 #define LIBRAPID_ARRAY_DENSE_STORAGE_HPP
3
4 /*
5  * This file defines the DenseStorage class, which contains a contiguous
6  * block of memory of a single data type.
7  */
8
9 namespace librapid {
10     namespace typetraits {
11         template<typename Scalar_, typename Allocator_>
12         struct TypeInfo<Storage<Scalar_, Allocator_>> {
13             static constexpr bool isLibRapidType = true;
14             using Scalar                         = Scalar_;
15         };
16     } // namespace typetraits
17
18     template<typename Scalar_, typename Allocator_ = std::allocator<Scalar_>>
19     class Storage {
20     public:
21         using Allocator            = Allocator_;
22         using Scalar               = Scalar_;
23         using Pointer              = typename std::allocator_traits<Allocator>::pointer;
24         using ConstPointer         = typename std::allocator_traits<Allocator>::const_pointer;
25         using Reference            = Scalar &;
26         using ConstReference       = const Scalar &;
27         using SizeType             = typename std::allocator_traits<Allocator>::size_type;
28         using DifferenceType       = typename std::allocator_traits<Allocator>::difference_type;
29         using Iterator             = Pointer;
30         using ConstIterator        = ConstPointer;
31         using ReverseIterator      = std::reverse_iterator<Iterator>;
32         using ConstReverseIterator = std::reverse_iterator<ConstIterator>;
33
35         Storage() = default;
36
41         LIBRAPID_ALWAYS_INLINE explicit Storage(SizeType size,
42                                                 const Allocator &alloc = Allocator());
43
49         LIBRAPID_ALWAYS_INLINE Storage(SizeType size, ConstReference value,
50                                        const Allocator &alloc = Allocator());
51
56         LIBRAPID_ALWAYS_INLINE Storage(const Storage &other, const Allocator &alloc = Allocator());
57
60         LIBRAPID_ALWAYS_INLINE Storage(Storage &&other) noexcept;
61
66         template<typename V>
67         LIBRAPID_ALWAYS_INLINE Storage(const std::initializer_list<V> &list,
68                                        const Allocator &alloc = Allocator());
69
74         template<typename V>
75         LIBRAPID_ALWAYS_INLINE explicit Storage(const std::vector<V> &vec,
76                                                 const Allocator &alloc = Allocator());
77
81         LIBRAPID_ALWAYS_INLINE Storage &operator=(const Storage &other);
82
86         LIBRAPID_ALWAYS_INLINE Storage &operator=(Storage &&other) noexcept;
87
89         ~Storage();
90
94         LIBRAPID_ALWAYS_INLINE void resize(SizeType newSize);
95
99         LIBRAPID_ALWAYS_INLINE void resize(SizeType newSize, int);
100
101         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE SizeType size() const noexcept;
102
103         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstReference operator[](SizeType index) const;
```

```
104         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Reference operator[](SizeType index);
105
106         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Iterator begin() noexcept;
107         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE Iterator end() noexcept;
108
109         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstIterator begin() const noexcept;
110         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstIterator end() const noexcept;
111
112         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstIterator cbegin() const noexcept;
113         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstIterator cend() const noexcept;
114
115         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ReverseIterator rbegin() noexcept;
116         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ReverseIterator rend() noexcept;
117
118         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstReverseIterator rbegin() const noexcept;
119         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstReverseIterator rend() const noexcept;
120
121         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstReverseIterator crbegin() const noexcept;
122         LIBRAPID_NODISCARD LIBRAPID_ALWAYS_INLINE ConstReverseIterator crend() const noexcept;
123
124     private:
129         template<typename P>
130         LIBRAPID_ALWAYS_INLINE void initData(P begin, P end);
131
135         LIBRAPID_ALWAYS_INLINE void resizeImpl(SizeType newSize, int);
136
140         LIBRAPID_ALWAYS_INLINE void resizeImpl(SizeType newSize);
141
142         Allocator m_allocator;
143         Pointer m_begin = nullptr; // It is more efficient to store pointers to the start
144         Pointer m_end   = nullptr; // and end of the data block than to store the size
145     };
146
147     namespace detail {
148         template<typename A>
149         typename std::allocator_traits<A>::pointer
150         safeAllocate(A &alloc, typename std::allocator_traits<A>::size_type size) {
151             using Traits    = std::allocator_traits<A>;
152             using Pointer   = typename Traits::pointer;
153             using ValueType = typename Traits::value_type;
154             Pointer ptr     = alloc.allocate(size);
155
156             // If the type cannot be trivially constructed, we need to
157             // initialize each value
158             if (!typetraits::TriviallyDefaultConstructible<ValueType>::value) {
159                 for (Pointer p = ptr; p != ptr + size; ++p) {
160                     Traits::construct(alloc, p, ValueType());
161                 }
162             }
163
164             return ptr;
165         }
166
167         template<typename A>
168         void safeDeallocate(A &alloc, typename std::allocator_traits<A>::pointer ptr,
169                             typename std::allocator_traits<A>::size_type size) {
170             using Traits    = std::allocator_traits<A>;
171             using Pointer   = typename Traits::pointer;
172             using ValueType = typename Traits::value_type;
173
174             // If the type cannot be trivially destructed, we need to
175             // destroy each value
176             if (!typetraits::TriviallyDefaultConstructible<ValueType>::value) {
177                 for (Pointer p = ptr; p != ptr + size; ++p) { Traits::destroy(alloc, p); }
178             }
179             Traits::deallocate(alloc, ptr, size);
180         }
181     } // namespace detail
182
183     template<typename T, typename A>
184     Storage<T, A>::Storage(SizeType size, const Allocator &alloc) : m_allocator(alloc) {
185         m_begin = detail::safeAllocate(m_allocator, size);
186         m_end   = m_begin + size;
187     }
188
189     template<typename T, typename A>
190     Storage<T, A>::Storage(SizeType size, ConstReference value, const Allocator &alloc) :
191             m_allocator(alloc) {
192         m_begin = detail::safeAllocate(m_allocator, size);
193         m_end   = m_begin + size;
194         std::fill(m_begin, m_end, value);
195     }
196
197     template<typename T, typename A>
198     Storage<T, A>::Storage(const Storage &other, const Allocator &alloc) :
199             m_allocator(alloc), m_begin(nullptr), m_end(nullptr) {
200         initData(other.begin(), other.end());
```

```
201     }
202
203     template<typename T, typename A>
204     Storage<T, A>::Storage(Storage &&other) noexcept :
205             m_allocator(std::move(other.m_allocator)), m_begin(other.m_begin), m_end(other.m_end) {
206         other.m_begin = nullptr;
207         other.m_end   = nullptr;
208     }
209
210     template<typename T, typename A>
211     template<typename V>
212     Storage<T, A>::Storage(const std::initializer_list<V> &list, const Allocator &alloc) :
213             m_allocator(alloc), m_begin(nullptr), m_end(nullptr) {
214         initData(list.begin(), list.end());
215     }
216
217     template<typename T, typename A>
218     template<typename V>
219     Storage<T, A>::Storage(const std::vector<V> &vector, const Allocator &alloc) :
220             m_allocator(alloc), m_begin(nullptr), m_end(nullptr) {
221         initData(vector.begin(), vector.end());
222     }
223
224     template<typename T, typename A>
225     Storage<T, A> &Storage<T, A>::operator=(const Storage &other) {
226         if (this != &other) {
227             m_allocator =
228               std::allocator_traits<A>::select_on_container_copy_construction(other.m_allocator);
229             resizeImpl(other.size());
230             if (typetraits::TriviallyDefaultConstructible<T>::value) {
231                 // Use a slightly faster memcpy if the type is trivially default constructible
232                 std::uninitialized_copy(other.begin(), other.end(), m_begin);
233             } else {
234                 // Otherwise, use the standard copy algorithm
235                 std::copy(other.begin(), other.end(), m_begin);
236             }
237         }
238         return *this;
239     }
240
241     template<typename T, typename A>
242     Storage<T, A> &Storage<T, A>::operator=(Storage &&other) noexcept {
243         if (this != &other) {
244             m_allocator = std::move(other.m_allocator);
245             std::swap(m_begin, other.m_begin);
246             std::swap(m_end, other.m_end);
247         }
248         return *this;
249     }
250
251     template<typename T, typename A>
252     Storage<T, A>::~Storage() {
253         detail::safeDeallocate(m_allocator, m_begin, size());
254         m_begin = nullptr;
255         m_end   = nullptr;
256     }
257
258     template<typename T, typename A>
259     template<typename P>
260     void Storage<T, A>::initData(P begin, P end) {
261         auto size = static_cast<SizeType>(std::distance(begin, end));
262         m_begin   = detail::safeAllocate(m_allocator, size);
263         m_end     = m_begin + size;
264         std::uninitialized_copy(begin, end, m_begin);
265     }
266
267     template<typename T, typename A>
268     auto Storage<T, A>::size() const noexcept -> SizeType {
269         return static_cast<SizeType>(std::distance(m_begin, m_end));
270     }
271
272     template<typename T, typename A>
273     void Storage<T, A>::resize(SizeType newSize) {
274         resizeImpl(newSize);
275     }
276
277     template<typename T, typename A>
278     void Storage<T, A>::resize(SizeType newSize, int) {
279         resizeImpl(newSize);
280     }
281
282     template<typename T, typename A>
283     LIBRAPID_ALWAYS_INLINE void Storage<T, A>::resizeImpl(SizeType newSize) {
284         SizeType oldSize = size();
285         Pointer oldBegin = m_begin;
286         if (oldSize != newSize) {
287             // Reallocate
```

```
288             m_begin = detail::safeAllocate(m_allocator, newSize);
289             m_end   = m_begin + newSize;
290 
291             if constexpr (typetraits::TriviallyDefaultConstructible<T>::value) {
292                 // Use a slightly faster memcpy if the type is trivially default constructible
293                 std::uninitialized_copy(oldBegin, oldBegin + std::min(oldSize, newSize), m_begin);
294             } else {
295                 // Otherwise, use the standard copy algorithm
296                 std::copy(oldBegin, oldBegin + std::min(oldSize, newSize), m_begin);
297             }
298 
299             detail::safeDeallocate(m_allocator, oldBegin, oldSize);
300         }
301     }
302 
303     template<typename T, typename A>
304     LIBRAPID_ALWAYS_INLINE void Storage<T, A>::resizeImpl(SizeType newSize, int) {
305         SizeType oldSize = size();
306         Pointer oldBegin = m_begin;
307         if (oldSize != newSize) {
308             // Reallocate
309             m_begin = detail::safeAllocate(m_allocator, newSize);
310             m_end   = m_begin + newSize;
311             detail::safeDeallocate(m_allocator, oldBegin, oldSize);
312         }
313     }
314 
315     template<typename T, typename A>
316     auto Storage<T, A>::operator[](Storage<T, A>::SizeType index) const -> ConstReference {
317         LIBRAPID_ASSERT(index < size(), "Index out of bounds");
318         return m_begin[index];
319     }
320 
321     template<typename T, typename A>
322     auto Storage<T, A>::operator[](Storage<T, A>::SizeType index) -> Reference {
323         LIBRAPID_ASSERT(index < size(), "Index out of bounds");
324         return m_begin[index];
325     }
326 
327     template<typename T, typename A>
328     auto Storage<T, A>::begin() noexcept -> Iterator {
329         return m_begin;
330     }
331 
332     template<typename T, typename A>
333     auto Storage<T, A>::end() noexcept -> Iterator {
334         return m_end;
335     }
336 
337     template<typename T, typename A>
338     auto Storage<T, A>::begin() const noexcept -> ConstIterator {
339         return m_begin;
340     }
341 
342     template<typename T, typename A>
343     auto Storage<T, A>::end() const noexcept -> ConstIterator {
344         return m_end;
345     }
346 
347     template<typename T, typename A>
348     auto Storage<T, A>::cbegin() const noexcept -> ConstIterator {
349         return begin();
350     }
351 
352     template<typename T, typename A>
353     auto Storage<T, A>::cend() const noexcept -> ConstIterator {
354         return end();
355     }
356 
357     template<typename T, typename A>
358     auto Storage<T, A>::rbegin() noexcept -> ReverseIterator {
359         return ReverseIterator(m_end);
360     }
361 
362     template<typename T, typename A>
363     auto Storage<T, A>::rend() noexcept -> ReverseIterator {
364         return ReverseIterator(m_begin);
365     }
366 
367     template<typename T, typename A>
368     auto Storage<T, A>::rbegin() const noexcept -> ConstReverseIterator {
369         return ConstReverseIterator(m_end);
370     }
371 
372     template<typename T, typename A>
373     auto Storage<T, A>::rend() const noexcept -> ConstReverseIterator {
374         return ConstReverseIterator(m_begin);
```

```
375     }
376
377     template<typename T, typename A>
378     auto Storage<T, A>::crbegin() const noexcept -> ConstReverseIterator {
379         return rbegin();
380     }
381
382     template<typename T, typename A>
383     auto Storage<T, A>::crend() const noexcept -> ConstReverseIterator {
384         return rend();
385     }
386 } // namespace librapid
387
388 #endif // LIBRAPID_ARRAY_DENSE_STORAGE_HPP
```

7.154 config.hpp

```
1 #ifndef LIBRAPID_CORE_CONFIG_HPP
2 #define LIBRAPID_CORE_CONFIG_HPP
3
4 /*
5  * This file defines a few macros and includes other files, depending on the
6  * compiler/system being used.
7  */
8
9 // Include the precompiled header
10 #include "librapidPch.hpp"
11
12 // Detect Release vs Debug builds
13 #if !defined(NDEBUG)
14 #   define LIBRAPID_DEBUG
15 #   define LIBRAPID_RELEASE_NOEXCEPT
16 #else
17 #   define LIBRAPID_RELEASE
18 #   define LIBRAPID_RELEASE_NOEXCEPT noexcept
19 #endif
20
21 // Detect the operating system
22 #if defined(_WIN32)
23 #   define LIBRAPID_WINDOWS // Windows
24 #   define LIBRAPID_OS_NAME "windows"
25 #elif defined(_WIN64)
26 #   define LIBRAPID_WINDOWS // Windows
27 #   define LIBRAPID_OS_NAME "windows"
28 #elif defined(__CYGWIN__) && !defined(_WIN32)
29 #   define LIBRAPID_WINDOWS // Windows (Cygwin POSIX under Microsoft Window)
30 #   define LIBRAPID_OS_NAME "windows"
31 #elif defined(__ANDROID__)
32 #   define LIBRAPID_ANDROID // Android (implies Linux, so it must come first)
33 #   define LIBRAPID_OS_NAME "android"
34 #elif defined(__linux__)
35 #   define LIBRAPID_LINUX // Debian, Ubuntu, Gentoo, Fedora, openSUSE, RedHat, Centos and other
36 #   define LIBRAPID_UNIX
37 #   define LIBRAPID_OS_NAME "linux"
38 #elif defined(__unix__) || !defined(__APPLE__) && defined(__MACH__)
39 #   include <sys/param.h>
40 #   if defined(BSD)
41 #       define LIBRAPID_BSD // FreeBSD, NetBSD, OpenBSD, DragonFly BSD
42 #       define LIBRAPID_UNIX
43 #       define LIBRAPID_OS_NAME "bsd"
44 #   endif
45 #elif defined(__hpux)
46 #   define LIBRAPID_HP_UX    // HP-UX
47 #   define LIBRAPID_OS_NAME "hp-ux"
48 #elif defined(_AIX)
49 #   define LIBRAPID_AIX  // IBM AIX
50 #   define LIBRAPID_OS_NAME "aix"
51 #elif defined(__APPLE__) && defined(__MACH__) // Apple OSX and iOS (Darwin)
52 #   define LIBRAPID_APPLE
53 #   define LIBRAPID_UNIX
54 #   include <TargetConditionals.h>
55 #   if TARGET_IPHONE_SIMULATOR == 1
56 #       define LIBRAPID_IOS  // Apple iOS
57 #       define LIBRAPID_OS_NAME "ios"
58 #   elif TARGET_OS_IPHONE == 1
59 #       define LIBRAPID_IOS  // Apple iOS
60 #       define LIBRAPID_OS_NAME "ios"
61 #   elif TARGET_OS_MAC == 1
62 #       define LIBRAPID_OSX  // Apple OSX
63 #       define LIBRAPID_OS_NAME "osx"
64 #   endif
65 #elif defined(__sun) && defined(__SVR4)
66 #   define LIBRAPID_SOLARIS // Oracle Solaris, Open Indiana
```

```
67 #   define LIBRAPID_OS_NAME "solaris"
68 #else
69 #   define LIBRAPID_UNKNOWN
70 #   define LIBRAPID_OS_NAME "unknown"
71 #endif
72
73 // Compiler information
74 #if defined(__GNUC__)
75 #   define LIBRAPID_GNU
76 #   define LIBRAPID_COMPILER_NAME "GNU C/C++ Compiler"
77 #endif
78
79 #if defined(__MINGW32__)
80 #   define LIBRAPID_MINGW
81 #   define LIBRAPID_COMPILER_NAME "Mingw or GNU C/C++ Compiler ported for Windows NT"
82 #endif
83
84 #if defined(__MINGW64__)
85 #   define LIBRAPID_MINGW
86 #   define LIBRAPID_COMPILER_NAME                                                    \
87         "Mingw or GNU C/C++ Compiler ported for Windows NT - 64 bits only"
88 #endif
89
90 #if defined(__GFORTRAN__)
91 #   define LIBRAPID_FORTRAN
92 #   define LIBRAPID_COMPILER_NAME "Fortran / GNU Fortran Compiler"
93 #endif
94
95 #if defined(__clang__) && !defined(_MSC_VER)
96 #   define LIBRAPID_CLANG
97 #   define LIBRAPID_COMPILER_NAME "Clang / LLVM Compiler"
98 #endif
99
100 #if defined(_MSC_VER)
101 #   define LIBRAPID_MSVC
102 #   define LIBRAPID_COMPILER_NAME "Microsoft Visual Studio Compiler MSVC"
103 #endif
104
105 #if defined(_MANAGED) || defined(__cplusplus_cli)
106 #   define LIBRAPID_DOTNET
107 #   define LIBRAPID_COMPILER_NAME "Compilation to C++/CLI .NET (CLR) bytecode"
108 #endif
109
110 #if defined(__INTEL_COMPILER)
111 #   define LIBRAPID_INTEL
112 #   define LIBRAPID_COMPILER_NAME "Intel C/C++ Compiler"
113 #endif
114
115 #if defined(__PGI) || defined(__PGIC__)
116 #   define LIBRAPID_PORTLAND
117 #   define LIBRAPID_COMPILER_NAME "Portland Group C/C++ Compiler"
118 #endif
119
120 #if defined(__BORLANDC__)
121 #   define LIBRAPID_BORLAND
122 #   define LIBRAPID_COMPILER_NAME "Borland C++ Compiler"
123 #endif
124
125 #if defined(__EMSCRIPTEN__)
126 #   define LIBRAPID_EMSCRIPTEN
127 #   define LIBRAPID_COMPILER_NAME "emscripten (asm.js - web assembly)"
128 #endif
129
130 #if defined(__asmjs__)
131 #   define LIBRAPID_ASMJS
132 #   define LIBRAPID_COMPILER_NAME "asm.js"
133 #endif
134
135 #if defined(__wasm__)
136 #   define LIBRAPID_WASM
137 #   define LIBRAPID_COMPILER_NAME "WebAssembly"
138 #endif
139
140 #if defined(__NVCC__)
141 #   define LIBRAPID_NVCC
142 #   define LIBRAPID_COMPILER_NAME "NVIDIA NVCC CUDA Compiler"
143 #endif
144
145 #if defined(__CLING__)
146 #   define LIBRAPID_CLING
147 #   define LIBRAPID_COMPILER_NAME "CERN's ROOT Cling C++ Interactive Shell"
148 #endif
149
150 // Check for 32bit vs 64bit
151 #if _WIN32 || _WIN64 // Check windows
152 #   if _WIN64
153 #       define LIBRAPID_64BIT
```

```
154 #   else
155 #       define LIBRAPID_32BIT
156 #   endif
157 #elif __GNUC__
158 #   if __x86_64__ || __ppc64__
159 #       define LIBRAPID_64BIT
160 #   else
161 #       define LIBRAPID_32BIT
162 #   endif
163 #else
164 #   LIBRAPID_64BIT // Default to 64bit
165 #endif
166
167 // Branch prediction hints
168 #ifdef LIBRAPID_20
169 #   define LIBRAPID_LIKELY    [[likely]]
170 #   define LIBRAPID_UNLIKELY [[unlikely]]
171 #else
172 #   define LIBRAPID_LIKELY
173 #   define LIBRAPID_UNLIKELY
174 #endif
175
176 // [[nodiscard]] macro
177 #define LIBRAPID_NODISCARD [[nodiscard]]
178
179 // Nicer FILENAME macro
180 #if defined(FILENAME)
181 #   warning                                                                                    \
182       "The macro 'FILENAME' is already defined. LibRapid's logging system might not function correctly
    as a result"
183 #else
184 #   ifdef LIBRAPID_OS_WINDOWS
185 #       define FILENAME (strrchr(__FILE__, '\\') ? strrchr(__FILE__, '\\') + 1 : __FILE__)
186 #   else
187 #       define FILENAME (strrchr(__FILE__, '/') ? strrchr(__FILE__, '/') + 1 : __FILE__)
188 #   endif
189 #endif
190
191 // Nicer FUNCTION macro
192 #if defined(FUNCTION)
193 #   warning                                                                                    \
194       "The macro 'FUNCTION' is already defined. LibRapid's logging system might not function correctly
    as a result"
195 #else
196 #   if defined(LIBRAPID_MSVC)
197 #       define FUNCTION __FUNCSIG__
198 #   elif defined(LIBRAPID_GNU) || defined(LIBRAPID_CLANG) || defined(LIBRAPID_CLING)
199 #       define FUNCTION __PRETTY_FUNCTION__
200 #   else
201 #       define FUNCTION "Function Signature Unknown"
202 #   endif
203 #endif
204
205 // STRINGIFY
206 #define STRINGIFY(a) STR_IMPL_(a)
207 #define STR_IMPL_(a) #a
208
209 // Assertions, warnings and errors
210 #if defined(LIBRAPID_DEBUG) && !defined(LIBRAPID_ENABLE_ASSERT)
211 #   define LIBRAPID_ENABLE_ASSERT
212 #endif // LIBRAPID_DEBUG && !LIBRAPID_ASSERT
213
214 // Warn the user the first time this is called, but never again
215 #if defined(LIBRAPID_ASSERT)
216 #   define LIBRAPID_WARN_ONCE(msg, ...)                                                        \
217         do {                                                                                   \
218             static bool _alerted = false;                                                      \
219             if (!_alerted) {                                                                   \
220                 LIBRAPID_WARN(msg, __VA_ARGS__);                                               \
221                 _alerted = true;                                                               \
222             }                                                                                  \
223         } while (false)
224 #endif // LIBRAPID_ASSERT
225
226 // Compiler-specific attributes
227 #if defined(LIBRAPID_MSVC)
228 #   include "msvcConfig.hpp"
229 #elif defined(LIBRAPID_GNU) || defined(LIBRAPID_CLANG) || defined(LIBRAPID_CLING)
230 #   include "gnuConfig.hpp"
231 #else
232 #   include "genericConfig.hpp"
233 #endif
234
235 #include "cudaConfig.hpp"
236
237 #ifndef LIBRAPID_MAX_ARRAY_DIMS
238 #   define LIBRAPID_MAX_ARRAY_DIMS 32
```

```
239 #endif // LIBRAPID_MAX_ARRAY_DIMS
240
241 // Code to be run *before* main()
242 #include "preMain.hpp"
243
244 #endif // LIBRAPID_CORE_CONFIG_HPP
```

7.155 core.hpp

```
1 #ifndef LIBRAPID_CORE
2 #define LIBRAPID_CORE
3
4 #include "warningSuppress.hpp"
5 #include "librapidPch.hpp"
6 #include "config.hpp"
7 #include "global.hpp"
8 #include "traits.hpp"
9 #include "typetraits.hpp"
10 #include "helperMacros.hpp"
11
12 #include "forward.hpp"
13
14 // BLAS
15 #include "cxxblas/cxxblas.h"
16 #include "cxxblas/cxxblas.tcc"
17
18 #endif // LIBRAPID_CORE
```

7.156 cudaConfig.hpp

```
1 #ifndef LIBRAPID_CORE_CUDA_CONFIG_HPP
2 #define LIBRAPID_CORE_CUDA_CONFIG_HPP
3
4 // CUDA enabled LibRapid
5 #ifdef LIBRAPID_HAS_CUDA
6
7 // Under MSVC, supress a few warnings
8 #ifdef _MSC_VER
9 #pragma warning(push)
10 #pragma warning(disable : 4505) // unreferenced local function has been removed
11 #endif
12
13 #   define CUDA_NO_HALF // Ensure the cuda_helpers "half" data type is not defined
14 #   include <cublas_v2.h>
15 #   include <cuda.h>
16 #   include <curand.h>
17 #   include <curand_kernel.h>
18
19 #ifdef _MSC_VER
20 #pragma warning(pop)
21 #endif
22
23 #   include "../vendor/jitify/jitify.hpp"
24
25 // cuBLAS API errors
26 const char *getCublasErrorEnum_(cublasStatus_t error);
27
28 //********************//
29 // cuBLAS ERROR CHECK //
30 //********************//
31
32 #   if !defined(cublasSafeCall)
33 #       define cublasSafeCall(err)                                                    \
34             LR_ASSERT_ALWAYS(                                                        \
35               (err) == CUBLAS_STATUS_SUCCESS, "cuBLAS error: {}", getCublasErrorEnum_(err))
36 #   endif
37
38 //********************//
39 //  CUDA ERROR CHECK  //
40 //********************//
41
42 #   if defined(LIBRAPID_ENABLE_ASSERT)
43 #       define cudaSafeCall(call)                                                     \
44             LR_ASSERT(!(call), "CUDA Assertion Failed: {}", cudaGetErrorString(call))
45
46 #       define jitifyCall(call)                                                       \
47             do {                                                                     \
48                 if ((call) != CUDA_SUCCESS) {                                        \
49                     const char *str;                                                 \
```

```
50                      cuGetErrorName(call, &str);                                                 \
51                      throw std::runtime_error(std::string("CUDA JIT failed: ") + str);           \
52                  }                                                                               \
53              } while (0)
54 #   else
55 #       define cudaSafeCall(call) (call)
56
57 #       define jitifyCall(call) (call)
58 #   endif
59
60 #   ifdef _MSC_VER
61 #       pragma warning(default : 4996)
62 #   endif
63
64 #   include "../cuda/helper_cuda.h"
65 #   include "../cuda/helper_functions.h"
66
67 #else
68
69 #endif // LIBRAPID_HAS_CUDA
70
71 namespace librapid::device {
72     // Signifies that host memory should be used
73     struct CPU {};
74
75     // Signifies that device memory should be used
76     struct GPU {};
77 } // namespace librapid::device
78
79 #endif // LIBRAPID_CORE_CUDA_CONFIG_HPP
```

7.157 forward.hpp

```
1 #ifndef LIBRAPID_CORE_FORWARD_HPP
2 #define LIBRAPID_CORE_FORWARD_HPP
3
4 namespace librapid {
5     template<typename Scalar_, typename Allocator_>
6     class Storage;
7
8     template<typename ShapeType_, typename StorageType_>
9     class ArrayContainer;
10
11     namespace detail {
12         template<typename Functor_, typename... Args>
13         class Function;
14
22         template<typename ShapeType_, typename StorageType_, typename Functor_, typename... Args>
23         LIBRAPID_ALWAYS_INLINE void assign(ArrayContainer<ShapeType_, StorageType_> &lhs,
24                                            const detail::Function<Functor_, Args...> &function);
25     } // namespace detail
26 } // namespace librapid
27
28 #endif // LIBRAPID_CORE_FORWARD_HPP
```

7.158 genericConfig.hpp

```
1 #ifndef LIBRAPID_CORE_GNU_CONFIG_HPP
2 #define LIBRAPID_CORE_GNU_CONFIG_HPP
3
4 #define LIBRAPID_INLINE        inline
5 #define LIBRAPID_ALWAYS_INLINE inline
6
7 #if defined(LIBRAPID_ENABLE_ASSERT)
8 #   define LIBRAPID_STATUS(msg, ...)                                                                \
9         do {                                                                                       \
10            std::string funcName = FUNCTION;                                                       \
11            if (funcName.length() > 75) funcName = "<Signature too Long>";                         \
12            int maxLen = librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6,     \
13                                                       (int)strlen(FILENAME) + 6,                  \
14                                                       (int)funcName.length() + 6,                 \
15                                                       (int)strlen("WARN ASSERTION FAILED"));      \
16            fmt::print(fmt::fg(fmt::color::green),                                                 \
17                       "[{0:-^{5}}]\n[File {1:>{6}}]\n[Function "                                  \
18                       "{2:>{7}}]\n[Line {3:>{8}}]\n{4}\n",                                        \
19                       "STATUS",                                                                   \
20                       FILENAME,                                                                   \
21                       funcName,                                                                   \
22                       __LINE__,                                                                   \
```

```
23                      fmt::format(msg __VA_OPT__(, ) __VA_ARGS__),                                    \
24                      maxLen + 5,                                                                     \
25                      maxLen + 0,                                                                     \
26                      maxLen - 4,                                                                     \
27                      maxLen);                                                                        \
28          } while (0)
29
30  #   define LIBRAPID_WARN(msg, ...)                                                                 \
31          do {                                                                                        \
32              std::string funcName = FUNCTION;                                                        \
33              if (funcName.length() > 75) funcName = "<Signature too Long>";                          \
34              int maxLen = librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6,      \
35                                                         (int)strlen(FILENAME) + 6,                   \
36                                                         (int)funcName.length() + 6,                  \
37                                                         (int)strlen("WARN ASSERTION FAILED"));       \
38              fmt::print(fmt::fg(fmt::color::yellow),                                                 \
39                         "[{0:-^{5}}]\n[File {1:>{6}}]\n[Function "                                   \
40                         "{2:>{7}}]\n[Line {3:>{8}}]\n{4}\n",                                         \
41                         "WARNING",                                                                   \
42                         FILENAME,                                                                    \
43                         funcName,                                                                    \
44                         __LINE__,                                                                    \
45                         fmt::format(msg __VA_OPT__(, ) __VA_ARGS__),                                 \
46                         maxLen + 5,                                                                  \
47                         maxLen + 0,                                                                  \
48                         maxLen - 4,                                                                  \
49                         maxLen);                                                                     \
50          } while (0)
51
52  #   define LIBRAPID_ERROR(msg, ...)                                                                \
53          do {                                                                                        \
54              std::string funcName = FUNCTION;                                                        \
55              if (funcName.length() > 75) funcName = "<Signature too Long>";                          \
56              int maxLen = librapiod::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6,     \
57                                                          (int)strlen(FILENAME) + 6,                  \
58                                                          (int)funcName.length() + 6,                 \
59                                                          (int)strlen("WARN ASSERTION FAILED"));      \
60              fmt::print(fmt::fg(fmt::color::red),                                                    \
61                         "[{0:-^{5}}]\n[File {1:>{6}}]\n[Function "                                   \
62                         "{2:>{7}}]\n[Line {3:>{8}}]\n{4}\n",                                         \
63                         "ERROR",                                                                     \
64                         FILENAME,                                                                    \
65                         funcName,                                                                    \
66                         __LINE__,                                                                    \
67                         fmt::format(msg __VA_OPT__(, ) __VA_ARGS__),                                 \
68                         maxLen + 5,                                                                  \
69                         maxLen + 0,                                                                  \
70                         maxLen - 4,                                                                  \
71                         maxLen);                                                                     \
72              if (librapid::global::throwOnAssert) {                                                  \
73                  throw std::runtime_error(formatted);                                                \
74              } else {                                                                                \
75                  fmt::print(fmt::fg(fmt::color::red), formatted);                                    \
76                  std::exit(1);                                                                       \
77              }                                                                                       \
78          } while (0)
79
80  #   define LIBRAPID_WASSERT(cond, msg, ...)                                                        \
81          do {                                                                                        \
82              if (!(cond)) {                                                                          \
83                  std::string funcName = FUNCTION;                                                    \
84                  if (funcName.length() > 75) funcName = "<Signature too Long>";                      \
85                  \ int maxLen =                                                                      \
86                    librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6,             \
87                                                  (int)strlen(FILENAME) + 6,                          \
88                                                  (int)funcName.length() + 6,                         \
89                                                  (int)strlen(#cond) + 6,                             \
90                                                  (int)strlen("WARN ASSERTION FAILED"));              \
91                  fmt::print(fmt::fg(fmt::color::yellow),                                             \
92                             "[{0:-^{6}}]\n[File {1:>{7}}]\n[Function "                               \
93                             "{2:>{8}}]\n[Line {3:>{9}}]\n[Condition "                                \
94                             "{4:>{10}}]\n{5}\n",                                                     \
95                             "WARN ASSERTION FAILED",                                                 \
96                             FILENAME,                                                                \
97                             funcName,                                                                \
98                             __LINE__,                                                                \
99                             #cond,                                                                   \
100                             fmt::format(msg __VA_OPT__(, ) __VA_ARGS__),                            \
101                             maxLen + 5,                                                             \
102                             maxLen + 0,                                                             \
103                             maxLen - 4,                                                             \
104                             maxLen + 0,                                                             \
105                             maxLen - 5);                                                            \
106             }                                                                                       \
107         } while (0)
108
109 #   define LIBRAPID_ASSERT(cond, msg, ...)                                                         \
```

```
110         do {                                                                                                \
111             std::string funcName = FUNCTION;                                                                \
112             if (funcName.length() > 75) funcName = "<Signature too Long>";                                  \
113             if (!(cond)) {                                                                                  \
114                 int maxLen = librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)),              \
115                                                            (int)strlen(FILENAME),                           \
116                                                            (int)funcName.length(),                          \
117                                                            (int)strlen(#cond),                              \
118                                                            (int)strlen("ASSERTION FAILED"));                \
119                 std::string formatted = fmt::format(                                                        \
120                   "[{0:-^{6}}]\n[File {1:>{7}}]\n[Function "                                                \
121                   "{2:>{8}}]\n[Line {3:>{9}}]\n[Condition "                                                 \
122                   "{4:>{10}}]\n{5}\n",                                                                      \
123                   "ASSERTION FAILED",                                                                       \
124                   FILENAME,                                                                                 \
125                   funcName,                                                                                 \
126                   __LINE__,                                                                                 \
127                   #cond,                                                                                    \
128                   fmt::format(msg __VA_OPT__(, ) __VA_ARGS__),                                              \
129                   maxLen + 14,                                                                              \
130                   maxLen + 9,                                                                               \
131                   maxLen + 5,                                                                               \
132                   maxLen + 9,                                                                               \
133                   maxLen + 4);                                                                              \
134                 if (librapid::global::throwOnAssert) {                                                      \
135                     throw std::runtime_error(formatted);                                                    \
136                 } else {                                                                                    \
137                     fmt::print(fmt::fg(fmt::color::red), formatted);                                        \
138                     std::exit(1);                                                                           \
139                 }                                                                                           \
140             }                                                                                               \
141         } while (0)
142 #else
143 #   define LIBRAPID_WARN_ONCE(msg, ...)                                                                     \
144         do {                                                                                                \
145         } while (0)
146 #   define LIBRAPID_STATUS(msg, ...)                                                                        \
147         do {                                                                                                \
148         } while (0)
149 #   define LIBRAPID_WARN(msg, ...)                                                                          \
150         do {                                                                                                \
151         } while (0)
152 #   define LIBRAPID_ERROR(msg, ...)                                                                         \
153         do {                                                                                                \
154         } while (0)
155 #   define LIBRAPID_LOG(msg, ...)                                                                           \
156         do {                                                                                                \
157         } while (0)
158 #   define LIBRAPID_WASSERT(cond, ...)                                                                      \
159         do {                                                                                                \
160         } while (0)
161 #   define LIBRAPID_ASSERT(cond, ...)                                                                       \
162         do {                                                                                                \
163         } while (0)
164 #endif // LIBRAPID_ENABLE_ASSERT
165
166 #endif // LIBRAPID_CORE_GNU_CONFIG_HPP
```

7.159 global.hpp

```
1 #ifndef LIBRAPID_CORE_GLOBAL_HPP
2 #define LIBRAPID_CORE_GLOBAL_HPP
3
4 /*
5  * Global variables required for LibRapid, such as version number, number of threads,
6  * CUDA-related configuration, etc.
7  */
8
9 namespace librapid::global {
10     // Should ASSERT functions error or throw exceptions?
11     extern bool throwOnAssert;
12
14     extern int64_t multithreadThreshold;
15
16     // Number of columns required for a matrix to be parallelized in GEMM
17     extern int64_t gemmMultithreadThreshold;
18
19     // Number of threads used by LibRapid
20     extern int64_t numThreads;
21 } // namespace librapid::global
22
23 #endif // LIBRAPID_CORE_GLOBAL_HPP
```

7.160 gnuConfig.hpp

```
1 #ifndef LIBRAPID_CORE_GNU_CONFIG_HPP
2 #define LIBRAPID_CORE_GNU_CONFIG_HPP
3
4 #define LIBRAPID_INLINE        inline
5 #define LIBRAPID_ALWAYS_INLINE inline __attribute__((always_inline))
6
7 #if defined(LIBRAPID_ENABLE_ASSERT)
8 #   define LIBRAPID_STATUS(msg, ...)                                                                \
9         do {                                                                                       \
10            std::string funcName = FUNCTION;                                                       \
11            if (funcName.length() > 75) funcName = "<Signature too Long>";                         \
12            int maxLen = librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6,     \
13                                                       (int)strlen(FILENAME) + 6,                  \
14                                                       (int)funcName.length() + 6,                 \
15                                                       (int)strlen("WARN ASSERTION FAILED"));      \
16            fmt::print(fmt::fg(fmt::color::green),                                                 \
17                       "[{0:-^{5}}]\n[File {1:>{6}}]\n[Function "                                  \
18                       "{2:>{7}}]\n[Line {3:>{8}}]\n{4}\n",                                        \
19                       "STATUS",                                                                   \
20                       FILENAME,                                                                   \
21                       funcName,                                                                   \
22                       __LINE__,                                                                   \
23                       fmt::format(msg __VA_OPT__(, ) __VA_ARGS__),                                \
24                       maxLen + 5,                                                                 \
25                       maxLen + 0,                                                                 \
26                       maxLen - 4,                                                                 \
27                       maxLen);                                                                    \
28        } while (0)
29
30 #   define LIBRAPID_WARN(msg, ...)                                                                \
31        do {                                                                                       \
32            std::string funcName = FUNCTION;                                                       \
33            if (funcName.length() > 75) funcName = "<Signature too Long>";                         \
34            int maxLen = librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6,     \
35                                                       (int)strlen(FILENAME) + 6,                  \
36                                                       (int)funcName.length() + 6,                 \
37                                                       (int)strlen("WARN ASSERTION FAILED"));      \
38            fmt::print(fmt::fg(fmt::color::yellow),                                                \
39                       "[{0:-^{5}}]\n[File {1:>{6}}]\n[Function "                                  \
40                       "{2:>{7}}]\n[Line {3:>{8}}]\n{4}\n",                                        \
41                       "WARNING",                                                                  \
42                       FILENAME,                                                                   \
43                       funcName,                                                                   \
44                       __LINE__,                                                                   \
45                       fmt::format(msg __VA_OPT__(, ) __VA_ARGS__),                                \
46                       maxLen + 5,                                                                 \
47                       maxLen + 0,                                                                 \
48                       maxLen - 4,                                                                 \
49                       maxLen);                                                                    \
50        } while (0)
51
52 #   define LIBRAPID_ERROR(msg, ...)                                                               \
53        do {                                                                                       \
54            std::string funcName = FUNCTION;                                                       \
55            if (funcName.length() > 75) funcName = "<Signature too Long>";                         \
56            int maxLen = librapiod::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6,    \
57                                                        (int)strlen(FILENAME) + 6,                 \
58                                                        (int)funcName.length() + 6,                \
59                                                        (int)strlen("WARN ASSERTION FAILED"));     \
60            fmt::print(fmt::fg(fmt::color::red),                                                   \
61                       "[{0:-^{5}}]\n[File {1:>{6}}]\n[Function "                                  \
62                       "{2:>{7}}]\n[Line {3:>{8}}]\n{4}\n",                                        \
63                       "ERROR",                                                                    \
64                       FILENAME,                                                                   \
65                       funcName,                                                                   \
66                       __LINE__,                                                                   \
67                       fmt::format(msg __VA_OPT__(, ) __VA_ARGS__),                                \
68                       maxLen + 5,                                                                 \
69                       maxLen + 0,                                                                 \
70                       maxLen - 4,                                                                 \
71                       maxLen);                                                                    \
72            if (librapid::global::throwOnAssert) {                                                 \
73                throw std::runtime_error(formatted);                                               \
74            } else {                                                                               \
75                fmt::print(fmt::fg(fmt::color::red), formatted);                                   \
76                std::exit(1);                                                                      \
77            }                                                                                      \
78        } while (0)
79
80 #   define LIBRAPID_WASSERT(cond, msg, ...)                                                       \
81        do {                                                                                       \
82            if (!(cond)) {                                                                         \
83                std::string funcName = FUNCTION;                                                   \
84                if (funcName.length() > 75) funcName = "<Signature too Long>";                     \
85                \ int maxLen =                                                                     \
```

```
86                   librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6,           \
87                                                 (int)strlen(FILENAME) + 6,                        \
88                                                 (int)funcName.length() + 6,                       \
89                                                 (int)strlen(#cond) + 6,                           \
90                                                 (int)strlen("WARN ASSERTION FAILED"));            \
91                   fmt::print(fmt::fg(fmt::color::yellow),                                         \
92                              "[{0:-^{6}}]\n[File {1:>{7}}]\n[Function "                           \
93                              "{2:>{8}}]\n[Line {3:>{9}}]\n[Condition "                            \
94                              "{4:>{10}}]\n{5}\n",                                                 \
95                              "WARN ASSERTION FAILED",                                             \
96                              FILENAME,                                                            \
97                              funcName,                                                            \
98                              __LINE__,                                                            \
99                              #cond,                                                               \
100                             fmt::format(msg __VA_OPT__(, ) __VA_ARGS__),                         \
101                             maxLen + 5,                                                          \
102                             maxLen + 0,                                                          \
103                             maxLen - 4,                                                          \
104                             maxLen + 0,                                                          \
105                             maxLen - 5);                                                         \
106              }                                                                                   \
107          } while (0)
108
109  #   define LIBRAPID_ASSERT(cond, msg, ...)                                                     \
110          do {                                                                                    \
111              std::string funcName = FUNCTION;                                                    \
112              if (funcName.length() > 75) funcName = "<Signature too Long>";                      \
113              if (!(cond)) {                                                                      \
114                  int maxLen = librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)),  \
115                                                             (int)strlen(FILENAME),               \
116                                                             (int)funcName.length(),              \
117                                                             (int)strlen(#cond),                  \
118                                                             (int)strlen("ASSERTION FAILED"));    \
119                  std::string formatted = fmt::format(                                            \
120                    "[{0:-^{6}}]\n[File {1:>{7}}]\n[Function "                                    \
121                    "{2:>{8}}]\n[Line {3:>{9}}]\n[Condition "                                     \
122                    "{4:>{10}}]\n{5}\n",                                                          \
123                    "ASSERTION FAILED",                                                           \
124                    FILENAME,                                                                     \
125                    funcName,                                                                     \
126                    __LINE__,                                                                     \
127                    #cond,                                                                        \
128                    fmt::format(msg __VA_OPT__(, ) __VA_ARGS__),                                  \
129                    maxLen + 14,                                                                  \
130                    maxLen + 9,                                                                   \
131                    maxLen + 5,                                                                   \
132                    maxLen + 9,                                                                   \
133                    maxLen + 4);                                                                  \
134                  if (librapid::global::throwOnAssert) {                                          \
135                      throw std::runtime_error(formatted);                                        \
136                  } else {                                                                        \
137                      fmt::print(fmt::fg(fmt::color::red), formatted);                            \
138                      std::exit(1);                                                               \
139                  }                                                                               \
140              }                                                                                   \
141          } while (0)
142  #else
143  #   define LIBRAPID_WARN_ONCE(msg, ...)                                                        \
144          do {                                                                                    \
145          } while (0)
146  #   define LIBRAPID_STATUS(msg, ...)                                                           \
147          do {                                                                                    \
148          } while (0)
149  #   define LIBRAPID_WARN(msg, ...)                                                             \
150          do {                                                                                    \
151          } while (0)
152  #   define LIBRAPID_ERROR(msg, ...)                                                            \
153          do {                                                                                    \
154          } while (0)
155  #   define LIBRAPID_LOG(msg, ...)                                                              \
156          do {                                                                                    \
157          } while (0)
158  #   define LIBRAPID_WASSERT(cond, ...)                                                         \
159          do {                                                                                    \
160          } while (0)
161  #   define LIBRAPID_ASSERT(cond, ...)                                                          \
162          do {                                                                                    \
163          } while (0)
164  #endif // LIBRAPID_ENABLE_ASSERT
165
166  #endif // LIBRAPID_CORE_GNU_CONFIG_HPP
```

7.161 helperMacros.hpp

```
1 #ifndef LIBRAPID_CORE_HELPER_MACROS
2 #define LIBRAPID_CORE_HELPER_MACROS
3
4 /*
5  * Defines a set of basic macros for common uses
6  */
7
8 #define COMMA ,
9
10 // Provide {fmt} printing capabilities
11 #define LIBRAPID_SIMPLE_IO_IMPL(TEMPLATE_, TYPE_)                                                \
12     template<TEMPLATE_>                                                                          \
13     struct fmt::formatter<TYPE_> {                                                               \
14         std::string formatStr = "{}";                                                            \
15                                                                                                  \
16         template<typename ParseContext>                                                          \
17         constexpr auto parse(ParseContext &ctx) {                                                \
18             formatStr = "{:";                                                                    \
19             auto it   = ctx.begin();                                                             \
20             for (; it != ctx.end(); ++it) {                                                      \
21                 if (*it == '}') break;                                                           \
22                 formatStr += *it;                                                                \
23             }                                                                                    \
24             formatStr += "}";                                                                    \
25             return it;                                                                           \
26         }                                                                                        \
27                                                                                                  \
28         template<typename FormatContext>                                                         \
29         auto format(const TYPE_ &object, FormatContext &ctx) {                                   \
30             try {                                                                                \
31                 return fmt::format_to(ctx.out(), object.str());                                  \
32             } catch (std::exception & e) { return fmt::format_to(ctx.out(), e.what()); }         \
33         }                                                                                        \
34     };                                                                                           \
35                                                                                                  \
36     template<TEMPLATE_>                                                                          \
37     std::ostream &operator«(std::ostream &os, const TYPE_ &object) {                            \
38         os « object.str();                                                                       \
39         return os;                                                                               \
40     }
41
42 #endif // LIBRAPID_CORE_HELPER_MACROS
```

7.162 librapidPch.hpp

```
1 #ifndef LIBRAPID_CORE_LIBRAPID_PCH_HPP
2 #define LIBRAPID_CORE_LIBRAPID_PCH_HPP
3
4 /*
5  * Include standard library headers and precompile them as part of
6  * librapid. This reduces compile times dramatically.
7  *
8  * Additionally, include the header files of dependencies.
9  */
10
11 // Standard Library
12 #include <algorithm>
13 #include <array>
14 #include <atomic>
15 #include <cfenv>
16 #include <cfloat>
17 #include <cmath>
18 #include <cmath>
19 #include <complex>
20 #include <cstddef>
21 #include <cstdint>
22 #include <cstdlib>
23 #include <cstdlib>
24 #include <fstream>
25 #include <future>
26 #include <iomanip>
27 #include <iostream>
28 #include <iterator>
29 #include <limits>
30 #include <map>
31 #include <memory>
32 #include <mutex>
33 #include <numeric>
34 #include <queue>
35 #include <random>
36 #include <regex>
```

```
37 #include <set>
38 #include <utility>
39
40 #if defined(LIBRAPID_HAS_OMP)
41 #   include <omp.h>
42 #endif // LIBRAPID_HAS_OMP
43
44 #if defined(_WIN32) || defined(_WIN64)
45 #   define WIN32_LEAN_AND_MEAN
46 #   include <Windows.h>
47 #endif
48
49 // Remove a few macros
50 #undef min
51 #undef max
52
53 // fmtlib
54 #include <fmt/core.h>
55 #include <fmt/format.h>
56 #include <fmt/ranges.h>
57 #include <fmt/chrono.h>
58 #include <fmt/compile.h>
59 #include <fmt/color.h>
60 #include <fmt/os.h>
61 #include <fmt/ostream.h>
62 #include <fmt/printf.h>
63 #include <fmt/xchar.h>
64
65 // scnlib
66 #include <scn/scn.h>
67 #include <scn/tuple_return/tuple_return.h>
68
69 // Vc -- SIMD instructions
70 #if defined(_MSC_VER)
71 // For Vc, we need to disable the following warnings
72 #   pragma warning(push)
73 #   pragma warning(disable : 4244) // conversion from 'int' to 'float', possible loss of data
74 #   pragma warning(disable : 4324) // structure was padded due to alignment specifier
75 #   pragma warning(disable : 4127) // conditional expression is constant
76 #endif
77
78 #include <Vc/Vc>
79 #include <Vc/algorithm>
80 #include <Vc/cpuid.h>
81
82 #if defined(_MSC_VER)
83 #   pragma warning(pop)
84 #endif
85
86 #endif // LIBRAPID_CORE_LIBRAPID_PCH_HPP
```

7.163 msvcConfig.hpp

```
1 #ifndef LIBRAPID_CORE_MSVC_CONFIG_HPP
2 #define LIBRAPID_CORE_MSVC_CONFIG_HPP
3
4 #define LIBRAPID_INLINE        inline
5 #define LIBRAPID_ALWAYS_INLINE inline __forceinline
6
7 #if defined(LIBRAPID_ENABLE_ASSERT)
8 #   define LIBRAPID_STATUS(msg, ...)                                                                \
9         do {                                                                                        \
10             std::string funcName = FUNCTION;                                                       \
11             if (funcName.length() > 75) funcName = "<Signature too Long>";                         \
12             int maxLen = librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6,     \
13                                                        (int)strlen(FILENAME) + 6,                  \
14                                                        (int)funcName.length() + 6,                 \
15                                                        (int)strlen("WARN ASSERTION FAILED"));      \
16             fmt::print(fmt::fg(fmt::color::green),                                                 \
17                        "[{0:-^{5}}]\n[File {1:>{6}}]\n[Function "                                  \
18                        "{2:>{7}}]\n[Line {3:>{8}}]\n{4}\n",                                        \
19                        "STATUS",                                                                   \
20                        FILENAME,                                                                   \
21                        funcName,                                                                   \
22                        __LINE__,                                                                   \
23                        fmt::format(msg, __VA_ARGS__),                                              \
24                        maxLen + 5,                                                                 \
25                        maxLen + 0,                                                                 \
26                        maxLen - 4,                                                                 \
27                        maxLen);                                                                    \
28         } while (0)
29
30 #   define LIBRAPID_WARN(msg, ...)                                                                 \
```

```
31          do {                                                                                \
32              std::string funcName = FUNCTION;                                                \
33              if (funcName.length() > 75) funcName = "<Signature too Long>";                  \
34              int maxLen = librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6,      \
35                                                         (int)strlen(FILENAME) + 6,           \
36                                                         (int)funcName.length() + 6,          \
37                                                         (int)strlen("WARN ASSERTION FAILED"));       \
38              fmt::print(fmt::fg(fmt::color::yellow),                                         \
39                         "[{0:-^{5}}]\n[File {1:>{6}}]\n[Function "                           \
40                         "{2:>{7}}]\n[Line {3:>{8}}]\n{4}\n",                                 \
41                         "WARNING",                                                           \
42                         FILENAME,                                                            \
43                         funcName,                                                            \
44                         __LINE__,                                                            \
45                         fmt::format(msg, __VA_ARGS__),                                       \
46                         maxLen + 5,                                                          \
47                         maxLen + 0,                                                          \
48                         maxLen - 4,                                                          \
49                         maxLen);                                                             \
50          } while (0)
51
52  #   define LIBRAPID_ERROR(msg, ...)                                                        \
53          do {                                                                                \
54              std::string funcName = FUNCTION;                                                \
55              if (funcName.length() > 75) funcName = "<Signature too Long>";                  \
56              int maxLen = librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6,      \
57                                                         (int)strlen(FILENAME) + 6,           \
58                                                         (int)funcName.length() + 6,          \
59                                                         (int)strlen("WARN ASSERTION FAILED"));       \
60              std::string formatted = fmt::format(                                            \
61                "[{0:-^{5}}]\n[File {1:>{6}}]\n[Function "                                    \
62                "{2:>{7}}]\n[Line {3:>{8}}]\n{4}\n",                                          \
63                "ERROR",                                                                      \
64                FILENAME,                                                                     \
65                funcName,                                                                     \
66                __LINE__,                                                                     \
67                fmt::format(msg, __VA_ARGS__),                                                \
68                maxLen + 5,                                                                   \
69                maxLen + 0,                                                                   \
70                maxLen - 4,                                                                   \
71                maxLen);                                                                      \
72              if (librapid::global::throwOnAssert) {                                          \
73                  throw std::runtime_error(formatted);                                        \
74              } else {                                                                        \
75                  fmt::print(fmt::fg(fmt::color::red), formatted);                            \
76                  std::exit(1);                                                               \
77              }                                                                               \
78          } while (0)
79
80  #   define LIBRAPID_WASSERT(cond, msg, ...)                                                \
81          std::string funcName = FUNCTION;                                                    \
82          if (funcName.length() > 75) funcName = "<Signature too Long>";                      \
83          do {                                                                                \
84              if (!(cond)) {                                                                  \
85                  int maxLen = librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)) + 6, \
86                                                             (int)strlen(FILENAME) + 6,       \
87                                                             (int)funcName.length() + 6,      \
88                                                             (int)strlen(#cond) + 6,          \
89                                                             (int)strlen("WARN ASSERTION FAILED"));  \
90                  fmt::print(fmt::fg(fmt::color::yellow),                                     \
91                             "[{0:-^{6}}]\n[File {1:>{7}}]\n[Function "                       \
92                             "{2:>{8}}]\n[Line {3:>{9}}]\n[Condition "                        \
93                             "{4:>{10}}]\n{5}\n",                                             \
94                             "WARN ASSERTION FAILED",                                         \
95                             FILENAME,                                                        \
96                             funcName,                                                        \
97                             __LINE__,                                                        \
98                             #cond,                                                           \
99                             fmt::format(msg, __VA_ARGS__),                                   \
100                             maxLen + 5,                                                      \
101                             maxLen + 0,                                                      \
102                             maxLen - 4,                                                      \
103                             maxLen + 0,                                                      \
104                             maxLen - 5);                                                     \
105             }                                                                               \
106         } while (0)
107
108 #   define LIBRAPID_ASSERT(cond, msg, ...)                                                 \
109         do {                                                                                \
110             if (!(cond)) {                                                                  \
111                 std::string funcName = FUNCTION;                                            \
112                 if (funcName.length() > 75) funcName = "<Signature too Long>";              \
113                 int maxLen = librapid::detail::internalMax((int)std::ceil(std::log(__LINE__)),      \
114                                                            (int)strlen(FILENAME),           \
115                                                            (int)funcName.length(),          \
116                                                            (int)strlen(#cond),              \
117                                                            (int)strlen("ASSERTION FAILED"));    \
```

```
118                 std::string formatted = fmt::format(                                                \
119                   "[{0:-^{6}}]\n[File {1:>{7}}]\n[Function "                                        \
120                   "{2:>{8}}]\n[Line {3:>{9}}]\n[Condition "                                         \
121                   "{4:>{10}}]\n{5}\n",                                                              \
122                   "ASSERTION FAILED",                                                               \
123                   FILENAME,                                                                         \
124                   funcName,                                                                         \
125                   __LINE__,                                                                         \
126                   #cond,                                                                            \
127                   fmt::format(msg, __VA_ARGS__),                                                    \
128                   maxLen + 14,                                                                      \
129                   maxLen + 9,                                                                       \
130                   maxLen + 5,                                                                       \
131                   maxLen + 9,                                                                       \
132                   maxLen + 4);                                                                      \
133                 if (librapid::global::throwOnAssert) {                                              \
134                     throw std::runtime_error(formatted);                                            \
135                 } else {                                                                            \
136                     fmt::print(fmt::fg(fmt::color::red), formatted);                                 \
137                     std::exit(1);                                                                   \
138                 }                                                                                   \
139             }                                                                                       \
140         } while (0)
141 #else
142 #   define LIBRAPID_WARN_ONCE(msg, ...)                                                            \
143         do {                                                                                        \
144         } while (0)
145 #   define LIBRAPID_STATUS(msg, ...)                                                               \
146         do {                                                                                        \
147         } while (0)
148 #   define LIBRAPID_WARN(msg, ...)                                                                 \
149         do {                                                                                        \
150         } while (0)
151 #   define LIBRAPID_ERROR(msg, ...)                                                                \
152         do {                                                                                        \
153         } while (0)
154 #   define LIBRAPID_LOG(msg, ...)                                                                  \
155         do {                                                                                        \
156         } while (0)
157 #   define LIBRAPID_WASSERT(cond, ...)                                                             \
158         do {                                                                                        \
159         } while (0)
160 #   define LIBRAPID_ASSERT(cond, ...)                                                              \
161         do {                                                                                        \
162         } while (0)
163 #endif // LIBRAPID_ENABLE_ASSERT
164
165 #endif // LIBRAPID_CORE_MSVC_CONFIG_HPP
```

7.164 preMain.hpp

```
1 #ifndef LIBRAPID_CORE_PREMAIN
2 #define LIBRAPID_CORE_PREMAIN
3
4 /*
5  * This file defines internal functions which must run *before* main() is called.
6  */
7
8 namespace librapid::detail {
9     class PreMain {
10     public:
11         PreMain() {
12 #if defined(LIBRAPID_WINDOWS)
13             // Force the terminal to accept ANSI characters
14             system(("chcp " + std::to_string(CP_UTF8)).c_str());
15 #endif // LIBRAPID_WINDOWS
16         }
17
18     private:
19     };
20
21     // These must be declared here for use in ASSERT functions
22     template<typename T>
23     T internalMax(T val) {
24         return val;
25     }
26
27     template<typename T, typename... Tn>
28     T internalMax(T val, Tn... vals) {
29         auto maxOther = internalMax(vals...);
30         return val < maxOther ? maxOther : val;
31     }
32
```

```
33     [[maybe_unused]] static PreMain preMain = PreMain();
34 } // namespace librapid::detail
35
36 #endif // LIBRAPID_CORE_PREMAIN
```

7.165 traits.hpp

```
1 #ifndef LIBRAPID_CORE_TRAITS_HPP
2 #define LIBRAPID_CORE_TRAITS_HPP
3
4 /*
5  * The TypeInfo struct provides compile-time about types (templated types, in particular).
6  * This allows for easier SFINAE implementations and simpler function dispatching.
7  * Furthermore, the TypeInfo struct defines some useful conversion functions to cast between
8  * types without raising compiler warnings, or worse, errors.
9  *
10  * A TypeInfo struct should be defined for every class defined by LibRapid.
11  */
12
13 namespace librapid::typetraits {
14     namespace detail {
15         /*
16          * Pretty string representations of data types at compile time. This is adapted from
17          *
     https://bitwizeshift.github.io/posts/2021/03/09/getting-an-unmangled-type-name-at-compile-time/
18          * and I have simply adapted it to work with LibRapid.
19          */
20
21         template<std::size_t... Idxs>
22         constexpr auto substringAsArray(std::string_view str, std::index_sequence<Idxs...>) {
23             return std::array {str[Idxs]...};
24         }
25
26         template<typename T>
27         constexpr auto typeNameArray() {
28 #if defined(__clang__)
29             constexpr auto prefix   = std::string_view {"[T = "};
30             constexpr auto suffix   = std::string_view {"]"};
31             constexpr auto function = std::string_view {__PRETTY_FUNCTION__};
32 #elif defined(__GNUC__)
33             constexpr auto prefix   = std::string_view {"with T = "};
34             constexpr auto suffix   = std::string_view {"]"};
35             constexpr auto function = std::string_view {__PRETTY_FUNCTION__};
36 #elif defined(_MSC_VER)
37             constexpr auto prefix   = std::string_view {"type_name_array<"};
38             constexpr auto suffix   = std::string_view {">(void)"};
39             constexpr auto function = std::string_view {__FUNCSIG__};
40 #else
41 #   define LIBRAPID_NO_TYPE_TO_STRING
42 #endif
43
44 #if !defined(LIBRAPID_NO_TYPE_TO_STRING)
45             constexpr auto start = function.find(prefix) + prefix.size();
46             constexpr auto end   = function.rfind(suffix);
47
48             static_assert(start < end);
49
50             constexpr auto name = function.substr(start, (end - start));
51             return substringAsArray(name, std::make_index_sequence<name.size()> {});
52 #else
53             return std::array<char, 0> {};
54 #endif
55         }
56
57         template<typename T>
58         struct TypeNameHolder {
59             static inline constexpr auto value = typeNameArray<T>();
60         };
61     } // namespace detail
62
63     template<typename T>
64     constexpr auto typeName() -> std::string_view {
65         constexpr auto &value = detail::TypeNameHolder<T>::value;
66         return std::string_view {value.data(), value.size()};
67     }
68
72     template<typename T>
73     struct TypeInfo {
74         static constexpr bool isLibRapidType     = false;
75         using Scalar                             = T;
76         using Packet                             = std::false_type;
77         static constexpr int64_t packetWidth     = 1;
78         static constexpr char name[]             = "[NO DEFINED TYPE]";
```

```
79          static constexpr bool supportsArithmetic = true;
80          static constexpr bool supportsLogical    = true;
81          static constexpr bool supportsBinary     = true;
82
83  #if defined(LIBRAPID_HAS_CUDA)
84          static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_64F;
85  #endif
86          static constexpr bool canAlign  = true;
87          static constexpr bool canMemcpy = true;
88      };
89
90      template<>
91      struct TypeInfo<bool> {
92          static constexpr bool isLibRapidType     = false;
93          using Scalar                             = bool;
94          using Packet                             = std::false_type;
95          static constexpr int64_t packetWidth     = 1;
96          static constexpr char name[]             = "char";
97          static constexpr bool supportsArithmetic = false;
98          static constexpr bool supportsLogical    = false;
99          static constexpr bool supportsBinary     = true;
100
101 #if defined(LIBRAPID_HAS_CUDA)
102         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_8I;
103 #endif
104
105         static constexpr bool canAlign  = true;
106         static constexpr bool canMemcpy = true;
107     };
108
109     template<>
110     struct TypeInfo<char> {
111         static constexpr bool isLibRapidType     = false;
112         using Scalar                             = char;
113         using Packet                             = std::false_type;
114         static constexpr int64_t packetWidth     = 1;
115         static constexpr char name[]             = "bool";
116         static constexpr bool supportsArithmetic = false;
117         static constexpr bool supportsLogical    = false;
118         static constexpr bool supportsBinary     = true;
119
120 #if defined(LIBRAPID_HAS_CUDA)
121         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_8I;
122 #endif
123
124         static constexpr bool canAlign  = true;
125         static constexpr bool canMemcpy = true;
126     };
127
128     template<>
129     struct TypeInfo<int8_t> {
130         static constexpr bool isLibRapidType     = false;
131         using Scalar                             = int8_t;
132         using Packet                             = Vc::Vector<int8_t>;
133         static constexpr int64_t packetWidth     = Packet::size();
134         static constexpr char name[]             = "int8_t";
135         static constexpr bool supportsArithmetic = true;
136         static constexpr bool supportsLogical    = true;
137         static constexpr bool supportsBinary     = true;
138
139 #if defined(LIBRAPID_HAS_CUDA)
140         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_8I;
141 #endif
142
143         static constexpr bool canAlign  = true;
144         static constexpr bool canMemcpy = true;
145     };
146
147     template<>
148     struct TypeInfo<uint8_t> {
149         static constexpr bool isLibRapidType     = false;
150         using Scalar                             = uint8_t;
151         using Packet                             = Vc::Vector<uint8_t>;
152         static constexpr int64_t packetWidth     = Packet::size();
153         static constexpr char name[]             = "uint8_t";
154         static constexpr bool supportsArithmetic = true;
155         static constexpr bool supportsLogical    = true;
156         static constexpr bool supportsBinary     = true;
157
158 #if defined(LIBRAPID_HAS_CUDA)
159         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_8U;
160 #endif
161
162         static constexpr bool canAlign  = true;
163         static constexpr bool canMemcpy = true;
164     };
165
```

```
166     template<>
167     struct TypeInfo<int16_t> {
168         static constexpr bool isLibRapidType     = false;
169         using Scalar                             = int16_t;
170         using Packet                             = Vc::Vector<int16_t>;
171         static constexpr int64_t packetWidth     = Packet::size();
172         static constexpr char name[]             = "int16_t";
173         static constexpr bool supportsArithmetic = true;
174         static constexpr bool supportsLogical    = true;
175         static constexpr bool supportsBinary     = true;
176 
177 #if defined(LIBRAPID_HAS_CUDA)
178         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_16I;
179 #endif
180 
181         static constexpr bool canAlign  = true;
182         static constexpr bool canMemcpy = true;
183     };
184 
185     template<>
186     struct TypeInfo<uint16_t> {
187         static constexpr bool isLibRapidType     = false;
188         using Scalar                             = uint16_t;
189         using Packet                             = Vc::Vector<uint16_t>;
190         static constexpr int64_t packetWidth     = Packet::size();
191         static constexpr char name[]             = "uint16_t";
192         static constexpr bool supportsArithmetic = true;
193         static constexpr bool supportsLogical    = true;
194         static constexpr bool supportsBinary     = true;
195 
196 #if defined(LIBRAPID_HAS_CUDA)
197         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_16U;
198 #endif
199 
200         static constexpr bool canAlign  = true;
201         static constexpr bool canMemcpy = true;
202     };
203 
204     template<>
205     struct TypeInfo<int32_t> {
206         static constexpr bool isLibRapidType     = false;
207         using Scalar                             = int32_t;
208         using Packet                             = Vc::Vector<int32_t>;
209         static constexpr int64_t packetWidth     = Packet::size();
210         static constexpr char name[]             = "int32_t";
211         static constexpr bool supportsArithmetic = true;
212         static constexpr bool supportsLogical    = true;
213         static constexpr bool supportsBinary     = true;
214 
215 #if defined(LIBRAPID_HAS_CUDA)
216         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_32I;
217 #endif
218 
219         static constexpr bool canAlign  = true;
220         static constexpr bool canMemcpy = true;
221     };
222 
223     template<>
224     struct TypeInfo<uint32_t> {
225         static constexpr bool isLibRapidType     = false;
226         using Scalar                             = uint32_t;
227         using Packet                             = Vc::Vector<uint32_t>;
228         static constexpr int64_t packetWidth     = Packet::size();
229         static constexpr char name[]             = "uint32_t";
230         static constexpr bool supportsArithmetic = true;
231         static constexpr bool supportsLogical    = true;
232         static constexpr bool supportsBinary     = true;
233 
234 #if defined(LIBRAPID_HAS_CUDA)
235         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_32U;
236 #endif
237 
238         static constexpr bool canAlign  = true;
239         static constexpr bool canMemcpy = true;
240     };
241 
242     template<>
243     struct TypeInfo<int64_t> {
244         static constexpr bool isLibRapidType     = false;
245         using Scalar                             = int64_t;
246         using Packet                             = Vc::Vector<int64_t>;
247         static constexpr int64_t packetWidth     = Packet::size();
248         static constexpr char name[]             = "int64_t";
249         static constexpr bool supportsArithmetic = true;
250         static constexpr bool supportsLogical    = true;
251         static constexpr bool supportsBinary     = true;
252 
```

```
253 #if defined(LIBRAPID_HAS_CUDA)
254         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_64I;
255 #endif
256
257         static constexpr bool canAlign  = true;
258         static constexpr bool canMemcpy = true;
259     };
260
261     template<>
262     struct TypeInfo<uint64_t> {
263         static constexpr bool isLibRapidType     = false;
264         using Scalar                             = uint64_t;
265         using Packet                             = Vc::Vector<uint64_t>;
266         static constexpr int64_t packetWidth     = Packet::size();
267         static constexpr char name[]             = "uint64_t";
268         static constexpr bool supportsArithmetic = true;
269         static constexpr bool supportsLogical    = true;
270         static constexpr bool supportsBinary     = true;
271
272 #if defined(LIBRAPID_HAS_CUDA)
273         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_64U;
274 #endif
275
276         static constexpr bool canAlign  = true;
277         static constexpr bool canMemcpy = true;
278     };
279
280     template<>
281     struct TypeInfo<float> {
282         static constexpr bool isLibRapidType     = false;
283         using Scalar                             = float;
284         using Packet                             = Vc::Vector<float>;
285         static constexpr int64_t packetWidth     = Packet::size();
286         static constexpr char name[]             = "float";
287         static constexpr bool supportsArithmetic = true;
288         static constexpr bool supportsLogical    = true;
289         static constexpr bool supportsBinary     = false;
290
291 #if defined(LIBRAPID_HAS_CUDA)
292         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_32F;
293 #endif
294
295         static constexpr bool canAlign  = true;
296         static constexpr bool canMemcpy = true;
297     };
298
299     template<>
300     struct TypeInfo<double> {
301         static constexpr bool isLibRapidType     = false;
302         using Scalar                             = double;
303         using Packet                             = Vc::Vector<double>;
304         static constexpr int64_t packetWidth     = Packet::size();
305         static constexpr char name[]             = "double";
306         static constexpr bool supportsArithmetic = true;
307         static constexpr bool supportsLogical    = true;
308         static constexpr bool supportsBinary     = false;
309
310 #if defined(LIBRAPID_HAS_CUDA)
311         static constexpr cudaDataType_t CudaType = cudaDataType_t::CUDA_R_64F;
312 #endif
313
314         static constexpr bool canAlign  = true;
315         static constexpr bool canMemcpy = true;
316     };
317 } // namespace librapid::typetraits
318
319 #endif // LIBRAPID_CORE_TRAITS_HPP
```

7.166 typetraits.hpp

```
1 #ifndef LIBRAPID_CORE_TYPETRAITS_HPP
2 #define LIBRAPID_CORE_TYPETRAITS_HPP
3
4 /*
5  * Defines a range of helper template types to increase code clarity
6  * while simultaneously reducing code verbosity.
7  */
8
9 namespace librapid::typetraits {
10     template<bool Cond, typename T = int>
11     using EnableIf = std::enable_if_t<Cond, T>;
12
13     template<typename A, typename B>
```

```
14      constexpr bool IsSame = std::is_same<A, B>::value;
15
16      namespace impl {
17          /*
18           * These functions test for the presence of certain features of a type
19           * by providing two valid function overloads, but the preferred one
20           * (the one taking an integer) is only valid if the requested feature
21           * exists. The return type of both functions differ, and can be evaluated
22           * as "true" and "false" depending on the presence of the feature.
23           *
24           * This is really cool :)
25           */
26
27          template<typename T, typename Index,
28                   typename = decltype(std::declval<T &>()[std::declval<Index>()])>
29          std::true_type testSubscript(int);
30          template<typename T, typename Index>
31          std::false_type testSubscript(float);
32
33          template<typename T, typename V,
34                   typename = decltype(std::declval<T &>() + std::declval<V &>())>
35          std::true_type testAddition(int);
36          template<typename T, typename V>
37          std::false_type testAddition(float);
38
39          template<typename T, typename V,
40                   typename = decltype(std::declval<T &>() * std::declval<V &>())>
41          std::true_type testMultiplication(int);
42          template<typename T, typename V>
43          std::false_type testMultiplication(float);
44
45          template<typename From, typename To, typename = decltype((From)std::declval<From &>())>
46          std::true_type testCast(int);
47          template<typename From, typename To>
48          std::false_type testCast(float);
49      } // namespace impl
50
51      template<typename T, typename Index = int64_t>
52      struct HasSubscript : public decltype(impl::testSubscript<T, Index>(1)) {};
53
54      template<typename T, typename V = T>
55      struct HasAddition : public decltype(impl::testAddition<T, V>(1)) {};
56
57      template<typename T, typename V = T>
58      struct HasMultiplication : public decltype(impl::testMultiplication<T, V>(1)) {};
59
60      template<typename From, typename To>
61      struct CanCast : public decltype(impl::testCast<From, To>(1)) {};
62
63      // Detect whether a class can be default constructed
64      template<class T>
65      using TriviallyDefaultConstructible = std::is_trivially_default_constructible<T>;
66  } // namespace librapid::typetraits
67
68  #endif // LIBRAPID_CORE_TYPETRAITS_HPP
```

7.167 warningSuppress.hpp

```
1  #ifndef LIBRAPID_WARNING_SUPPRESS
2  #define LIBRAPID_WARNING_SUPPRESS
3
4  #ifdef _MSC_VER
5  #   define LIBRAPID_MSVC_SUPPRESS(WARNING_) __pragma(warning(suppress : WARNING_))
6  #else
7  #   define LIBRAPID_MSVC_SUPPRESS(WARNING_)
8  #endif
9
10 // Disable warnings for GCC/Clang
11 #ifdef __GNUC__
12 #   define LIBRAPID_GCC_SUPPRESS(WARNING_)                                                \
13         _Pragma("GCC diagnostic push") _Pragma("GCC diagnostic ignored \"-W" #WARNING_ "\"")
14 #else
15 #   define LIBRAPID_GCC_SUPPRESS(WARNING_)
16 #endif
17
18 LIBRAPID_MSVC_SUPPRESS(4996) // Disable warnings about unsafe classes
19 LIBRAPID_MSVC_SUPPRESS(4723) // Disable zero division errors
20 LIBRAPID_MSVC_SUPPRESS(5245) // unreferenced function with internal linkage has been removed
21
22 #endif // LIBRAPID_WARNING_SUPPRESS
```

7.168 exception.h

```
1 /* Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
2  *
3  * Redistribution and use in source and binary forms, with or without
4  * modification, are permitted provided that the following conditions
5  * are met:
6  *  * Redistributions of source code must retain the above copyright
7  *    notice, this list of conditions and the following disclaimer.
8  *  * Redistributions in binary form must reproduce the above copyright
9  *    notice, this list of conditions and the following disclaimer in the
10 *    documentation and/or other materials provided with the distribution.
11 *  * Neither the name of NVIDIA CORPORATION nor the names of its
12 *    contributors may be used to endorse or promote products derived
13 *    from this software without specific prior written permission.
14 *
15 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS" AND ANY
16 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
17 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
18 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
19 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
20 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
21 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
22 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
23 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
24 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
25 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26 */
27
28 /* CUda UTility Library */
29 #ifndef COMMON_EXCEPTION_H_
30 #define COMMON_EXCEPTION_H_
31
32 // includes, system
33 #include <exception>
34 #include <iostream>
35 #include <stdexcept>
36 #include <stdlib.h>
37 #include <string>
38
41 template<class Std_Exception>
42 class Exception : public Std_Exception {
43 public:
49     static void throw_it(const char *file, const int line, const char *detailed = "-");
50
56     static void throw_it(const char *file, const int line, const std::string &detailed);
57
59     virtual ~Exception() throw();
60
61 private:
63     Exception();
64
67     explicit Exception(const std::string &str);
68 };
69
74 template<class Exception_Typ>
75 inline void handleException(const Exception_Typ &ex) {
76     std::cerr « ex.what() « std::endl;
77
78     exit(EXIT_FAILURE);
79 }
80
82
84 #define RUNTIME_EXCEPTION(msg) Exception<std::runtime_error>::throw_it(__FILE__, __LINE__, msg)
85
87 #define LOGIC_EXCEPTION(msg) Exception<std::logic_error>::throw_it(__FILE__, __LINE__, msg)
88
90 #define RANGE_EXCEPTION(msg) Exception<std::range_error>::throw_it(__FILE__, __LINE__, msg)
91
94
95 // includes, system
96 #include <sstream>
97
102 /*static*/ template<class Std_Exception>
103 void Exception<Std_Exception>::throw_it(const char *file, const int line, const char *detailed) {
104     std::stringstream s;
105
106     // Quiet heavy-weight but exceptions are not for
107     // performance / release versions
108     s « "Exception in file '" « file « "' in line " « line « "\n"
109       « "Detailed description: " « detailed « "\n";
110
111     throw Exception(s.str());
112 }
113
118 /*static*/ template<class Std_Exception>
119 void Exception<Std_Exception>::throw_it(const char *file, const int line, const std::string &msg) {
```

```
120     throw_it(file, line, msg.c_str());
121 }
122
126 template<class Std_Exception>
127 Exception<Std_Exception>::Exception() : Std_Exception("Unknown Exception.\n") {}
128
133 template<class Std_Exception>
134 Exception<Std_Exception>::Exception(const std::string &s) : Std_Exception(s) {}
135
139 template<class Std_Exception>
140 Exception<Std_Exception>::~Exception() throw() {}
141
142 // functions, exported
143
144 #endif // COMMON_EXCEPTION_H_
```

7.169 helper_cuda.h

```
1 /* Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
2  *
3  * Redistribution and use in source and binary forms, with or without
4  * modification, are permitted provided that the following conditions
5  * are met:
6  *  * Redistributions of source code must retain the above copyright
7  *    notice, this list of conditions and the following disclaimer.
8  *  * Redistributions in binary form must reproduce the above copyright
9  *    notice, this list of conditions and the following disclaimer in the
10 *    documentation and/or other materials provided with the distribution.
11 *  * Neither the name of NVIDIA CORPORATION nor the names of its
12 *    contributors may be used to endorse or promote products derived
13 *    from this software without specific prior written permission.
14 *
15 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS" AND ANY
16 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
17 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
18 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
19 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
20 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
21 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
22 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
23 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
24 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
25 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26 */
27
29 // These are CUDA Helper functions for initialization and error checking
30
31 #ifndef COMMON_HELPER_CUDA_H_
32 #define COMMON_HELPER_CUDA_H_
33
34 #pragma once
35
36 #include "helper_string.h"
37 #include <stdint.h>
38 #include <stdio.h>
39 #include <stdlib.h>
40 #include <string.h>
41
42 #ifndef EXIT_WAIVED
43 #   define EXIT_WAIVED 2
44 #endif
45
46 // Note, it is required that your SDK sample to include the proper header
47 // files, please refer the CUDA examples for examples of the needed CUDA
48 // headers, which may change depending on which CUDA functions are used.
49
50 // CUDA Runtime error messages
51 #ifdef __DRIVER_TYPES_H__
52 const char *_cudaGetErrorEnum(cudaError_t error);
53 #endif
54
55 #ifdef CUDA_DRIVER_API
56 // CUDA Driver API errors
57 const char *_cudaGetErrorEnum(CUresult error);
58 #endif
59
60 #ifdef CUBLAS_API_H_
61 // cuBLAS API errors
62 const char *_cudaGetErrorEnum(cublasStatus_t error);
63 #endif
64
65 #ifdef _CUFFT_H_
66 // cuFFT API errors
```

```
67 static const char *_cudaGetErrorEnum(cufftResult error);
68 #endif
69
70 #ifdef CUSPARSEAPI
71 // cuSPARSE API errors
72 const char *_cudaGetErrorEnum(cusparseStatus_t error);
73 #endif
74
75 #ifdef CUSOLVER_COMMON_H_
76 // cuSOLVER API errors
77 const char *_cudaGetErrorEnum(cusolverStatus_t error);
78 #endif
79
80 #ifdef CURAND_H_
81 // cuRAND API errors
82 const char *_cudaGetErrorEnum(curandStatus_t error);
83 #endif
84
85 #ifdef NVJPEGAPI
86 // nvJPEG API errors
87 const char *_cudaGetErrorEnum(nvjpegStatus_t error);
88 #endif
89
90 #ifdef NV_NPPIDEFS_H
91 // NPP API errors
92 const char *_cudaGetErrorEnum(NppStatus error);
93 #endif
94
95 template<typename T>
96 void check(T result, char const *const func, const char *const file, int const line) {
97     if (result) {
98         fprintf(stderr,
99                 "CUDA error at %s:%d code=%d(%s) \"%s\" \n",
100                 file,
101                 line,
102                 static_cast<unsigned int>(result),
103                 _cudaGetErrorEnum(result),
104                 func);
105         exit(EXIT_FAILURE);
106     }
107 }
108
109 #ifdef __DRIVER_TYPES_H__
110 // This will output the proper CUDA error strings in the event
111 // that a CUDA host call returns an error
112 #   define checkCudaErrors(val) check((val), #    val, __FILE__, __LINE__)
113
114 // This will output the proper error string when calling cudaGetLastError
115 #   define getLastCudaError(msg) __getLastCudaError(msg, __FILE__, __LINE__)
116
117 inline void __getLastCudaError(const char *errorMessage, const char *file, const int line) {
118     cudaError_t err = cudaGetLastError();
119
120     if (cudaSuccess != err) {
121         fprintf(stderr,
122                 "%s(%i) : getLastCudaError() CUDA error :"
123                 " %s : (%d) %s.\n",
124                 file,
125                 line,
126                 errorMessage,
127                 static_cast<int>(err),
128                 cudaGetErrorString(err));
129         exit(EXIT_FAILURE);
130     }
131 }
132
133 // This will only print the proper error string when calling cudaGetLastError
134 // but not exit program incase error detected.
135 #   define printLastCudaError(msg) __printLastCudaError(msg, __FILE__, __LINE__)
136
137 inline void __printLastCudaError(const char *errorMessage, const char *file, const int line) {
138     cudaError_t err = cudaGetLastError();
139
140     if (cudaSuccess != err) {
141         fprintf(stderr,
142                 "%s(%i) : getLastCudaError() CUDA error :"
143                 " %s : (%d) %s.\n",
144                 file,
145                 line,
146                 errorMessage,
147                 static_cast<int>(err),
148                 cudaGetErrorString(err));
149     }
150 }
151
152 #endif
153
```

```
154 #ifndef MAX
155 #   define MAX(a, b) (a > b ? a : b)
156 #endif
157
158 // Float To Int conversion
159 inline int ftoi(float value) {
160     return (value >= 0 ? static_cast<int>(value + 0.5) : static_cast<int>(value - 0.5));
161 }
162
163 // Beginning of GPU Architecture definitions
164 inline int _ConvertSMVer2Cores(int major, int minor) {
165     // Defines for GPU Architecture types (using the SM version to determine
166     // the # of cores per SM
167     typedef struct {
168         int SM; // 0xMm (hexidecimal notation), M = SM Major version,
169         // and m = SM minor version
170         int Cores;
171     } sSMtoCores;
172
173     sSMtoCores nGpuArchCoresPerSM[] = {{0x30, 192},
174                                        {0x32, 192},
175                                        {0x35, 192},
176                                        {0x37, 192},
177                                        {0x50, 128},
178                                        {0x52, 128},
179                                        {0x53, 128},
180                                        {0x60, 64},
181                                        {0x61, 128},
182                                        {0x62, 128},
183                                        {0x70, 64},
184                                        {0x72, 64},
185                                        {0x75, 64},
186                                        {0x80, 64},
187                                        {0x86, 128},
188                                        {-1, -1}};
189
190     int index = 0;
191
192     while (nGpuArchCoresPerSM[index].SM != -1) {
193         if (nGpuArchCoresPerSM[index].SM == ((major « 4) + minor)) {
194             return nGpuArchCoresPerSM[index].Cores;
195         }
196
197         index++;
198     }
199
200     // If we don't find the values, we default use the previous one
201     // to run properly
202     printf(
203       "MapSMtoCores for SM %d.%d is undefined."
204       "  Default to use %d Cores/SM\n",
205       major,
206       minor,
207       nGpuArchCoresPerSM[index - 1].Cores);
208     return nGpuArchCoresPerSM[index - 1].Cores;
209 }
210
211 inline const char *_ConvertSMVer2ArchName(int major, int minor) {
212     // Defines for GPU Architecture types (using the SM version to determine
213     // the GPU Arch name)
214     typedef struct {
215         int SM; // 0xMm (hexidecimal notation), M = SM Major version,
216         // and m = SM minor version
217         const char *name;
218     } sSMtoArchName;
219
220     sSMtoArchName nGpuArchNameSM[] = {{0x30, "Kepler"},
221                                       {0x32, "Kepler"},
222                                       {0x35, "Kepler"},
223                                       {0x37, "Kepler"},
224                                       {0x50, "Maxwell"},
225                                       {0x52, "Maxwell"},
226                                       {0x53, "Maxwell"},
227                                       {0x60, "Pascal"},
228                                       {0x61, "Pascal"},
229                                       {0x62, "Pascal"},
230                                       {0x70, "Volta"},
231                                       {0x72, "Xavier"},
232                                       {0x75, "Turing"},
233                                       {0x80, "Ampere"},
234                                       {0x86, "Ampere"},
235                                       {-1, "Graphics Device"}};
236
237     int index = 0;
238
239     while (nGpuArchNameSM[index].SM != -1) {
240         if (nGpuArchNameSM[index].SM == ((major « 4) + minor)) {
```

```
241             return nGpuArchNameSM[index].name;
242         }
243 
244         index++;
245     }
246 
247     // If we don't find the values, we default use the previous one
248     // to run properly
249     printf(
250       "MapSMtoArchName for SM %d.%d is undefined."
251       "  Default to use %s\n",
252       major,
253       minor,
254       nGpuArchNameSM[index - 1].name);
255     return nGpuArchNameSM[index - 1].name;
256 }
257 // end of GPU Architecture definitions
258 
259 #ifdef __CUDA_RUNTIME_H__
260 
261 // General GPU Device CUDA Initialization
262 inline int gpuDeviceInit(int devID) {
263     int device_count;
264     checkCudaErrors(cudaGetDeviceCount(&device_count));
265 
266     if (device_count == 0) {
267         fprintf(stderr,
268                 "gpuDeviceInit() CUDA error: "
269                 "no devices supporting CUDA.\n");
270         exit(EXIT_FAILURE);
271     }
272 
273     if (devID < 0) { devID = 0; }
274 
275     if (devID > device_count - 1) {
276         fprintf(stderr, "\n");
277         fprintf(stderr, "» %d CUDA capable GPU device(s) detected. «\n", device_count);
278         fprintf(stderr,
279                 "» gpuDeviceInit (-device=%d) is not a valid"
280                 " GPU device. «\n",
281                 devID);
282         fprintf(stderr, "\n");
283         return -devID;
284     }
285 
286     int computeMode = -1, major = 0, minor = 0;
287     checkCudaErrors(cudaDeviceGetAttribute(&computeMode, cudaDevAttrComputeMode, devID));
288     checkCudaErrors(cudaDeviceGetAttribute(&major, cudaDevAttrComputeCapabilityMajor, devID));
289     checkCudaErrors(cudaDeviceGetAttribute(&minor, cudaDevAttrComputeCapabilityMinor, devID));
290     if (computeMode == cudaComputeModeProhibited) {
291         fprintf(stderr,
292                 "Error: device is running in <Compute Mode "
293                 "Prohibited>, no threads can use cudaSetDevice().\n");
294         return -1;
295     }
296 
297     if (major < 1) {
298         fprintf(stderr, "gpuDeviceInit(): GPU device does not support CUDA.\n");
299         exit(EXIT_FAILURE);
300     }
301 
302     checkCudaErrors(cudaSetDevice(devID));
303     printf("gpuDeviceInit() CUDA Device [%d]: \"%s\n", devID, _ConvertSMVer2ArchName(major, minor));
304 
305     return devID;
306 }
307 
308 // This function returns the best GPU (with maximum GFLOPS)
309 inline int gpuGetMaxGflopsDeviceId() {
310     int current_device = 0, sm_per_multiproc = 0;
311     int max_perf_device    = 0;
312     int device_count       = 0;
313     int devices_prohibited = 0;
314 
315     uint64_t max_compute_perf = 0;
316     checkCudaErrors(cudaGetDeviceCount(&device_count));
317 
318     if (device_count == 0) {
319         fprintf(stderr,
320                 "gpuGetMaxGflopsDeviceId() CUDA error:"
321                 " no devices supporting CUDA.\n");
322         exit(EXIT_FAILURE);
323     }
324 
325     // Find the best CUDA capable GPU device
326     current_device = 0;
327 
```

```
328     while (current_device < device_count) {
329         int computeMode = -1, major = 0, minor = 0;
330         checkCudaErrors(
331           cudaDeviceGetAttribute(&computeMode, cudaDevAttrComputeMode, current_device));
332         checkCudaErrors(
333           cudaDeviceGetAttribute(&major, cudaDevAttrComputeCapabilityMajor, current_device));
334         checkCudaErrors(
335           cudaDeviceGetAttribute(&minor, cudaDevAttrComputeCapabilityMinor, current_device));
336 
337         // If this GPU is not running on Compute Mode prohibited,
338         // then we can add it to the list
339         if (computeMode != cudaComputeModeProhibited) {
340             if (major == 9999 && minor == 9999) {
341                 sm_per_multiproc = 1;
342             } else {
343                 sm_per_multiproc = _ConvertSMVer2Cores(major, minor);
344             }
345             int multiProcessorCount = 0, clockRate = 0;
346             checkCudaErrors(cudaDeviceGetAttribute(
347               &multiProcessorCount, cudaDevAttrMultiProcessorCount, current_device));
348             cudaError_t result =
349               cudaDeviceGetAttribute(&clockRate, cudaDevAttrClockRate, current_device);
350             if (result != cudaSuccess) {
351                 // If cudaDevAttrClockRate attribute is not supported we
352                 // set clockRate as 1, to consider GPU with most SMs and CUDA
353                 // Cores.
354                 if (result == cudaErrorInvalidValue) {
355                     clockRate = 1;
356                 } else {
357                     fprintf(stderr,
358                             "CUDA error at %s:%d code=%d(%s) \n",
359                             __FILE__,
360                             __LINE__,
361                             static_cast<unsigned int>(result),
362                             _cudaGetErrorEnum(result));
363                     exit(EXIT_FAILURE);
364                 }
365             }
366             uint64_t compute_perf = (uint64_t)multiProcessorCount * sm_per_multiproc * clockRate;
367 
368             if (compute_perf > max_compute_perf) {
369                 max_compute_perf = compute_perf;
370                 max_perf_device  = current_device;
371             }
372         } else {
373             devices_prohibited++;
374         }
375 
376         ++current_device;
377     }
378 
379     if (devices_prohibited == device_count) {
380         fprintf(stderr,
381                 "gpuGetMaxGflopsDeviceId() CUDA error:"
382                 " all devices have compute mode prohibited.\n");
383         exit(EXIT_FAILURE);
384     }
385 
386     return max_perf_device;
387 }
388 
389 // Initialization code to find the best CUDA Device
390 inline int findCudaDevice(int argc, const char **argv) {
391     int devID = 0;
392 
393     // If the command-line has a device number specified, use it
394     if (checkCmdLineFlag(argc, argv, "device")) {
395         devID = getCmdLineArgumentInt(argc, argv, "device=");
396 
397         if (devID < 0) {
398             printf("Invalid command line parameter\n ");
399             exit(EXIT_FAILURE);
400         } else {
401             devID = gpuDeviceInit(devID);
402 
403             if (devID < 0) {
404                 printf("exiting...\n");
405                 exit(EXIT_FAILURE);
406             }
407         }
408     } else {
409         // Otherwise pick the device with highest Gflops/s
410         devID = gpuGetMaxGflopsDeviceId();
411         checkCudaErrors(cudaSetDevice(devID));
412         int major = 0, minor = 0;
413         checkCudaErrors(cudaDeviceGetAttribute(&major, cudaDevAttrComputeCapabilityMajor, devID));
414         checkCudaErrors(cudaDeviceGetAttribute(&minor, cudaDevAttrComputeCapabilityMinor, devID));
```

```
415         printf("GPU Device %d: \"%s\" with compute capability %d.%d\n\n",
416                devID,
417                _ConvertSMVer2ArchName(major, minor),
418                major,
419                minor);
420     }
421
422     return devID;
423 }
424
425 inline int findIntegratedGPU() {
426     int current_device     = 0;
427     int device_count       = 0;
428     int devices_prohibited = 0;
429
430     checkCudaErrors(cudaGetDeviceCount(&device_count));
431
432     if (device_count == 0) {
433         fprintf(stderr, "CUDA error: no devices supporting CUDA.\n");
434         exit(EXIT_FAILURE);
435     }
436
437     // Find the integrated GPU which is compute capable
438     while (current_device < device_count) {
439         int computeMode = -1, integrated = -1;
440         checkCudaErrors(
441           cudaDeviceGetAttribute(&computeMode, cudaDevAttrComputeMode, current_device));
442         checkCudaErrors(cudaDeviceGetAttribute(&integrated, cudaDevAttrIntegrated, current_device));
443         // If GPU is integrated and is not running on Compute Mode prohibited,
444         // then cuda can map to GLES resource
445         if (integrated && (computeMode != cudaComputeModeProhibited)) {
446             checkCudaErrors(cudaSetDevice(current_device));
447
448             int major = 0, minor = 0;
449             checkCudaErrors(
450               cudaDeviceGetAttribute(&major, cudaDevAttrComputeCapabilityMajor, current_device));
451             checkCudaErrors(
452               cudaDeviceGetAttribute(&minor, cudaDevAttrComputeCapabilityMinor, current_device));
453             printf("GPU Device %d: \"%s\" with compute capability %d.%d\n\n",
454                    current_device,
455                    _ConvertSMVer2ArchName(major, minor),
456                    major,
457                    minor);
458
459             return current_device;
460         } else {
461             devices_prohibited++;
462         }
463
464         current_device++;
465     }
466
467     if (devices_prohibited == device_count) {
468         fprintf(stderr,
469                 "CUDA error:"
470                 " No GLES-CUDA Interop capable GPU found.\n");
471         exit(EXIT_FAILURE);
472     }
473
474     return -1;
475 }
476
477 // General check for CUDA GPU SM Capabilities
478 inline bool checkCudaCapabilities(int major_version, int minor_version) {
479     int dev;
480     int major = 0, minor = 0;
481
482     checkCudaErrors(cudaGetDevice(&dev));
483     checkCudaErrors(cudaDeviceGetAttribute(&major, cudaDevAttrComputeCapabilityMajor, dev));
484     checkCudaErrors(cudaDeviceGetAttribute(&minor, cudaDevAttrComputeCapabilityMinor, dev));
485
486     if ((major > major_version) || (major == major_version && minor >= minor_version)) {
487         printf("  Device %d: <%16s >, Compute SM %d.%d detected\n",
488                dev,
489                _ConvertSMVer2ArchName(major, minor),
490                major,
491                minor);
492         return true;
493     } else {
494         printf(
495           "  No GPU device was found that can support "
496           "CUDA compute capability %d.%d.\n",
497           major_version,
498           minor_version);
499         return false;
500     }
501 }
```

```
502
503 #endif
504
505 // end of CUDA Helper Functions
506
507 #endif // COMMON_HELPER_CUDA_H_
```

7.170 helper_cuda_drvapi.h

```
1 /* Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
2  *
3  * Redistribution and use in source and binary forms, with or without
4  * modification, are permitted provided that the following conditions
5  * are met:
6  *  * Redistributions of source code must retain the above copyright
7  *    notice, this list of conditions and the following disclaimer.
8  *  * Redistributions in binary form must reproduce the above copyright
9  *    notice, this list of conditions and the following disclaimer in the
10 *    documentation and/or other materials provided with the distribution.
11 *  * Neither the name of NVIDIA CORPORATION nor the names of its
12 *    contributors may be used to endorse or promote products derived
13 *    from this software without specific prior written permission.
14 *
15 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS" AND ANY
16 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
17 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
18 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
19 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
20 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
21 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
22 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
23 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
24 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
25 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26 */
27
28 // Helper functions for CUDA Driver API error handling (make sure that CUDA_H is
29 // included in your projects)
30 #ifndef COMMON_HELPER_CUDA_DRVAPI_H_
31 #define COMMON_HELPER_CUDA_DRVAPI_H_
32
33 #include <cstring>
34 #include "helper_string.h"
35 #include <iostream>
36 #include <sstream>
37 #include <stdio.h>
38 #include <stdlib.h>
39 #include <string.h>
40
41 #ifndef MAX
42 #   define MAX(a, b) (a > b ? a : b)
43 #endif
44
45 #ifndef COMMON_HELPER_CUDA_H_
46
47 inline int ftoi(float value) {
48     return (value >= 0 ? static_cast<int>(value + 0.5) : static_cast<int>(value - 0.5));
49 }
50
51 #endif
52
53 #ifndef EXIT_WAIVED
54 #   define EXIT_WAIVED 2
55 #endif
56
57 // These are CUDA Helper functions
58
59 // add a level of protection to the CUDA SDK samples, let's force samples to
60 // explicitly include CUDA.H
61 #ifdef __cuda_cuda_h__
62 // This will output the proper CUDA error strings in the event that a CUDA host
63 // call returns an error
64 #   ifndef checkCudaErrors
65 #       define checkCudaErrors(err) __checkCudaErrors(err, __FILE__, __LINE__)
66
67 // These are the inline versions for all of the SDK helper functions
68 inline void __checkCudaErrors(CUresult err, const char *file, const int line) {
69     if (CUDA_SUCCESS != err) {
70         const char *errorStr = NULL;
71         cuGetErrorString(err, &errorStr);
72         fprintf(stderr,
73                 "checkCudaErrors() Driver API error = %04d \"%s\" from file <%s>, "
74                 "line %i.\n",
75
```

```
76                 err,
77                 errorStr,
78                 file,
79                 line);
80         exit(EXIT_FAILURE);
81     }
82 }
83 #   endif
84
85 // This function wraps the CUDA Driver API into a template function
86 template<class T>
87 inline void getCudaAttribute(T *attribute, CUdevice_attribute device_attribute, int device) {
88     checkCudaErrors(cuDeviceGetAttribute(attribute, device_attribute, device));
89 }
90 #endif
91
92 // Beginning of GPU Architecture definitions
93 inline int _ConvertSMVer2CoresDRV(int major, int minor) {
94     // Defines for GPU Architecture types (using the SM version to determine the
95     // # of cores per SM
96     typedef struct {
97         int SM; // 0xMm (hexidecimal notation), M = SM Major version, and m = SM
98         // minor version
99         int Cores;
100     } sSMtoCores;
101
102     sSMtoCores nGpuArchCoresPerSM[] = {{0x30, 192},
103                                        {0x32, 192},
104                                        {0x35, 192},
105                                        {0x37, 192},
106                                        {0x50, 128},
107                                        {0x52, 128},
108                                        {0x53, 128},
109                                        {0x60, 64},
110                                        {0x61, 128},
111                                        {0x62, 128},
112                                        {0x70, 64},
113                                        {0x72, 64},
114                                        {0x75, 64},
115                                        {0x80, 64},
116                                        {0x86, 128},
117                                        {-1, -1}};
118
119     int index = 0;
120
121     while (nGpuArchCoresPerSM[index].SM != -1) {
122         if (nGpuArchCoresPerSM[index].SM == ((major « 4) + minor)) {
123             return nGpuArchCoresPerSM[index].Cores;
124         }
125
126         index++;
127     }
128
129     // If we don’t find the values, we default use the previous one to run
130     // properly
131     printf("MapSMtoCores for SM %d.%d is undefined.  Default to use %d Cores/SM\n",
132            major,
133            minor,
134            nGpuArchCoresPerSM[index - 1].Cores);
135     return nGpuArchCoresPerSM[index - 1].Cores;
136 }
137 // end of GPU Architecture definitions
138
139 #ifdef __cuda_cuda_h__
140 // General GPU Device CUDA Initialization
141 inline int gpuDeviceInitDRV(int ARGC, const char **ARGV) {
142     int cuDevice    = 0;
143     int deviceCount = 0;
144     checkCudaErrors(cuInit(0));
145
146     checkCudaErrors(cuDeviceGetCount(&deviceCount));
147
148     if (deviceCount == 0) {
149         fprintf(stderr, "cudaDeviceInit error: no devices supporting CUDA\n");
150         exit(EXIT_FAILURE);
151     }
152
153     int dev = 0;
154     dev     = getCmdLineArgumentInt(ARGC, (const char **)ARGV, "device=");
155
156     if (dev < 0) { dev = 0; }
157
158     if (dev > deviceCount - 1) {
159         fprintf(stderr, "\n");
160         fprintf(stderr, "» %d CUDA capable GPU device(s) detected. «\n", deviceCount);
161         fprintf(stderr, "» cudaDeviceInit (-device=%d) is not a valid GPU device. «\n", dev);
162         fprintf(stderr, "\n");
```

```
163         return -dev;
164     }
165
166     checkCudaErrors(cuDeviceGet(&cuDevice, dev));
167     char name[100];
168     checkCudaErrors(cuDeviceGetName(name, 100, cuDevice));
169
170     int computeMode;
171     getCudaAttribute<int>(&computeMode, CU_DEVICE_ATTRIBUTE_COMPUTE_MODE, dev);
172
173     if (computeMode == CU_COMPUTEMODE_PROHIBITED) {
174         fprintf(stderr,
175                 "Error: device is running in <CU_COMPUTEMODE_PROHIBITED>, no "
176                 "threads can use this CUDA Device.\n");
177         return -1;
178     }
179
180     if (checkCmdLineFlag(ARGC, (const char **)ARGV, "quiet") == false) {
181         printf("gpuDeviceInitDRV() Using CUDA Device [%d]: %s\n", dev, name);
182     }
183
184     return dev;
185 }
186
187 // This function returns the best GPU based on performance
188 inline int gpuGetMaxGflopsDeviceIdDRV() {
189     CUdevice current_device             = 0;
190     CUdevice max_perf_device            = 0;
191     int device_count                    = 0;
192     int sm_per_multiproc                = 0;
193     unsigned long long max_compute_perf = 0;
194     int major                           = 0;
195     int minor                           = 0;
196     int multiProcessorCount;
197     int clockRate;
198     int devices_prohibited = 0;
199
200     cuInit(0);
201     checkCudaErrors(cuDeviceGetCount(&device_count));
202
203     if (device_count == 0) {
204         fprintf(stderr, "gpuGetMaxGflopsDeviceIdDRV error: no devices supporting CUDA\n");
205         exit(EXIT_FAILURE);
206     }
207
208     // Find the best CUDA capable GPU device
209     current_device = 0;
210
211     while (current_device < device_count) {
212         checkCudaErrors(cuDeviceGetAttribute(
213           &multiProcessorCount, CU_DEVICE_ATTRIBUTE_MULTIPROCESSOR_COUNT, current_device));
214         checkCudaErrors(
215           cuDeviceGetAttribute(&clockRate, CU_DEVICE_ATTRIBUTE_CLOCK_RATE, current_device));
216         checkCudaErrors(cuDeviceGetAttribute(
217           &major, CU_DEVICE_ATTRIBUTE_COMPUTE_CAPABILITY_MAJOR, current_device));
218         checkCudaErrors(cuDeviceGetAttribute(
219           &minor, CU_DEVICE_ATTRIBUTE_COMPUTE_CAPABILITY_MINOR, current_device));
220
221         int computeMode;
222         getCudaAttribute<int>(&computeMode, CU_DEVICE_ATTRIBUTE_COMPUTE_MODE, current_device);
223
224         if (computeMode != CU_COMPUTEMODE_PROHIBITED) {
225             if (major == 9999 && minor == 9999) {
226                 sm_per_multiproc = 1;
227             } else {
228                 sm_per_multiproc = _ConvertSMVer2CoresDRV(major, minor);
229             }
230
231             unsigned long long compute_perf =
232               (unsigned long long)(multiProcessorCount * sm_per_multiproc * clockRate);
233
234             if (compute_perf > max_compute_perf) {
235                 max_compute_perf = compute_perf;
236                 max_perf_device  = current_device;
237             }
238         } else {
239             devices_prohibited++;
240         }
241
242         ++current_device;
243     }
244
245     if (devices_prohibited == device_count) {
246         fprintf(stderr,
247                 "gpuGetMaxGflopsDeviceIdDRV error: all devices have compute mode "
248                 "prohibited.\n");
249         exit(EXIT_FAILURE);
```

```
250     }
251
252     return max_perf_device;
253 }
254
255 // General initialization call to pick the best CUDA Device
256 inline CUdevice findCudaDeviceDRV(int argc, const char **argv) {
257     CUdevice cuDevice;
258     int devID = 0;
259
260     // If the command-line has a device number specified, use it
261     if (checkCmdLineFlag(argc, (const char **)argv, "device")) {
262         devID = gpuDeviceInitDRV(argc, argv);
263
264         if (devID < 0) {
265             printf("exiting...\n");
266             exit(EXIT_SUCCESS);
267         }
268     } else {
269         // Otherwise pick the device with highest Gflops/s
270         char name[100];
271         devID = gpuGetMaxGflopsDeviceIdDRV();
272         checkCudaErrors(cuDeviceGet(&cuDevice, devID));
273         cuDeviceGetName(name, 100, cuDevice);
274         printf("> Using CUDA Device [%d]: %s\n", devID, name);
275     }
276
277     cuDeviceGet(&cuDevice, devID);
278
279     return cuDevice;
280 }
281
282 inline CUdevice findIntegratedGPUDrv() {
283     CUdevice current_device = 0;
284     int device_count        = 0;
285     int devices_prohibited  = 0;
286     int isIntegrated;
287
288     cuInit(0);
289     checkCudaErrors(cuDeviceGetCount(&device_count));
290
291     if (device_count == 0) {
292         fprintf(stderr, "CUDA error: no devices supporting CUDA.\n");
293         exit(EXIT_FAILURE);
294     }
295
296     // Find the integrated GPU which is compute capable
297     while (current_device < device_count) {
298         int computeMode = -1;
299         checkCudaErrors(
300           cuDeviceGetAttribute(&isIntegrated, CU_DEVICE_ATTRIBUTE_INTEGRATED, current_device));
301         checkCudaErrors(
302           cuDeviceGetAttribute(&computeMode, CU_DEVICE_ATTRIBUTE_COMPUTE_MODE, current_device));
303
304         // If GPU is integrated and is not running on Compute Mode prohibited
305         // use that
306         if (isIntegrated && (computeMode != CU_COMPUTEMODE_PROHIBITED)) {
307             int major = 0, minor = 0;
308             char deviceName[256];
309             checkCudaErrors(cuDeviceGetAttribute(
310               &major, CU_DEVICE_ATTRIBUTE_COMPUTE_CAPABILITY_MAJOR, current_device));
311             checkCudaErrors(cuDeviceGetAttribute(
312               &minor, CU_DEVICE_ATTRIBUTE_COMPUTE_CAPABILITY_MINOR, current_device));
313             checkCudaErrors(cuDeviceGetName(deviceName, 256, current_device));
314             printf("GPU Device %d: \"%s\" with compute capability %d.%d\n\n",
315                    current_device,
316                    deviceName,
317                    major,
318                    minor);
319
320             return current_device;
321         } else {
322             devices_prohibited++;
323         }
324
325         current_device++;
326     }
327
328     if (devices_prohibited == device_count) {
329         fprintf(stderr, "CUDA error: No Integrated CUDA capable GPU found.\n");
330         exit(EXIT_FAILURE);
331     }
332
333     return -1;
334 }
335
336 // General check for CUDA GPU SM Capabilities
```

```
337 inline bool checkCudaCapabilitiesDRV(int major_version, int minor_version, int devID) {
338     CUdevice cuDevice;
339     char name[256];
340     int major = 0, minor = 0;
341
342     checkCudaErrors(cuDeviceGet(&cuDevice, devID));
343     checkCudaErrors(cuDeviceGetName(name, 100, cuDevice));
344     checkCudaErrors(
345       cuDeviceGetAttribute(&major, CU_DEVICE_ATTRIBUTE_COMPUTE_CAPABILITY_MAJOR, cuDevice));
346     checkCudaErrors(
347       cuDeviceGetAttribute(&minor, CU_DEVICE_ATTRIBUTE_COMPUTE_CAPABILITY_MINOR, cuDevice));
348
349     if ((major > major_version) || (major == major_version && minor >= minor_version)) {
350         printf("> Device %d: <%16s >, Compute SM %d.%d detected\n", devID, name, major, minor);
351         return true;
352     } else {
353         printf(
354           "No GPU device was found that can support CUDA compute capability "
355           "%d.%d.\n",
356           major_version,
357           minor_version);
358         return false;
359     }
360 }
361 #endif
362
363 bool inline findFatbinPath(const char *module_file, std::string &module_path, char **argv,
364                            std::ostringstream &ostrm) {
365     char *actual_path = sdkFindFilePath(module_file, argv[0]);
366
367     if (actual_path) {
368         module_path = actual_path;
369     } else {
370         printf("> findModulePath file not found: <%s> \n", module_file);
371         return false;
372     }
373
374     if (module_path.empty()) {
375         printf("> findModulePath could not find file: <%s> \n", module_file);
376         return false;
377     } else {
378         printf("> findModulePath found file at <%s>\n", module_path.c_str());
379         if (module_path.rfind("fatbin") != std::string::npos) {
380             std::ifstream fileIn(module_path.c_str(), std::ios::binary);
381             ostrm << fileIn.rdbuf();
382             fileIn.close();
383         }
384         return true;
385     }
386 }
387
388 // end of CUDA Helper Functions
389
390 #endif // COMMON_HELPER_CUDA_DRVAPI_H_
```

7.171 helper_cusolver.h

```
1 /* Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
2  *
3  * Redistribution and use in source and binary forms, with or without
4  * modification, are permitted provided that the following conditions
5  * are met:
6  *  * Redistributions of source code must retain the above copyright
7  *    notice, this list of conditions and the following disclaimer.
8  *  * Redistributions in binary form must reproduce the above copyright
9  *    notice, this list of conditions and the following disclaimer in the
10 *    documentation and/or other materials provided with the distribution.
11 *  * Neither the name of NVIDIA CORPORATION nor the names of its
12 *    contributors may be used to endorse or promote products derived
13 *    from this software without specific prior written permission.
14 *
15 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS" AND ANY
16 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
17 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
18 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
19 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
20 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
21 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
22 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
23 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
24 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
25 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26 */
```

```
27
28 #ifndef HELPER_CUSOLVER
29 #define HELPER_CUSOLVER
30
31 #include "cusparse.h"
32 #include <ctype.h>
33 #include <cuda_runtime.h>
34 #include <math.h>
35 #include <stdio.h>
36 #include <stdlib.h>
37 #include <string.h>
38
39 #define SWITCH_CHAR '-'
40
41 struct testOpts {
42     char *sparse_mat_filename; // by switch -F<filename>
43     const char *testFunc;      // by switch -R<name>
44     const char *reorder;       // by switch -P<name>
45     int lda;                   // by switch -lda<int>
46 };
47
48 double vec_norminf(int n, const double *x) {
49     double norminf = 0;
50     for (int j = 0; j < n; j++) {
51         double x_abs = fabs(x[j]);
52         norminf      = (norminf > x_abs) ? norminf : x_abs;
53     }
54     return norminf;
55 }
56
57 /*
58  * |A| = max { |A|*ones(m,1) }
59  */
60 double mat_norminf(int m, int n, const double *A, int lda) {
61     double norminf = 0;
62     for (int i = 0; i < m; i++) {
63         double sum = 0.0;
64         for (int j = 0; j < n; j++) {
65             double A_abs = fabs(A[i + j * lda]);
66             sum += A_abs;
67         }
68         norminf = (norminf > sum) ? norminf : sum;
69     }
70     return norminf;
71 }
72
73 /*
74  * |A| = max { |A|*ones(m,1) }
75  */
76 double csr_mat_norminf(int m, int n, int nnzA, const cusparseMatDescr_t descrA,
77                        const double *csrValA, const int *csrRowPtrA, const int *csrColIndA) {
78     const int baseA = (CUSPARSE_INDEX_BASE_ONE == cusparseGetMatIndexBase(descrA)) ? 1 : 0;
79
80     double norminf = 0;
81     for (int i = 0; i < m; i++) {
82         double sum      = 0.0;
83         const int start = csrRowPtrA[i] - baseA;
84         const int end   = csrRowPtrA[i + 1] - baseA;
85         for (int colidx = start; colidx < end; colidx++) {
86             // const int j = csrColIndA[colidx] - baseA;
87             double A_abs = fabs(csrValA[colidx]);
88             sum += A_abs;
89         }
90         norminf = (norminf > sum) ? norminf : sum;
91     }
92     return norminf;
93 }
94
95 void display_matrix(int m, int n, int nnzA, const cusparseMatDescr_t descrA, const double *csrValA,
96                     const int *csrRowPtrA, const int *csrColIndA) {
97     const int baseA = (CUSPARSE_INDEX_BASE_ONE == cusparseGetMatIndexBase(descrA)) ? 1 : 0;
98
99     printf("m = %d, n = %d, nnz = %d, matlab base-1\n", m, n, nnzA);
100
101     for (int row = 0; row < m; row++) {
102         const int start = csrRowPtrA[row] - baseA;
103         const int end   = csrRowPtrA[row + 1] - baseA;
104         for (int colidx = start; colidx < end; colidx++) {
105             const int col = csrColIndA[colidx] - baseA;
106             double Areg   = csrValA[colidx];
107             printf("A(%d, %d) = %20.16E\n", row + 1, col + 1, Areg);
108         }
109     }
110 }
111
112 #if defined(_WIN32)
113 #   if !defined(WIN32_LEAN_AND_MEAN)
```

```
#       define WIN32_LEAN_AND_MEAN
#   endif

#   include <Windows.h>

double second(void) {
    LARGE_INTEGER t;
    static double oofreq;
    static int checkedForHighResTimer;
    static BOOL hasHighResTimer;

    if (!checkedForHighResTimer) {
        hasHighResTimer        = QueryPerformanceFrequency(&t);
        oofreq                 = 1.0 / (double)t.QuadPart;
        checkedForHighResTimer = 1;
    }
    if (hasHighResTimer) {
        QueryPerformanceCounter(&t);
        return (double)t.QuadPart * oofreq;
    } else {
        return (double)GetTickCount() / 1000.0;
    }
}

#elif defined(__linux__) || defined(__QNX__)
#   include <stddef.h>
#   include <sys/resource.h>
#   include <sys/time.h>
double second(void) {
    struct timeval tv;
    gettimeofday(&tv, NULL);
    return (double)tv.tv_sec + (double)tv.tv_usec / 1000000.0;
}

#elif defined(__APPLE__)
#   include <stddef.h>
#   include <sys/resource.h>
#   include <sys/sysctl.h>
#   include <sys/time.h>
#   include <sys/types.h>
double second(void) {
    struct timeval tv;
    gettimeofday(&tv, NULL);
    return (double)tv.tv_sec + (double)tv.tv_usec / 1000000.0;
}
#else
#   error unsupported platform
#endif

#endif
```

7.172 helper_functions.h

```
/* Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 *  * Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 *  * Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *  * Neither the name of NVIDIA CORPORATION nor the names of its
 *    contributors may be used to endorse or promote products derived
 *    from this software without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS" AND ANY
 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
 */

// These are helper functions for the SDK samples (string parsing,
// timers, image helpers, etc)
#ifndef COMMON_HELPER_FUNCTIONS_H_
```

```
31 #define COMMON_HELPER_FUNCTIONS_H_
32
33 #ifdef WIN32
34 #   pragma warning(disable : 4996)
35 #endif
36
37 // includes, project
38 #include <algorithm>
39 #include <assert.h>
40 #include "exception.h"
41 #include <fstream>
42 #include <iostream>
43 #include <math.h>
44 #include <stdio.h>
45 #include <stdlib.h>
46 #include <string>
47 #include <vector>
48
49 // includes, timer, string parsing, image helpers
50 #include "helper_image.h"  // helper functions for image compare, dump, data comparisons
51 #include "helper_string.h" // helper functions for string parsing
52 #include "helper_timer.h"  // helper functions for timers
53
54 #ifndef EXIT_WAIVED
55 #   define EXIT_WAIVED 2
56 #endif
57
58 #endif // COMMON_HELPER_FUNCTIONS_H_
```

7.173 helper_image.h

```
1 /* Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
2  *
3  * Redistribution and use in source and binary forms, with or without
4  * modification, are permitted provided that the following conditions
5  * are met:
6  *  * Redistributions of source code must retain the above copyright
7  *    notice, this list of conditions and the following disclaimer.
8  *  * Redistributions in binary form must reproduce the above copyright
9  *    notice, this list of conditions and the following disclaimer in the
10 *    documentation and/or other materials provided with the distribution.
11 *  * Neither the name of NVIDIA CORPORATION nor the names of its
12 *    contributors may be used to endorse or promote products derived
13 *    from this software without specific prior written permission.
14 *
15 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS" AND ANY
16 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
17 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
18 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
19 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
20 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
21 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
22 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
23 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
24 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
25 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26 */
27
28 // These are helper functions for the SDK samples (image,bitmap)
29 #ifndef COMMON_HELPER_IMAGE_H_
30 #define COMMON_HELPER_IMAGE_H_
31
32 #include <algorithm>
33 #include <assert.h>
34 #include "exception.h"
35 #include <fstream>
36 #include <iostream>
37 #include <math.h>
38 #include <stdint.h>
39 #include <string>
40 #include <vector>
41
42 #ifndef MIN
43 #   define MIN(a, b) ((a < b) ? a : b)
44 #endif
45 #ifndef MAX
46 #   define MAX(a, b) ((a > b) ? a : b)
47 #endif
48
49 #ifndef EXIT_WAIVED
50 #   define EXIT_WAIVED 2
51 #endif
52
```

```
53 #include "helper_string.h"
54
55 // namespace unnamed (internal)
56 namespace helper_image_internal {
58     const unsigned int PGMHeaderSize = 0x40;
59
60     // types
61
63     template<class T>
64     struct ConverterFromUByte;
65
67     template<>
68     struct ConverterFromUByte<unsigned char> {
72         float operator()(const unsigned char &val) { return static_cast<unsigned char>(val); }
73     };
74
76     template<>
77     struct ConverterFromUByte<float> {
81         float operator()(const unsigned char &val) { return static_cast<float>(val) / 255.0f; }
82     };
83
85     template<class T>
86     struct ConverterToUByte;
87
89     template<>
90     struct ConverterToUByte<unsigned char> {
94         unsigned char operator()(const unsigned char &val) { return val; }
95     };
96
98     template<>
99     struct ConverterToUByte<float> {
103         unsigned char operator()(const float &val) {
104             return static_cast<unsigned char>(val * 255.0f);
105         }
106     };
107 } // namespace helper_image_internal
108
109 #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
110 #   ifndef FOPEN
111 #       define FOPEN(fHandle, filename, mode) fopen_s(&fHandle, filename, mode)
112 #   endif
113 #   ifndef FOPEN_FAIL
114 #       define FOPEN_FAIL(result) (result != 0)
115 #   endif
116 #   ifndef SSCANF
117 #       define SSCANF sscanf_s
118 #   endif
119 #else
120 #   ifndef FOPEN
121 #       define FOPEN(fHandle, filename, mode) (fHandle = fopen(filename, mode))
122 #   endif
123 #   ifndef FOPEN_FAIL
124 #       define FOPEN_FAIL(result) (result == NULL)
125 #   endif
126 #   ifndef SSCANF
127 #       define SSCANF sscanf
128 #   endif
129 #endif
130
131 inline bool __loadPPM(const char *file, unsigned char **data, unsigned int *w, unsigned int *h,
132                       unsigned int *channels) {
133     FILE *fp = NULL;
134
135     if (FOPEN_FAIL(FOPEN(fp, file, "rb"))) {
136         std::cerr « "__LoadPPM() : Failed to open file: " « file « std::endl;
137         return false;
138     }
139
140     // check header
141     char header[helper_image_internal::PGMHeaderSize];
142
143     if (fgets(header, helper_image_internal::PGMHeaderSize, fp) == NULL) {
144         std::cerr « "__LoadPPM() : reading PGM header returned NULL" « std::endl;
145         return false;
146     }
147
148     if (strncmp(header, "P5", 2) == 0) {
149         *channels = 1;
150     } else if (strncmp(header, "P6", 2) == 0) {
151         *channels = 3;
152     } else {
153         std::cerr « "__LoadPPM() : File is not a PPM or PGM image" « std::endl;
154         *channels = 0;
155         return false;
156     }
157
158     // parse header, read maxval, width and height
```

```
159     unsigned int width  = 0;
160     unsigned int height = 0;
161     unsigned int maxval = 0;
162     unsigned int i      = 0;
163 
164     while (i < 3) {
165         if (fgets(header, helper_image_internal::PGMHeaderSize, fp) == NULL) {
166             std::cerr « "__LoadPPM() : reading PGM header returned NULL" « std::endl;
167             return false;
168         }
169 
170         if (header[0] == '#') { continue; }
171 
172         if (i == 0) {
173             i += SSCANF(header, "%u %u %u", &width, &height, &maxval);
174         } else if (i == 1) {
175             i += SSCANF(header, "%u %u", &height, &maxval);
176         } else if (i == 2) {
177             i += SSCANF(header, "%u", &maxval);
178         }
179     }
180 
181     // check if given handle for the data is initialized
182     if (NULL != *data) {
183         if (*w != width || *h != height) {
184             std::cerr « "__LoadPPM() : Invalid image dimensions." « std::endl;
185         }
186     } else {
187         *data = (unsigned char *)malloc(sizeof(unsigned char) * width * height * *channels);
188         *w    = width;
189         *h    = height;
190     }
191 
192     // read and close file
193     if (fread(*data, sizeof(unsigned char), width * height * *channels, fp) == 0) {
194         std::cerr « "__LoadPPM() read data returned error." « std::endl;
195     }
196 
197     fclose(fp);
198 
199     return true;
200 }
201 
202 template<class T>
203 inline bool sdkLoadPGM(const char *file, T **data, unsigned int *w, unsigned int *h) {
204     unsigned char *idata = NULL;
205     unsigned int channels;
206 
207     if (true != __loadPPM(file, &idata, w, h, &channels)) { return false; }
208 
209     unsigned int size = *w * *h * channels;
210 
211     // initialize mem if necessary
212     // the correct size is checked / set in loadPGMc()
213     if (NULL == *data) { *data = reinterpret_cast<T *>(malloc(sizeof(T) * size)); }
214 
215     // copy and cast data
216     std::transform(idata, idata + size, *data, helper_image_internal::ConverterFromUByte<T>());
217 
218     free(idata);
219 
220     return true;
221 }
222 
223 template<class T>
224 inline bool sdkLoadPPM4(const char *file, T **data, unsigned int *w, unsigned int *h) {
225     unsigned char *idata = 0;
226     unsigned int channels;
227 
228     if (__loadPPM(file, &idata, w, h, &channels)) {
229         // pad 4th component
230         int size = *w * *h;
231         // keep the original pointer
232         unsigned char *idata_orig = idata;
233         *data                     = reinterpret_cast<T *>(malloc(sizeof(T) * size * 4));
234         unsigned char *ptr        = *data;
235 
236         for (int i = 0; i < size; i++) {
237             *ptr++ = *idata++;
238             *ptr++ = *idata++;
239             *ptr++ = *idata++;
240             *ptr++ = 0;
241         }
242 
243         free(idata_orig);
244         return true;
245     } else {
```

```
246         free(idata);
247         return false;
248     }
249 }
250
251 inline bool __savePPM(const char *file, unsigned char *data, unsigned int w, unsigned int h,
252                       unsigned int channels) {
253     assert(NULL != data);
254     assert(w > 0);
255     assert(h > 0);
256
257     std::fstream fh(file, std::fstream::out | std::fstream::binary);
258
259     if (fh.bad()) {
260         std::cerr « "__savePPM() : Opening file failed." « std::endl;
261         return false;
262     }
263
264     if (channels == 1) {
265         fh « "P5\n";
266     } else if (channels == 3) {
267         fh « "P6\n";
268     } else {
269         std::cerr « "__savePPM() : Invalid number of channels." « std::endl;
270         return false;
271     }
272
273     fh « w « "\n" « h « "\n" « 0xff « std::endl;
274
275     for (unsigned int i = 0; (i < (w * h * channels)) && fh.good(); ++i) { fh « data[i]; }
276
277     fh.flush();
278
279     if (fh.bad()) {
280         std::cerr « "__savePPM() : Writing data failed." « std::endl;
281         return false;
282     }
283
284     fh.close();
285
286     return true;
287 }
288
289 template<class T>
290 inline bool sdkSavePGM(const char *file, T *data, unsigned int w, unsigned int h) {
291     unsigned int size    = w * h;
292     unsigned char *idata = (unsigned char *)malloc(sizeof(unsigned char) * size);
293
294     std::transform(data, data + size, idata, helper_image_internal::ConverterToUByte<T>());
295
296     // write file
297     bool result = __savePPM(file, idata, w, h, 1);
298
299     // cleanup
300     free(idata);
301
302     return result;
303 }
304
305 inline bool sdkSavePPM4ub(const char *file, unsigned char *data, unsigned int w, unsigned int h) {
306     // strip 4th component
307     int size             = w * h;
308     unsigned char *ndata = (unsigned char *)malloc(sizeof(unsigned char) * size * 3);
309     unsigned char *ptr   = ndata;
310
311     for (int i = 0; i < size; i++) {
312         *ptr++ = *data++;
313         *ptr++ = *data++;
314         *ptr++ = *data++;
315         data++;
316     }
317
318     bool result = __savePPM(file, ndata, w, h, 3);
319     free(ndata);
320     return result;
321 }
322
331 template<class T>
332 inline bool sdkReadFile(const char *filename, T **data, unsigned int *len, bool verbose) {
333     // check input arguments
334     assert(NULL != filename);
335     assert(NULL != len);
336
337     // intermediate storage for the data read
338     std::vector<T> data_read;
339
340     // open file for reading
```

```
341     FILE *fh = NULL;
342
343     // check if filestream is valid
344     if (FOPEN_FAIL(FOPEN(fh, filename, "r"))) {
345         printf("Unable to open input file: %s\n", filename);
346         return false;
347     }
348
349     // read all data elements
350     T token;
351
352     while (!feof(fh)) {
353         fscanf(fh, "%f", &token);
354         data_read.push_back(token);
355     }
356
357     // the last element is read twice
358     data_read.pop_back();
359     fclose(fh);
360
361     // check if the given handle is already initialized
362     if (NULL != *data) {
363         if (*len != data_read.size()) {
364             std::cerr « "sdkReadFile() : Initialized memory given but "
365                      « "size  mismatch with signal read "
366                      « "(data read / data init = " « (unsigned int)data_read.size() « " / "
367                      « *len « ")" « std::endl;
368
369             return false;
370         }
371     } else {
372         // allocate storage for the data read
373         *data = reinterpret_cast<T *>(malloc(sizeof(T) * data_read.size()));
374         // store signal size
375         *len = static_cast<unsigned int>(data_read.size());
376     }
377
378     // copy data
379     memcpy(*data, &data_read.front(), sizeof(T) * data_read.size());
380
381     return true;
382 }
383
392 template<class T>
393 inline bool sdkReadFileBlocks(const char *filename, T **data, unsigned int *len,
394                               unsigned int block_num, unsigned int block_size, bool verbose) {
395     // check input arguments
396     assert(NULL != filename);
397     assert(NULL != len);
398
399     // open file for reading
400     FILE *fh = fopen(filename, "rb");
401
402     if (fh == NULL && verbose) {
403         std::cerr « "sdkReadFile() : Opening file failed." « std::endl;
404         return false;
405     }
406
407     // check if the given handle is already initialized
408     // allocate storage for the data read
409     data[block_num] = reinterpret_cast<T *>(malloc(block_size));
410
411     // read all data elements
412     fseek(fh, block_num * block_size, SEEK_SET);
413     *len = fread(data[block_num], sizeof(T), block_size / sizeof(T), fh);
414
415     fclose(fh);
416
417     return true;
418 }
419
428 template<class T, class S>
429 inline bool sdkWriteFile(const char *filename, const T *data, unsigned int len, const S epsilon,
430                          bool verbose, bool append = false) {
431     assert(NULL != filename);
432     assert(NULL != data);
433
434     // open file for writing
435     //    if (append) {
436     std::fstream fh(filename, std::fstream::out | std::fstream::ate);
437
438     if (verbose) {
439         std::cerr « "sdkWriteFile() : Open file " « filename « " for write/append." « std::endl;
440     }
441
442     /*    } else {
443             std::fstream fh(filename, std::fstream::out);
```

```
444             if (verbose) {
445                 std::cerr « "sdkWriteFile() : Open file " « filename « " for
446        write." « std::endl;
447             }
448         }
449     */
450 
451     // check if filestream is valid
452     if (!fh.good()) {
453         if (verbose) { std::cerr « "sdkWriteFile() : Opening file failed." « std::endl; }
454 
455         return false;
456     }
457 
458     // first write epsilon
459     fh « "# " « epsilon « "\n";
460 
461     // write data
462     for (unsigned int i = 0; (i < len) && (fh.good()); ++i) { fh « data[i] « ' '; }
463 
464     // Check if writing succeeded
465     if (!fh.good()) {
466         if (verbose) { std::cerr « "sdkWriteFile() : Writing file failed." « std::endl; }
467 
468         return false;
469     }
470 
471     // file ends with nl
472     fh « std::endl;
473 
474     return true;
475 }
476 
485 template<class T, class S>
486 inline bool compareData(const T *reference, const T *data, const unsigned int len, const S epsilon,
487                         const float threshold) {
488     assert(epsilon >= 0);
489 
490     bool result              = true;
491     unsigned int error_count = 0;
492 
493     for (unsigned int i = 0; i < len; ++i) {
494         float diff = static_cast<float>(reference[i]) - static_cast<float>(data[i]);
495         bool comp  = (diff <= epsilon) && (diff >= -epsilon);
496         result &= comp;
497 
498         error_count += !comp;
499 
500 #if 0
501 
502         if (!comp) {
503           std::cerr « "ERROR, i = " « i « ",\t "
504                     « reference[i] « " / "
505                     « data[i]
506                     « " (reference / data)\n";
507         }
508 
509 #endif
510     }
511 
512     if (threshold == 0.0f) {
513         return (result) ? true : false;
514     } else {
515         if (error_count) {
516             printf("%4.2f(%%) of bytes mismatched (count=%d)\n",
517                    static_cast<float>(error_count) * 100 / static_cast<float>(len),
518                    error_count);
519         }
520 
521         return (len * threshold > error_count) ? true : false;
522     }
523 }
524 
525 #ifndef __MIN_EPSILON_ERROR
526 #   define __MIN_EPSILON_ERROR 1e-3f
527 #endif
528 
538 template<class T, class S>
539 inline bool compareDataAsFloatThreshold(const T *reference, const T *data, const unsigned int len,
540                                         const S epsilon, const float threshold) {
541     assert(epsilon >= 0);
542 
543     // If we set epsilon to be 0, let's set a minimum threshold
544     float max_error = MAX((float)epsilon, __MIN_EPSILON_ERROR);
545     int error_count = 0;
546     bool result     = true;
547 
```

```
548     for (unsigned int i = 0; i < len; ++i) {
549         float diff = fabs(static_cast<float>(reference[i]) - static_cast<float>(data[i]));
550         bool comp  = (diff < max_error);
551         result &= comp;
552 
553         if (!comp) { error_count++; }
554     }
555 
556     if (threshold == 0.0f) {
557         if (error_count) { printf("total # of errors = %d\n", error_count); }
558 
559         return (error_count == 0) ? true : false;
560     } else {
561         if (error_count) {
562             printf("%4.2f(%%) of bytes mismatched (count=%d)\n",
563                    static_cast<float>(error_count) * 100 / static_cast<float>(len),
564                    error_count);
565         }
566 
567         return ((len * threshold > error_count) ? true : false);
568     }
569 }
570 
571 inline void sdkDumpBin(void *data, unsigned int bytes, const char *filename) {
572     printf("sdkDumpBin: <%s>\n", filename);
573     FILE *fp;
574     FOPEN(fp, filename, "wb");
575     fwrite(data, bytes, 1, fp);
576     fflush(fp);
577     fclose(fp);
578 }
579 
580 inline bool sdkCompareBin2BinUint(const char *src_file, const char *ref_file,
581                                   unsigned int nelements, const float epsilon,
582                                   const float threshold, char *exec_path) {
583     unsigned int *src_buffer, *ref_buffer;
584     FILE *src_fp = NULL, *ref_fp = NULL;
585 
586     uint64_t error_count = 0;
587     size_t fsize         = 0;
588 
589     if (FOPEN_FAIL(FOPEN(src_fp, src_file, "rb"))) {
590         printf("compareBin2Bin <unsigned int> unable to open src_file: %s\n", src_file);
591         error_count++;
592     }
593 
594     char *ref_file_path = sdkFindFilePath(ref_file, exec_path);
595 
596     if (ref_file_path == NULL) {
597         printf("compareBin2Bin <unsigned int>  unable to find <%s> in <%s>\n", ref_file, exec_path);
598         printf("»» Check info.xml and [project//data] folder <%s> ««\n", ref_file);
599         printf("Aborting comparison!\n");
600         printf("  FAILED\n");
601         error_count++;
602 
603         if (src_fp) { fclose(src_fp); }
604 
605         if (ref_fp) { fclose(ref_fp); }
606     } else {
607         if (FOPEN_FAIL(FOPEN(ref_fp, ref_file_path, "rb"))) {
608             printf(
609               "compareBin2Bin <unsigned int>"
610               " unable to open ref_file: %s\n",
611               ref_file_path);
612             error_count++;
613         }
614 
615         if (src_fp && ref_fp) {
616             src_buffer = (unsigned int *)malloc(nelements * sizeof(unsigned int));
617             ref_buffer = (unsigned int *)malloc(nelements * sizeof(unsigned int));
618 
619             fsize = fread(src_buffer, nelements, sizeof(unsigned int), src_fp);
620             fsize = fread(ref_buffer, nelements, sizeof(unsigned int), ref_fp);
621 
622             printf(
623               "> compareBin2Bin <unsigned int> nelements=%d,"
624               " epsilon=%4.2f, threshold=%4.2f\n",
625               nelements,
626               epsilon,
627               threshold);
628             printf("   src_file <%s>, size=%d bytes\n", src_file, static_cast<int>(fsize));
629             printf("   ref_file <%s>, size=%d bytes\n", ref_file_path, static_cast<int>(fsize));
630 
631             if (!compareData<unsigned int, float>(
632                   ref_buffer, src_buffer, nelements, epsilon, threshold)) {
633                 error_count++;
634             }
```

```
635
636             fclose(src_fp);
637             fclose(ref_fp);
638
639             free(src_buffer);
640             free(ref_buffer);
641         } else {
642             if (src_fp) { fclose(src_fp); }
643
644             if (ref_fp) { fclose(ref_fp); }
645         }
646     }
647
648     if (error_count == 0) {
649         printf("  OK\n");
650     } else {
651         printf("  FAILURE: %d errors...\n", (unsigned int)error_count);
652     }
653
654     return (error_count == 0); // returns true if all pixels pass
655 }
656
657 inline bool sdkCompareBin2BinFloat(const char *src_file, const char *ref_file,
658                                    unsigned int nelements, const float epsilon,
659                                    const float threshold, char *exec_path) {
660     float *src_buffer = NULL, *ref_buffer = NULL;
661     FILE *src_fp = NULL, *ref_fp = NULL;
662     size_t fsize = 0;
663
664     uint64_t error_count = 0;
665
666     if (FOPEN_FAIL(FOPEN(src_fp, src_file, "rb"))) {
667         printf("compareBin2Bin <float> unable to open src_file: %s\n", src_file);
668         error_count = 1;
669     }
670
671     char *ref_file_path = sdkFindFilePath(ref_file, exec_path);
672
673     if (ref_file_path == NULL) {
674         printf("compareBin2Bin <float> unable to find <%s> in <%s>\n", ref_file, exec_path);
675         printf("»» Check info.xml and [project//data] folder <%s> ««\n", exec_path);
676         printf("Aborting comparison!\n");
677         printf("  FAILED\n");
678         error_count++;
679
680         if (src_fp) { fclose(src_fp); }
681
682         if (ref_fp) { fclose(ref_fp); }
683     } else {
684         if (FOPEN_FAIL(FOPEN(ref_fp, ref_file_path, "rb"))) {
685             printf("compareBin2Bin <float> unable to open ref_file: %s\n", ref_file_path);
686             error_count = 1;
687         }
688
689         if (src_fp && ref_fp) {
690             src_buffer = reinterpret_cast<float *>(malloc(nelements * sizeof(float)));
691             ref_buffer = reinterpret_cast<float *>(malloc(nelements * sizeof(float)));
692
693             printf(
694               "> compareBin2Bin <float> nelements=%d, epsilon=%4.2f,"
695               " threshold=%4.2f\n",
696               nelements,
697               epsilon,
698               threshold);
699             fsize = fread(src_buffer, sizeof(float), nelements, src_fp);
700             printf("   src_file <%s>, size=%d bytes\n",
701                    src_file,
702                    static_cast<int>(fsize * sizeof(float)));
703             fsize = fread(ref_buffer, sizeof(float), nelements, ref_fp);
704             printf("   ref_file <%s>, size=%d bytes\n",
705                    ref_file_path,
706                    static_cast<int>(fsize * sizeof(float)));
707
708             if (!compareDataAsFloatThreshold<float, float>(
709                   ref_buffer, src_buffer, nelements, epsilon, threshold)) {
710                 error_count++;
711             }
712
713             fclose(src_fp);
714             fclose(ref_fp);
715
716             free(src_buffer);
717             free(ref_buffer);
718         } else {
719             if (src_fp) { fclose(src_fp); }
720
721             if (ref_fp) { fclose(ref_fp); }
```

```
722         }
723     }
724 
725     if (error_count == 0) {
726         printf("  OK\n");
727     } else {
728         printf("  FAILURE: %d errors...\n", (unsigned int)error_count);
729     }
730 
731     return (error_count == 0); // returns true if all pixels pass
732 }
733 
734 inline bool sdkCompareL2fe(const float *reference, const float *data, const unsigned int len,
735                            const float epsilon) {
736     assert(epsilon >= 0);
737 
738     float error = 0;
739     float ref   = 0;
740 
741     for (unsigned int i = 0; i < len; ++i) {
742         float diff = reference[i] - data[i];
743         error += diff * diff;
744         ref += reference[i] * reference[i];
745     }
746 
747     float normRef = sqrtf(ref);
748 
749     if (fabs(ref) < 1e-7) {
750 #ifdef _DEBUG
751         std::cerr « "ERROR, reference l2-norm is 0\n";
752 #endif
753         return false;
754     }
755 
756     float normError = sqrtf(error);
757     error           = normError / normRef;
758     bool result     = error < epsilon;
759 #ifdef _DEBUG
760 
761     if (!result) {
762         std::cerr « "ERROR, l2-norm error " « error « " is greater than epsilon " « epsilon
763                   « "\n";
764     }
765 
766 #endif
767 
768     return result;
769 }
770 
771 inline bool sdkLoadPPMub(const char *file, unsigned char **data, unsigned int *w, unsigned int *h) {
772     unsigned int channels;
773     return __loadPPM(file, data, w, h, &channels);
774 }
775 
776 inline bool sdkLoadPPM4ub(const char *file, unsigned char **data, unsigned int *w,
777                           unsigned int *h) {
778     unsigned char *idata = 0;
779     unsigned int channels;
780 
781     if (__loadPPM(file, &idata, w, h, &channels)) {
782         // pad 4th component
783         int size = *w * *h;
784         // keep the original pointer
785         unsigned char *idata_orig = idata;
786         *data                     = (unsigned char *)malloc(sizeof(unsigned char) * size * 4);
787         unsigned char *ptr        = *data;
788 
789         for (int i = 0; i < size; i++) {
790             *ptr++ = *idata++;
791             *ptr++ = *idata++;
792             *ptr++ = *idata++;
793             *ptr++ = 0;
794         }
795 
796         free(idata_orig);
797         return true;
798     } else {
799         free(idata);
800         return false;
801     }
802 }
803 
804 inline bool sdkComparePPM(const char *src_file, const char *ref_file, const float epsilon,
805                           const float threshold, bool verboseErrors) {
806     unsigned char *src_data, *ref_data;
807     uint64_t error_count = 0;
808     unsigned int ref_width, ref_height;
```

```
809     unsigned int src_width, src_height;
810
811     if (src_file == NULL || ref_file == NULL) {
812         if (verboseErrors) {
813             std::cerr « "PPMvsPPM: src_file or ref_file is NULL."
814                       "  Aborting comparison\n";
815         }
816
817         return false;
818     }
819
820     if (verboseErrors) {
821         std::cerr « "> Compare (a)rendered:  <" « src_file « ">\n";
822         std::cerr « ">          (b)reference: <" « ref_file « ">\n";
823     }
824
825     if (sdkLoadPPM4ub(ref_file, &ref_data, &ref_width, &ref_height) != true) {
826         if (verboseErrors) {
827             std::cerr « "PPMvsPPM: unable to load ref image file: " « ref_file « "\n";
828         }
829
830         return false;
831     }
832
833     if (sdkLoadPPM4ub(src_file, &src_data, &src_width, &src_height) != true) {
834         std::cerr « "PPMvsPPM: unable to load src image file: " « src_file « "\n";
835         return false;
836     }
837
838     if (src_height != ref_height || src_width != ref_width) {
839         if (verboseErrors) {
840             std::cerr « "PPMvsPPM: source and ref size mismatch (" « src_width « ","
841                       « src_height « ")vs(" « ref_width « "," « ref_height « ")\n";
842         }
843     }
844
845     if (verboseErrors) {
846         std::cerr « "PPMvsPPM: comparing images size (" « src_width « "," « src_height
847                   « ") epsilon(" « epsilon « "), threshold(" « threshold * 100 « "%)\n";
848     }
849
850     if (compareData(ref_data, src_data, src_width * src_height * 4, epsilon, threshold) == false) {
851         error_count = 1;
852     }
853
854     if (error_count == 0) {
855         if (verboseErrors) { std::cerr « "    OK\n\n"; }
856     } else {
857         if (verboseErrors) { std::cerr « "    FAILURE!  " « error_count « " errors...\n\n"; }
858     }
859
860     // returns true if all pixels pass
861     return (error_count == 0) ? true : false;
862 }
863
864 inline bool sdkComparePGM(const char *src_file, const char *ref_file, const float epsilon,
865                           const float threshold, bool verboseErrors) {
866     unsigned char *src_data = 0, *ref_data = 0;
867     uint64_t error_count = 0;
868     unsigned int ref_width, ref_height;
869     unsigned int src_width, src_height;
870
871     if (src_file == NULL || ref_file == NULL) {
872         if (verboseErrors) {
873             std::cerr « "PGMvsPGM: src_file or ref_file is NULL."
874                       "  Aborting comparison\n";
875         }
876
877         return false;
878     }
879
880     if (verboseErrors) {
881         std::cerr « "> Compare (a)rendered:  <" « src_file « ">\n";
882         std::cerr « ">          (b)reference: <" « ref_file « ">\n";
883     }
884
885     if (sdkLoadPPMub(ref_file, &ref_data, &ref_width, &ref_height) != true) {
886         if (verboseErrors) {
887             std::cerr « "PGMvsPGM: unable to load ref image file: " « ref_file « "\n";
888         }
889
890         return false;
891     }
892
893     if (sdkLoadPPMub(src_file, &src_data, &src_width, &src_height) != true) {
894         std::cerr « "PGMvsPGM: unable to load src image file: " « src_file « "\n";
895         return false;
```

```
896     }
897
898     if (src_height != ref_height || src_width != ref_width) {
899         if (verboseErrors) {
900             std::cerr « "PGMvsPGM: source and ref size mismatch (" « src_width « ","
901                       « src_height « ")vs(" « ref_width « "," « ref_height « ")\n";
902         }
903     }
904
905     if (verboseErrors)
906         std::cerr « "PGMvsPGM: comparing images size (" « src_width « "," « src_height
907                   « ") epsilon(" « epsilon « "), threshold(" « threshold * 100 « "%)\n";
908
909     if (compareData(ref_data, src_data, src_width * src_height, epsilon, threshold) == false) {
910         error_count = 1;
911     }
912
913     if (error_count == 0) {
914         if (verboseErrors) { std::cerr « "    OK\n\n"; }
915     } else {
916         if (verboseErrors) { std::cerr « "    FAILURE!  " « error_count « " errors...\n\n"; }
917     }
918
919     // returns true if all pixels pass
920     return (error_count == 0) ? true : false;
921 }
922
923 #endif // COMMON_HELPER_IMAGE_H_
```

7.174 helper_math.h

```
1  /* Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
2   *
3   * Redistribution and use in source and binary forms, with or without
4   * modification, are permitted provided that the following conditions
5   * are met:
6   *  * Redistributions of source code must retain the above copyright
7   *    notice, this list of conditions and the following disclaimer.
8   *  * Redistributions in binary form must reproduce the above copyright
9   *    notice, this list of conditions and the following disclaimer in the
10  *    documentation and/or other materials provided with the distribution.
11  *  * Neither the name of NVIDIA CORPORATION nor the names of its
12  *    contributors may be used to endorse or promote products derived
13  *    from this software without specific prior written permission.
14  *
15  * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS" AND ANY
16  * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
17  * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
18  * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
19  * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
20  * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
21  * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
22  * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
23  * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
24  * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
25  * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26  */
27
28 /*
29  *  This file implements common mathematical operations on vector types
30  *  (float3, float4 etc.) since these are not provided as standard by CUDA.
31  *
32  *  The syntax is modeled on the Cg standard library.
33  *
34  *  This is part of the Helper library includes
35  *
36  *    Thanks to Linh Hah for additions and fixes.
37  */
38
39 #ifndef HELPER_MATH_H
40 #define HELPER_MATH_H
41
42 #include "cuda_runtime.h"
43
44 typedef unsigned int uint;
45
46 typedef unsigned short ushort;
47
48 #ifndef EXIT_WAIVED
49 #   define EXIT_WAIVED 2
50 #endif
51
52 #ifndef __CUDACC__
```

```
53
54 #   include <math.h>
55
57 // host implementations of CUDA functions
59
60 inline float fminf(float a, float b) { return a < b ? a : b; }
61
62 inline float fmaxf(float a, float b) { return a > b ? a : b; }
63
64 inline int max(int a, int b) { return a > b ? a : b; }
65
66 inline int min(int a, int b) { return a < b ? a : b; }
67
68 inline float rsqrtf(float x) { return 1.0f / sqrtf(x); }
69
70 #endif
71
73 // constructors
75
76 inline __host__ __device__
77
78   float2
79   make_float2(float s) {
80     return make_float2(s, s);
81 }
82
83 inline __host__ __device__
84
85   float2
86   make_float2(float3 a) {
87     return make_float2(a.x, a.y);
88 }
89
90 inline __host__ __device__
91
92   float2
93   make_float2(int2 a) {
94     return make_float2(float(a.x), float(a.y));
95 }
96
97 inline __host__ __device__
98
99   float2
100   make_float2(uint2 a) {
101     return make_float2(float(a.x), float(a.y));
102 }
103
104 inline __host__ __device__
105
106   int2
107   make_int2(int s) {
108     return make_int2(s, s);
109 }
110
111 inline __host__ __device__
112
113   int2
114   make_int2(int3 a) {
115     return make_int2(a.x, a.y);
116 }
117
118 inline __host__ __device__
119
120   int2
121   make_int2(uint2 a) {
122     return make_int2(int(a.x), int(a.y));
123 }
124
125 inline __host__ __device__
126
127   int2
128   make_int2(float2 a) {
129     return make_int2(int(a.x), int(a.y));
130 }
131
132 inline __host__ __device__
133
134   uint2
135   make_uint2(uint s) {
136     return make_uint2(s, s);
137 }
138
139 inline __host__ __device__
140
141   uint2
142   make_uint2(uint3 a) {
143     return make_uint2(a.x, a.y);
```

```
144 }
145
146 inline __host__ __device__
147
148   uint2
149   make_uint2(int2 a) {
150     return make_uint2(uint(a.x), uint(a.y));
151 }
152
153 inline __host__ __device__
154
155   float3
156   make_float3(float s) {
157     return make_float3(s, s, s);
158 }
159
160 inline __host__ __device__
161
162   float3
163   make_float3(float2 a) {
164     return make_float3(a.x, a.y, 0.0f);
165 }
166
167 inline __host__ __device__
168
169   float3
170   make_float3(float2 a, float s) {
171     return make_float3(a.x, a.y, s);
172 }
173
174 inline __host__ __device__
175
176   float3
177   make_float3(float4 a) {
178     return make_float3(a.x, a.y, a.z);
179 }
180
181 inline __host__ __device__
182
183   float3
184   make_float3(int3 a) {
185     return make_float3(float(a.x), float(a.y), float(a.z));
186 }
187
188 inline __host__ __device__
189
190   float3
191   make_float3(uint3 a) {
192     return make_float3(float(a.x), float(a.y), float(a.z));
193 }
194
195 inline __host__ __device__
196
197   int3
198   make_int3(int s) {
199     return make_int3(s, s, s);
200 }
201
202 inline __host__ __device__
203
204   int3
205   make_int3(int2 a) {
206     return make_int3(a.x, a.y, 0);
207 }
208
209 inline __host__ __device__
210
211   int3
212   make_int3(int2 a, int s) {
213     return make_int3(a.x, a.y, s);
214 }
215
216 inline __host__ __device__
217
218   int3
219   make_int3(uint3 a) {
220     return make_int3(int(a.x), int(a.y), int(a.z));
221 }
222
223 inline __host__ __device__
224
225   int3
226   make_int3(float3 a) {
227     return make_int3(int(a.x), int(a.y), int(a.z));
228 }
229
230 inline __host__ __device__
```

```
231
232   uint3
233   make_uint3(uint s) {
234     return make_uint3(s, s, s);
235 }
236
237 inline __host__ __device__
238
239   uint3
240   make_uint3(uint2 a) {
241     return make_uint3(a.x, a.y, 0);
242 }
243
244 inline __host__ __device__
245
246   uint3
247   make_uint3(uint2 a, uint s) {
248     return make_uint3(a.x, a.y, s);
249 }
250
251 inline __host__ __device__
252
253   uint3
254   make_uint3(uint4 a) {
255     return make_uint3(a.x, a.y, a.z);
256 }
257
258 inline __host__ __device__
259
260   uint3
261   make_uint3(int3 a) {
262     return make_uint3(uint(a.x), uint(a.y), uint(a.z));
263 }
264
265 inline __host__ __device__
266
267   float4
268   make_float4(float s) {
269     return make_float4(s, s, s, s);
270 }
271
272 inline __host__ __device__
273
274   float4
275   make_float4(float3 a) {
276     return make_float4(a.x, a.y, a.z, 0.0f);
277 }
278
279 inline __host__ __device__
280
281   float4
282   make_float4(float3 a, float w) {
283     return make_float4(a.x, a.y, a.z, w);
284 }
285
286 inline __host__ __device__
287
288   float4
289   make_float4(int4 a) {
290     return make_float4(float(a.x), float(a.y), float(a.z), float(a.w));
291 }
292
293 inline __host__ __device__
294
295   float4
296   make_float4(uint4 a) {
297     return make_float4(float(a.x), float(a.y), float(a.z), float(a.w));
298 }
299
300 inline __host__ __device__
301
302   int4
303   make_int4(int s) {
304     return make_int4(s, s, s, s);
305 }
306
307 inline __host__ __device__
308
309   int4
310   make_int4(int3 a) {
311     return make_int4(a.x, a.y, a.z, 0);
312 }
313
314 inline __host__ __device__
315
316   int4
317   make_int4(int3 a, int w) {
```

```
318     return make_int4(a.x, a.y, a.z, w);
319 }
320
321 inline __host__ __device__
322
323   int4
324   make_int4(uint4 a) {
325     return make_int4(int(a.x), int(a.y), int(a.z), int(a.w));
326 }
327
328 inline __host__ __device__
329
330   int4
331   make_int4(float4 a) {
332     return make_int4(int(a.x), int(a.y), int(a.z), int(a.w));
333 }
334
335 inline __host__ __device__
336
337   uint4
338   make_uint4(uint s) {
339     return make_uint4(s, s, s, s);
340 }
341
342 inline __host__ __device__
343
344   uint4
345   make_uint4(uint3 a) {
346     return make_uint4(a.x, a.y, a.z, 0);
347 }
348
349 inline __host__ __device__
350
351   uint4
352   make_uint4(uint3 a, uint w) {
353     return make_uint4(a.x, a.y, a.z, w);
354 }
355
356 inline __host__ __device__
357
358   uint4
359   make_uint4(int4 a) {
360     return make_uint4(uint(a.x), uint(a.y), uint(a.z), uint(a.w));
361 }
362
364 // negate
366
367 inline __host__ __device__
368
369   float2
370   operator-(float2 &a) {
371     return make_float2(-a.x, -a.y);
372 }
373
374 inline __host__ __device__
375
376   int2
377   operator-(int2 &a) {
378     return make_int2(-a.x, -a.y);
379 }
380
381 inline __host__ __device__
382
383   float3
384   operator-(float3 &a) {
385     return make_float3(-a.x, -a.y, -a.z);
386 }
387
388 inline __host__ __device__
389
390   int3
391   operator-(int3 &a) {
392     return make_int3(-a.x, -a.y, -a.z);
393 }
394
395 inline __host__ __device__
396
397   float4
398   operator-(float4 &a) {
399     return make_float4(-a.x, -a.y, -a.z, -a.w);
400 }
401
402 inline __host__ __device__
403
404   int4
405   operator-(int4 &a) {
406     return make_int4(-a.x, -a.y, -a.z, -a.w);
```

```
407 }
408
410 // addition
412
413 inline __host__ __device__
414
415   float2
416   operator+(float2 a, float2 b) {
417     return make_float2(a.x + b.x, a.y + b.y);
418 }
419
420 inline __host__ __device__
421
422   void
423   operator+=(float2 &a, float2 b) {
424     a.x += b.x;
425     a.y += b.y;
426 }
427
428 inline __host__ __device__
429
430   float2
431   operator+(float2 a, float b) {
432     return make_float2(a.x + b, a.y + b);
433 }
434
435 inline __host__ __device__
436
437   float2
438   operator+(float b, float2 a) {
439     return make_float2(a.x + b, a.y + b);
440 }
441
442 inline __host__ __device__
443
444   void
445   operator+=(float2 &a, float b) {
446     a.x += b;
447     a.y += b;
448 }
449
450 inline __host__ __device__
451
452   int2
453   operator+(int2 a, int2 b) {
454     return make_int2(a.x + b.x, a.y + b.y);
455 }
456
457 inline __host__ __device__
458
459   void
460   operator+=(int2 &a, int2 b) {
461     a.x += b.x;
462     a.y += b.y;
463 }
464
465 inline __host__ __device__
466
467   int2
468   operator+(int2 a, int b) {
469     return make_int2(a.x + b, a.y + b);
470 }
471
472 inline __host__ __device__
473
474   int2
475   operator+(int b, int2 a) {
476     return make_int2(a.x + b, a.y + b);
477 }
478
479 inline __host__ __device__
480
481   void
482   operator+=(int2 &a, int b) {
483     a.x += b;
484     a.y += b;
485 }
486
487 inline __host__ __device__
488
489   uint2
490   operator+(uint2 a, uint2 b) {
491     return make_uint2(a.x + b.x, a.y + b.y);
492 }
493
494 inline __host__ __device__
495
```

```
496   void
497   operator+=(uint2 &a, uint2 b) {
498     a.x += b.x;
499     a.y += b.y;
500 }
501
502 inline __host__ __device__
503
504   uint2
505   operator+(uint2 a, uint b) {
506     return make_uint2(a.x + b, a.y + b);
507 }
508
509 inline __host__ __device__
510
511   uint2
512   operator+(uint b, uint2 a) {
513     return make_uint2(a.x + b, a.y + b);
514 }
515
516 inline __host__ __device__
517
518   void
519   operator+=(uint2 &a, uint b) {
520     a.x += b;
521     a.y += b;
522 }
523
524 inline __host__ __device__
525
526   float3
527   operator+(float3 a, float3 b) {
528     return make_float3(a.x + b.x, a.y + b.y, a.z + b.z);
529 }
530
531 inline __host__ __device__
532
533   void
534   operator+=(float3 &a, float3 b) {
535     a.x += b.x;
536     a.y += b.y;
537     a.z += b.z;
538 }
539
540 inline __host__ __device__
541
542   float3
543   operator+(float3 a, float b) {
544     return make_float3(a.x + b, a.y + b, a.z + b);
545 }
546
547 inline __host__ __device__
548
549   void
550   operator+=(float3 &a, float b) {
551     a.x += b;
552     a.y += b;
553     a.z += b;
554 }
555
556 inline __host__ __device__
557
558   int3
559   operator+(int3 a, int3 b) {
560     return make_int3(a.x + b.x, a.y + b.y, a.z + b.z);
561 }
562
563 inline __host__ __device__
564
565   void
566   operator+=(int3 &a, int3 b) {
567     a.x += b.x;
568     a.y += b.y;
569     a.z += b.z;
570 }
571
572 inline __host__ __device__
573
574   int3
575   operator+(int3 a, int b) {
576     return make_int3(a.x + b, a.y + b, a.z + b);
577 }
578
579 inline __host__ __device__
580
581   void
582   operator+=(int3 &a, int b) {
```

```
583     a.x += b;
584     a.y += b;
585     a.z += b;
586 }
587
588 inline __host__ __device__
589
590   uint3
591   operator+(uint3 a, uint3 b) {
592     return make_uint3(a.x + b.x, a.y + b.y, a.z + b.z);
593 }
594
595 inline __host__ __device__
596
597   void
598   operator+=(uint3 &a, uint3 b) {
599     a.x += b.x;
600     a.y += b.y;
601     a.z += b.z;
602 }
603
604 inline __host__ __device__
605
606   uint3
607   operator+(uint3 a, uint b) {
608     return make_uint3(a.x + b, a.y + b, a.z + b);
609 }
610
611 inline __host__ __device__
612
613   void
614   operator+=(uint3 &a, uint b) {
615     a.x += b;
616     a.y += b;
617     a.z += b;
618 }
619
620 inline __host__ __device__
621
622   int3
623   operator+(int b, int3 a) {
624     return make_int3(a.x + b, a.y + b, a.z + b);
625 }
626
627 inline __host__ __device__
628
629   uint3
630   operator+(uint b, uint3 a) {
631     return make_uint3(a.x + b, a.y + b, a.z + b);
632 }
633
634 inline __host__ __device__
635
636   float3
637   operator+(float b, float3 a) {
638     return make_float3(a.x + b, a.y + b, a.z + b);
639 }
640
641 inline __host__ __device__
642
643   float4
644   operator+(float4 a, float4 b) {
645     return make_float4(a.x + b.x, a.y + b.y, a.z + b.z, a.w + b.w);
646 }
647
648 inline __host__ __device__
649
650   void
651   operator+=(float4 &a, float4 b) {
652     a.x += b.x;
653     a.y += b.y;
654     a.z += b.z;
655     a.w += b.w;
656 }
657
658 inline __host__ __device__
659
660   float4
661   operator+(float4 a, float b) {
662     return make_float4(a.x + b, a.y + b, a.z + b, a.w + b);
663 }
664
665 inline __host__ __device__
666
667   float4
668   operator+(float b, float4 a) {
669     return make_float4(a.x + b, a.y + b, a.z + b, a.w + b);
```

```
670 }
671
672 inline __host__ __device__
673
674   void
675   operator+=(float4 &a, float b) {
676     a.x += b;
677     a.y += b;
678     a.z += b;
679     a.w += b;
680 }
681
682 inline __host__ __device__
683
684   int4
685   operator+(int4 a, int4 b) {
686     return make_int4(a.x + b.x, a.y + b.y, a.z + b.z, a.w + b.w);
687 }
688
689 inline __host__ __device__
690
691   void
692   operator+=(int4 &a, int4 b) {
693     a.x += b.x;
694     a.y += b.y;
695     a.z += b.z;
696     a.w += b.w;
697 }
698
699 inline __host__ __device__
700
701   int4
702   operator+(int4 a, int b) {
703     return make_int4(a.x + b, a.y + b, a.z + b, a.w + b);
704 }
705
706 inline __host__ __device__
707
708   int4
709   operator+(int b, int4 a) {
710     return make_int4(a.x + b, a.y + b, a.z + b, a.w + b);
711 }
712
713 inline __host__ __device__
714
715   void
716   operator+=(int4 &a, int b) {
717     a.x += b;
718     a.y += b;
719     a.z += b;
720     a.w += b;
721 }
722
723 inline __host__ __device__
724
725   uint4
726   operator+(uint4 a, uint4 b) {
727     return make_uint4(a.x + b.x, a.y + b.y, a.z + b.z, a.w + b.w);
728 }
729
730 inline __host__ __device__
731
732   void
733   operator+=(uint4 &a, uint4 b) {
734     a.x += b.x;
735     a.y += b.y;
736     a.z += b.z;
737     a.w += b.w;
738 }
739
740 inline __host__ __device__
741
742   uint4
743   operator+(uint4 a, uint b) {
744     return make_uint4(a.x + b, a.y + b, a.z + b, a.w + b);
745 }
746
747 inline __host__ __device__
748
749   uint4
750   operator+(uint b, uint4 a) {
751     return make_uint4(a.x + b, a.y + b, a.z + b, a.w + b);
752 }
753
754 inline __host__ __device__
755
756   void
```

```
757   operator+=(uint4 &a, uint b) {
758     a.x += b;
759     a.y += b;
760     a.z += b;
761     a.w += b;
762 }
763
765 // subtract
767
768 inline __host__ __device__
769
770   float2
771   operator-(float2 a, float2 b) {
772     return make_float2(a.x - b.x, a.y - b.y);
773 }
774
775 inline __host__ __device__
776
777   void
778   operator-=(float2 &a, float2 b) {
779     a.x -= b.x;
780     a.y -= b.y;
781 }
782
783 inline __host__ __device__
784
785   float2
786   operator-(float2 a, float b) {
787     return make_float2(a.x - b, a.y - b);
788 }
789
790 inline __host__ __device__
791
792   float2
793   operator-(float b, float2 a) {
794     return make_float2(b - a.x, b - a.y);
795 }
796
797 inline __host__ __device__
798
799   void
800   operator-=(float2 &a, float b) {
801     a.x -= b;
802     a.y -= b;
803 }
804
805 inline __host__ __device__
806
807   int2
808   operator-(int2 a, int2 b) {
809     return make_int2(a.x - b.x, a.y - b.y);
810 }
811
812 inline __host__ __device__
813
814   void
815   operator-=(int2 &a, int2 b) {
816     a.x -= b.x;
817     a.y -= b.y;
818 }
819
820 inline __host__ __device__
821
822   int2
823   operator-(int2 a, int b) {
824     return make_int2(a.x - b, a.y - b);
825 }
826
827 inline __host__ __device__
828
829   int2
830   operator-(int b, int2 a) {
831     return make_int2(b - a.x, b - a.y);
832 }
833
834 inline __host__ __device__
835
836   void
837   operator-=(int2 &a, int b) {
838     a.x -= b;
839     a.y -= b;
840 }
841
842 inline __host__ __device__
843
844   uint2
845   operator-(uint2 a, uint2 b) {
```

```
846     return make_uint2(a.x - b.x, a.y - b.y);
847 }
848
849 inline __host__ __device__
850
851   void
852   operator-=(uint2 &a, uint2 b) {
853     a.x -= b.x;
854     a.y -= b.y;
855 }
856
857 inline __host__ __device__
858
859   uint2
860   operator-(uint2 a, uint b) {
861     return make_uint2(a.x - b, a.y - b);
862 }
863
864 inline __host__ __device__
865
866   uint2
867   operator-(uint b, uint2 a) {
868     return make_uint2(b - a.x, b - a.y);
869 }
870
871 inline __host__ __device__
872
873   void
874   operator-=(uint2 &a, uint b) {
875     a.x -= b;
876     a.y -= b;
877 }
878
879 inline __host__ __device__
880
881   float3
882   operator-(float3 a, float3 b) {
883     return make_float3(a.x - b.x, a.y - b.y, a.z - b.z);
884 }
885
886 inline __host__ __device__
887
888   void
889   operator-=(float3 &a, float3 b) {
890     a.x -= b.x;
891     a.y -= b.y;
892     a.z -= b.z;
893 }
894
895 inline __host__ __device__
896
897   float3
898   operator-(float3 a, float b) {
899     return make_float3(a.x - b, a.y - b, a.z - b);
900 }
901
902 inline __host__ __device__
903
904   float3
905   operator-(float b, float3 a) {
906     return make_float3(b - a.x, b - a.y, b - a.z);
907 }
908
909 inline __host__ __device__
910
911   void
912   operator-=(float3 &a, float b) {
913     a.x -= b;
914     a.y -= b;
915     a.z -= b;
916 }
917
918 inline __host__ __device__
919
920   int3
921   operator-(int3 a, int3 b) {
922     return make_int3(a.x - b.x, a.y - b.y, a.z - b.z);
923 }
924
925 inline __host__ __device__
926
927   void
928   operator-=(int3 &a, int3 b) {
929     a.x -= b.x;
930     a.y -= b.y;
931     a.z -= b.z;
932 }
```

```
933
934 inline __host__ __device__
935
936   int3
937   operator-(int3 a, int b) {
938     return make_int3(a.x - b, a.y - b, a.z - b);
939 }
940
941 inline __host__ __device__
942
943   int3
944   operator-(int b, int3 a) {
945     return make_int3(b - a.x, b - a.y, b - a.z);
946 }
947
948 inline __host__ __device__
949
950   void
951   operator-=(int3 &a, int b) {
952     a.x -= b;
953     a.y -= b;
954     a.z -= b;
955 }
956
957 inline __host__ __device__
958
959   uint3
960   operator-(uint3 a, uint3 b) {
961     return make_uint3(a.x - b.x, a.y - b.y, a.z - b.z);
962 }
963
964 inline __host__ __device__
965
966   void
967   operator-=(uint3 &a, uint3 b) {
968     a.x -= b.x;
969     a.y -= b.y;
970     a.z -= b.z;
971 }
972
973 inline __host__ __device__
974
975   uint3
976   operator-(uint3 a, uint b) {
977     return make_uint3(a.x - b, a.y - b, a.z - b);
978 }
979
980 inline __host__ __device__
981
982   uint3
983   operator-(uint b, uint3 a) {
984     return make_uint3(b - a.x, b - a.y, b - a.z);
985 }
986
987 inline __host__ __device__
988
989   void
990   operator-=(uint3 &a, uint b) {
991     a.x -= b;
992     a.y -= b;
993     a.z -= b;
994 }
995
996 inline __host__ __device__
997
998   float4
999   operator-(float4 a, float4 b) {
1000     return make_float4(a.x - b.x, a.y - b.y, a.z - b.z, a.w - b.w);
1001 }
1002
1003 inline __host__ __device__
1004
1005   void
1006   operator-=(float4 &a, float4 b) {
1007     a.x -= b.x;
1008     a.y -= b.y;
1009     a.z -= b.z;
1010     a.w -= b.w;
1011 }
1012
1013 inline __host__ __device__
1014
1015   float4
1016   operator-(float4 a, float b) {
1017     return make_float4(a.x - b, a.y - b, a.z - b, a.w - b);
1018 }
1019
```

```
1020 inline __host__ __device__
1021
1022   void
1023   operator-=(float4 &a, float b) {
1024     a.x -= b;
1025     a.y -= b;
1026     a.z -= b;
1027     a.w -= b;
1028 }
1029
1030 inline __host__ __device__
1031
1032   int4
1033   operator-(int4 a, int4 b) {
1034     return make_int4(a.x - b.x, a.y - b.y, a.z - b.z, a.w - b.w);
1035 }
1036
1037 inline __host__ __device__
1038
1039   void
1040   operator-=(int4 &a, int4 b) {
1041     a.x -= b.x;
1042     a.y -= b.y;
1043     a.z -= b.z;
1044     a.w -= b.w;
1045 }
1046
1047 inline __host__ __device__
1048
1049   int4
1050   operator-(int4 a, int b) {
1051     return make_int4(a.x - b, a.y - b, a.z - b, a.w - b);
1052 }
1053
1054 inline __host__ __device__
1055
1056   int4
1057   operator-(int b, int4 a) {
1058     return make_int4(b - a.x, b - a.y, b - a.z, b - a.w);
1059 }
1060
1061 inline __host__ __device__
1062
1063   void
1064   operator-=(int4 &a, int b) {
1065     a.x -= b;
1066     a.y -= b;
1067     a.z -= b;
1068     a.w -= b;
1069 }
1070
1071 inline __host__ __device__
1072
1073   uint4
1074   operator-(uint4 a, uint4 b) {
1075     return make_uint4(a.x - b.x, a.y - b.y, a.z - b.z, a.w - b.w);
1076 }
1077
1078 inline __host__ __device__
1079
1080   void
1081   operator-=(uint4 &a, uint4 b) {
1082     a.x -= b.x;
1083     a.y -= b.y;
1084     a.z -= b.z;
1085     a.w -= b.w;
1086 }
1087
1088 inline __host__ __device__
1089
1090   uint4
1091   operator-(uint4 a, uint b) {
1092     return make_uint4(a.x - b, a.y - b, a.z - b, a.w - b);
1093 }
1094
1095 inline __host__ __device__
1096
1097   uint4
1098   operator-(uint b, uint4 a) {
1099     return make_uint4(b - a.x, b - a.y, b - a.z, b - a.w);
1100 }
1101
1102 inline __host__ __device__
1103
1104   void
1105   operator-=(uint4 &a, uint b) {
1106     a.x -= b;
```

```
1107     a.y -= b;
1108     a.z -= b;
1109     a.w -= b;
1110 }
1111
1113 // multiply
1115
1116 inline __host__ __device__
1117
1118   float2
1119   operator*(float2 a, float2 b) {
1120     return make_float2(a.x * b.x, a.y * b.y);
1121 }
1122
1123 inline __host__ __device__
1124
1125   void
1126   operator*=(float2 &a, float2 b) {
1127     a.x *= b.x;
1128     a.y *= b.y;
1129 }
1130
1131 inline __host__ __device__
1132
1133   float2
1134   operator*(float2 a, float b) {
1135     return make_float2(a.x * b, a.y * b);
1136 }
1137
1138 inline __host__ __device__
1139
1140   float2
1141   operator*(float b, float2 a) {
1142     return make_float2(b * a.x, b * a.y);
1143 }
1144
1145 inline __host__ __device__
1146
1147   void
1148   operator*=(float2 &a, float b) {
1149     a.x *= b;
1150     a.y *= b;
1151 }
1152
1153 inline __host__ __device__
1154
1155   int2
1156   operator*(int2 a, int2 b) {
1157     return make_int2(a.x * b.x, a.y * b.y);
1158 }
1159
1160 inline __host__ __device__
1161
1162   void
1163   operator*=(int2 &a, int2 b) {
1164     a.x *= b.x;
1165     a.y *= b.y;
1166 }
1167
1168 inline __host__ __device__
1169
1170   int2
1171   operator*(int2 a, int b) {
1172     return make_int2(a.x * b, a.y * b);
1173 }
1174
1175 inline __host__ __device__
1176
1177   int2
1178   operator*(int b, int2 a) {
1179     return make_int2(b * a.x, b * a.y);
1180 }
1181
1182 inline __host__ __device__
1183
1184   void
1185   operator*=(int2 &a, int b) {
1186     a.x *= b;
1187     a.y *= b;
1188 }
1189
1190 inline __host__ __device__
1191
1192   uint2
1193   operator*(uint2 a, uint2 b) {
1194     return make_uint2(a.x * b.x, a.y * b.y);
1195 }
```

```
1196
1197 inline __host__ __device__
1198
1199   void
1200   operator*=(uint2 &a, uint2 b) {
1201     a.x *= b.x;
1202     a.y *= b.y;
1203 }
1204
1205 inline __host__ __device__
1206
1207   uint2
1208   operator*(uint2 a, uint b) {
1209     return make_uint2(a.x * b, a.y * b);
1210 }
1211
1212 inline __host__ __device__
1213
1214   uint2
1215   operator*(uint b, uint2 a) {
1216     return make_uint2(b * a.x, b * a.y);
1217 }
1218
1219 inline __host__ __device__
1220
1221   void
1222   operator*=(uint2 &a, uint b) {
1223     a.x *= b;
1224     a.y *= b;
1225 }
1226
1227 inline __host__ __device__
1228
1229   float3
1230   operator*(float3 a, float3 b) {
1231     return make_float3(a.x * b.x, a.y * b.y, a.z * b.z);
1232 }
1233
1234 inline __host__ __device__
1235
1236   void
1237   operator*=(float3 &a, float3 b) {
1238     a.x *= b.x;
1239     a.y *= b.y;
1240     a.z *= b.z;
1241 }
1242
1243 inline __host__ __device__
1244
1245   float3
1246   operator*(float3 a, float b) {
1247     return make_float3(a.x * b, a.y * b, a.z * b);
1248 }
1249
1250 inline __host__ __device__
1251
1252   float3
1253   operator*(float b, float3 a) {
1254     return make_float3(b * a.x, b * a.y, b * a.z);
1255 }
1256
1257 inline __host__ __device__
1258
1259   void
1260   operator*=(float3 &a, float b) {
1261     a.x *= b;
1262     a.y *= b;
1263     a.z *= b;
1264 }
1265
1266 inline __host__ __device__
1267
1268   int3
1269   operator*(int3 a, int3 b) {
1270     return make_int3(a.x * b.x, a.y * b.y, a.z * b.z);
1271 }
1272
1273 inline __host__ __device__
1274
1275   void
1276   operator*=(int3 &a, int3 b) {
1277     a.x *= b.x;
1278     a.y *= b.y;
1279     a.z *= b.z;
1280 }
1281
1282 inline __host__ __device__
```

```
1283
1284   int3
1285   operator*(int3 a, int b) {
1286     return make_int3(a.x * b, a.y * b, a.z * b);
1287 }
1288
1289 inline __host__ __device__
1290
1291   int3
1292   operator*(int b, int3 a) {
1293     return make_int3(b * a.x, b * a.y, b * a.z);
1294 }
1295
1296 inline __host__ __device__
1297
1298   void
1299   operator*=(int3 &a, int b) {
1300     a.x *= b;
1301     a.y *= b;
1302     a.z *= b;
1303 }
1304
1305 inline __host__ __device__
1306
1307   uint3
1308   operator*(uint3 a, uint3 b) {
1309     return make_uint3(a.x * b.x, a.y * b.y, a.z * b.z);
1310 }
1311
1312 inline __host__ __device__
1313
1314   void
1315   operator*=(uint3 &a, uint3 b) {
1316     a.x *= b.x;
1317     a.y *= b.y;
1318     a.z *= b.z;
1319 }
1320
1321 inline __host__ __device__
1322
1323   uint3
1324   operator*(uint3 a, uint b) {
1325     return make_uint3(a.x * b, a.y * b, a.z * b);
1326 }
1327
1328 inline __host__ __device__
1329
1330   uint3
1331   operator*(uint b, uint3 a) {
1332     return make_uint3(b * a.x, b * a.y, b * a.z);
1333 }
1334
1335 inline __host__ __device__
1336
1337   void
1338   operator*=(uint3 &a, uint b) {
1339     a.x *= b;
1340     a.y *= b;
1341     a.z *= b;
1342 }
1343
1344 inline __host__ __device__
1345
1346   float4
1347   operator*(float4 a, float4 b) {
1348     return make_float4(a.x * b.x, a.y * b.y, a.z * b.z, a.w * b.w);
1349 }
1350
1351 inline __host__ __device__
1352
1353   void
1354   operator*=(float4 &a, float4 b) {
1355     a.x *= b.x;
1356     a.y *= b.y;
1357     a.z *= b.z;
1358     a.w *= b.w;
1359 }
1360
1361 inline __host__ __device__
1362
1363   float4
1364   operator*(float4 a, float b) {
1365     return make_float4(a.x * b, a.y * b, a.z * b, a.w * b);
1366 }
1367
1368 inline __host__ __device__
1369
```

```
1370   float4
1371   operator*(float b, float4 a) {
1372     return make_float4(b * a.x, b * a.y, b * a.z, b * a.w);
1373 }
1374
1375 inline __host__ __device__
1376
1377   void
1378   operator*=(float4 &a, float b) {
1379     a.x *= b;
1380     a.y *= b;
1381     a.z *= b;
1382     a.w *= b;
1383 }
1384
1385 inline __host__ __device__
1386
1387   int4
1388   operator*(int4 a, int4 b) {
1389     return make_int4(a.x * b.x, a.y * b.y, a.z * b.z, a.w * b.w);
1390 }
1391
1392 inline __host__ __device__
1393
1394   void
1395   operator*=(int4 &a, int4 b) {
1396     a.x *= b.x;
1397     a.y *= b.y;
1398     a.z *= b.z;
1399     a.w *= b.w;
1400 }
1401
1402 inline __host__ __device__
1403
1404   int4
1405   operator*(int4 a, int b) {
1406     return make_int4(a.x * b, a.y * b, a.z * b, a.w * b);
1407 }
1408
1409 inline __host__ __device__
1410
1411   int4
1412   operator*(int b, int4 a) {
1413     return make_int4(b * a.x, b * a.y, b * a.z, b * a.w);
1414 }
1415
1416 inline __host__ __device__
1417
1418   void
1419   operator*=(int4 &a, int b) {
1420     a.x *= b;
1421     a.y *= b;
1422     a.z *= b;
1423     a.w *= b;
1424 }
1425
1426 inline __host__ __device__
1427
1428   uint4
1429   operator*(uint4 a, uint4 b) {
1430     return make_uint4(a.x * b.x, a.y * b.y, a.z * b.z, a.w * b.w);
1431 }
1432
1433 inline __host__ __device__
1434
1435   void
1436   operator*=(uint4 &a, uint4 b) {
1437     a.x *= b.x;
1438     a.y *= b.y;
1439     a.z *= b.z;
1440     a.w *= b.w;
1441 }
1442
1443 inline __host__ __device__
1444
1445   uint4
1446   operator*(uint4 a, uint b) {
1447     return make_uint4(a.x * b, a.y * b, a.z * b, a.w * b);
1448 }
1449
1450 inline __host__ __device__
1451
1452   uint4
1453   operator*(uint b, uint4 a) {
1454     return make_uint4(b * a.x, b * a.y, b * a.z, b * a.w);
1455 }
1456
```

```
1457 inline __host__ __device__
1458
1459   void
1460   operator*=(uint4 &a, uint b) {
1461     a.x *= b;
1462     a.y *= b;
1463     a.z *= b;
1464     a.w *= b;
1465 }
1466
1468 // divide
1470
1471 inline __host__ __device__
1472
1473   float2
1474   operator/(float2 a, float2 b) {
1475     return make_float2(a.x / b.x, a.y / b.y);
1476 }
1477
1478 inline __host__ __device__
1479
1480   void
1481   operator/=(float2 &a, float2 b) {
1482     a.x /= b.x;
1483     a.y /= b.y;
1484 }
1485
1486 inline __host__ __device__
1487
1488   float2
1489   operator/(float2 a, float b) {
1490     return make_float2(a.x / b, a.y / b);
1491 }
1492
1493 inline __host__ __device__
1494
1495   void
1496   operator/=(float2 &a, float b) {
1497     a.x /= b;
1498     a.y /= b;
1499 }
1500
1501 inline __host__ __device__
1502
1503   float2
1504   operator/(float b, float2 a) {
1505     return make_float2(b / a.x, b / a.y);
1506 }
1507
1508 inline __host__ __device__
1509
1510   float3
1511   operator/(float3 a, float3 b) {
1512     return make_float3(a.x / b.x, a.y / b.y, a.z / b.z);
1513 }
1514
1515 inline __host__ __device__
1516
1517   void
1518   operator/=(float3 &a, float3 b) {
1519     a.x /= b.x;
1520     a.y /= b.y;
1521     a.z /= b.z;
1522 }
1523
1524 inline __host__ __device__
1525
1526   float3
1527   operator/(float3 a, float b) {
1528     return make_float3(a.x / b, a.y / b, a.z / b);
1529 }
1530
1531 inline __host__ __device__
1532
1533   void
1534   operator/=(float3 &a, float b) {
1535     a.x /= b;
1536     a.y /= b;
1537     a.z /= b;
1538 }
1539
1540 inline __host__ __device__
1541
1542   float3
1543   operator/(float b, float3 a) {
1544     return make_float3(b / a.x, b / a.y, b / a.z);
1545 }
```

```
1546
1547 inline __host__ __device__
1548
1549   float4
1550   operator/(float4 a, float4 b) {
1551     return make_float4(a.x / b.x, a.y / b.y, a.z / b.z, a.w / b.w);
1552 }
1553
1554 inline __host__ __device__
1555
1556   void
1557   operator/=(float4 &a, float4 b) {
1558     a.x /= b.x;
1559     a.y /= b.y;
1560     a.z /= b.z;
1561     a.w /= b.w;
1562 }
1563
1564 inline __host__ __device__
1565
1566   float4
1567   operator/(float4 a, float b) {
1568     return make_float4(a.x / b, a.y / b, a.z / b, a.w / b);
1569 }
1570
1571 inline __host__ __device__
1572
1573   void
1574   operator/=(float4 &a, float b) {
1575     a.x /= b;
1576     a.y /= b;
1577     a.z /= b;
1578     a.w /= b;
1579 }
1580
1581 inline __host__ __device__
1582
1583   float4
1584   operator/(float b, float4 a) {
1585     return make_float4(b / a.x, b / a.y, b / a.z, b / a.w);
1586 }
1587
1589 // min
1591
1592 inline __host__ __device__
1593
1594   float2
1595   fminf(float2 a, float2 b) {
1596     return make_float2(fminf(a.x, b.x), fminf(a.y, b.y));
1597 }
1598
1599 inline __host__ __device__
1600
1601   float3
1602   fminf(float3 a, float3 b) {
1603     return make_float3(fminf(a.x, b.x), fminf(a.y, b.y), fminf(a.z, b.z));
1604 }
1605
1606 inline __host__ __device__
1607
1608   float4
1609   fminf(float4 a, float4 b) {
1610     return make_float4(fminf(a.x, b.x), fminf(a.y, b.y), fminf(a.z, b.z), fminf(a.w, b.w));
1611 }
1612
1613 inline __host__ __device__
1614
1615   int2
1616   min(int2 a, int2 b) {
1617     return make_int2(min(a.x, b.x), min(a.y, b.y));
1618 }
1619
1620 inline __host__ __device__
1621
1622   int3
1623   min(int3 a, int3 b) {
1624     return make_int3(min(a.x, b.x), min(a.y, b.y), min(a.z, b.z));
1625 }
1626
1627 inline __host__ __device__
1628
1629   int4
1630   min(int4 a, int4 b) {
1631     return make_int4(min(a.x, b.x), min(a.y, b.y), min(a.z, b.z), min(a.w, b.w));
1632 }
1633
1634 inline __host__ __device__
```

```
1635
1636   uint2
1637   min(uint2 a, uint2 b) {
1638     return make_uint2(min(a.x, b.x), min(a.y, b.y));
1639 }
1640
1641 inline __host__ __device__
1642
1643   uint3
1644   min(uint3 a, uint3 b) {
1645     return make_uint3(min(a.x, b.x), min(a.y, b.y), min(a.z, b.z));
1646 }
1647
1648 inline __host__ __device__
1649
1650   uint4
1651   min(uint4 a, uint4 b) {
1652     return make_uint4(min(a.x, b.x), min(a.y, b.y), min(a.z, b.z), min(a.w, b.w));
1653 }
1654
1656 // max
1658
1659 inline __host__ __device__
1660
1661   float2
1662   fmaxf(float2 a, float2 b) {
1663     return make_float2(fmaxf(a.x, b.x), fmaxf(a.y, b.y));
1664 }
1665
1666 inline __host__ __device__
1667
1668   float3
1669   fmaxf(float3 a, float3 b) {
1670     return make_float3(fmaxf(a.x, b.x), fmaxf(a.y, b.y), fmaxf(a.z, b.z));
1671 }
1672
1673 inline __host__ __device__
1674
1675   float4
1676   fmaxf(float4 a, float4 b) {
1677     return make_float4(fmaxf(a.x, b.x), fmaxf(a.y, b.y), fmaxf(a.z, b.z), fmaxf(a.w, b.w));
1678 }
1679
1680 inline __host__ __device__
1681
1682   int2
1683   max(int2 a, int2 b) {
1684     return make_int2(max(a.x, b.x), max(a.y, b.y));
1685 }
1686
1687 inline __host__ __device__
1688
1689   int3
1690   max(int3 a, int3 b) {
1691     return make_int3(max(a.x, b.x), max(a.y, b.y), max(a.z, b.z));
1692 }
1693
1694 inline __host__ __device__
1695
1696   int4
1697   max(int4 a, int4 b) {
1698     return make_int4(max(a.x, b.x), max(a.y, b.y), max(a.z, b.z), max(a.w, b.w));
1699 }
1700
1701 inline __host__ __device__
1702
1703   uint2
1704   max(uint2 a, uint2 b) {
1705     return make_uint2(max(a.x, b.x), max(a.y, b.y));
1706 }
1707
1708 inline __host__ __device__
1709
1710   uint3
1711   max(uint3 a, uint3 b) {
1712     return make_uint3(max(a.x, b.x), max(a.y, b.y), max(a.z, b.z));
1713 }
1714
1715 inline __host__ __device__
1716
1717   uint4
1718   max(uint4 a, uint4 b) {
1719     return make_uint4(max(a.x, b.x), max(a.y, b.y), max(a.z, b.z), max(a.w, b.w));
1720 }
1721
1723 // lerp
1724 // - linear interpolation between a and b, based on value t in [0, 1] range
```

```
1726
1727 inline __device__ __host__
1728
1729   float
1730   lerp(float a, float b, float t) {
1731     return a + t * (b - a);
1732 }
1733
1734 inline __device__ __host__
1735
1736   float2
1737   lerp(float2 a, float2 b, float t) {
1738     return a + t * (b - a);
1739 }
1740
1741 inline __device__ __host__
1742
1743   float3
1744   lerp(float3 a, float3 b, float t) {
1745     return a + t * (b - a);
1746 }
1747
1748 inline __device__ __host__
1749
1750   float4
1751   lerp(float4 a, float4 b, float t) {
1752     return a + t * (b - a);
1753 }
1754
1756 // clamp
1757 // - clamp the value v to be in the range [a, b]
1759
1760 inline __device__ __host__
1761
1762   float
1763   clamp(float f, float a, float b) {
1764     return fmaxf(a, fminf(f, b));
1765 }
1766
1767 inline __device__ __host__
1768
1769 int clamp(int f, int a, int b)
1770 {
1771     return max(a, min(f, b));
1772 }
1773
1774 inline __device__ __host__
1775
1776   uint
1777   clamp(uint f, uint a, uint b) {
1778     return max(a, min(f, b));
1779 }
1780
1781 inline __device__ __host__
1782
1783   float2
1784   clamp(float2 v, float a, float b) {
1785     return make_float2(clamp(v.x, a, b), clamp(v.y, a, b));
1786 }
1787
1788 inline __device__ __host__
1789
1790   float2
1791   clamp(float2 v, float2 a, float2 b) {
1792     return make_float2(clamp(v.x, a.x, b.x), clamp(v.y, a.y, b.y));
1793 }
1794
1795 inline __device__ __host__
1796
1797   float3
1798   clamp(float3 v, float a, float b) {
1799     return make_float3(clamp(v.x, a, b), clamp(v.y, a, b), clamp(v.z, a, b));
1800 }
1801
1802 inline __device__ __host__
1803
1804   float3
1805   clamp(float3 v, float3 a, float3 b) {
1806     return make_float3(clamp(v.x, a.x, b.x), clamp(v.y, a.y, b.y), clamp(v.z, a.z, b.z));
1807 }
1808
1809 inline __device__ __host__
1810
1811   float4
1812   clamp(float4 v, float a, float b) {
1813     return make_float4(clamp(v.x, a, b), clamp(v.y, a, b), clamp(v.z, a, b), clamp(v.w, a, b));
1814 }
```

```
1815
1816 inline __device__ __host__
1817
1818   float4
1819   clamp(float4 v, float4 a, float4 b) {
1820     return make_float4(
1821       clamp(v.x, a.x, b.x), clamp(v.y, a.y, b.y), clamp(v.z, a.z, b.z), clamp(v.w, a.w, b.w));
1822 }
1823
1824 inline __device__ __host__
1825
1826   int2
1827   clamp(int2 v, int a, int b) {
1828     return make_int2(clamp(v.x, a, b), clamp(v.y, a, b));
1829 }
1830
1831 inline __device__ __host__
1832
1833   int2
1834   clamp(int2 v, int2 a, int2 b) {
1835     return make_int2(clamp(v.x, a.x, b.x), clamp(v.y, a.y, b.y));
1836 }
1837
1838 inline __device__ __host__
1839
1840   int3
1841   clamp(int3 v, int a, int b) {
1842     return make_int3(clamp(v.x, a, b), clamp(v.y, a, b), clamp(v.z, a, b));
1843 }
1844
1845 inline __device__ __host__
1846
1847   int3
1848   clamp(int3 v, int3 a, int3 b) {
1849     return make_int3(clamp(v.x, a.x, b.x), clamp(v.y, a.y, b.y), clamp(v.z, a.z, b.z));
1850 }
1851
1852 inline __device__ __host__
1853
1854   int4
1855   clamp(int4 v, int a, int b) {
1856     return make_int4(clamp(v.x, a, b), clamp(v.y, a, b), clamp(v.z, a, b), clamp(v.w, a, b));
1857 }
1858
1859 inline __device__ __host__
1860
1861   int4
1862   clamp(int4 v, int4 a, int4 b) {
1863     return make_int4(
1864       clamp(v.x, a.x, b.x), clamp(v.y, a.y, b.y), clamp(v.z, a.z, b.z), clamp(v.w, a.w, b.w));
1865 }
1866
1867 inline __device__ __host__
1868
1869   uint2
1870   clamp(uint2 v, uint a, uint b) {
1871     return make_uint2(clamp(v.x, a, b), clamp(v.y, a, b));
1872 }
1873
1874 inline __device__ __host__
1875
1876   uint2
1877   clamp(uint2 v, uint2 a, uint2 b) {
1878     return make_uint2(clamp(v.x, a.x, b.x), clamp(v.y, a.y, b.y));
1879 }
1880
1881 inline __device__ __host__
1882
1883   uint3
1884   clamp(uint3 v, uint a, uint b) {
1885     return make_uint3(clamp(v.x, a, b), clamp(v.y, a, b), clamp(v.z, a, b));
1886 }
1887
1888 inline __device__ __host__
1889
1890   uint3
1891   clamp(uint3 v, uint3 a, uint3 b) {
1892     return make_uint3(clamp(v.x, a.x, b.x), clamp(v.y, a.y, b.y), clamp(v.z, a.z, b.z));
1893 }
1894
1895 inline __device__ __host__
1896
1897   uint4
1898   clamp(uint4 v, uint a, uint b) {
1899     return make_uint4(clamp(v.x, a, b), clamp(v.y, a, b), clamp(v.z, a, b), clamp(v.w, a, b));
1900 }
1901
```

```
1902 inline __device__ __host__
1903
1904   uint4
1905   clamp(uint4 v, uint4 a, uint4 b) {
1906     return make_uint4(
1907       clamp(v.x, a.x, b.x), clamp(v.y, a.y, b.y), clamp(v.z, a.z, b.z), clamp(v.w, a.w, b.w));
1908 }
1909
1911 // dot product
1913
1914 inline __host__ __device__
1915
1916   float
1917   dot(float2 a, float2 b) {
1918     return a.x * b.x + a.y * b.y;
1919 }
1920
1921 inline __host__ __device__
1922
1923   float
1924   dot(float3 a, float3 b) {
1925     return a.x * b.x + a.y * b.y + a.z * b.z;
1926 }
1927
1928 inline __host__ __device__
1929
1930   float
1931   dot(float4 a, float4 b) {
1932     return a.x * b.x + a.y * b.y + a.z * b.z + a.w * b.w;
1933 }
1934
1935 inline __host__ __device__
1936
1937 int dot(int2 a, int2 b)
1938 {
1939     return a.x * b.x + a.y * b.y;
1940 }
1941
1942 inline __host__ __device__
1943
1944 int dot(int3 a, int3 b)
1945 {
1946     return a.x * b.x + a.y * b.y + a.z * b.z;
1947 }
1948
1949 inline __host__ __device__
1950
1951 int dot(int4 a, int4 b)
1952 {
1953     return a.x * b.x + a.y * b.y + a.z * b.z + a.w * b.w;
1954 }
1955
1956 inline __host__ __device__
1957
1958   uint
1959   dot(uint2 a, uint2 b) {
1960     return a.x * b.x + a.y * b.y;
1961 }
1962
1963 inline __host__ __device__
1964
1965   uint
1966   dot(uint3 a, uint3 b) {
1967     return a.x * b.x + a.y * b.y + a.z * b.z;
1968 }
1969
1970 inline __host__ __device__
1971
1972   uint
1973   dot(uint4 a, uint4 b) {
1974     return a.x * b.x + a.y * b.y + a.z * b.z + a.w * b.w;
1975 }
1976
1978 // length
1980
1981 inline __host__ __device__
1982
1983   float
1984   length(float2 v) {
1985     return sqrtf(dot(v, v));
1986 }
1987
1988 inline __host__ __device__
1989
1990   float
1991   length(float3 v) {
1992     return sqrtf(dot(v, v));
```

```
1993 }
1994
1995 inline __host__ __device__
1996
1997   float
1998   length(float4 v) {
1999     return sqrtf(dot(v, v));
2000 }
2001
2003 // normalize
2005
2006 inline __host__ __device__
2007
2008   float2
2009   normalize(float2 v) {
2010     float invLen = rsqrtf(dot(v, v));
2011     return v * invLen;
2012 }
2013
2014 inline __host__ __device__
2015
2016   float3
2017   normalize(float3 v) {
2018     float invLen = rsqrtf(dot(v, v));
2019     return v * invLen;
2020 }
2021
2022 inline __host__ __device__
2023
2024   float4
2025   normalize(float4 v) {
2026     float invLen = rsqrtf(dot(v, v));
2027     return v * invLen;
2028 }
2029
2031 // floor
2033
2034 inline __host__ __device__
2035
2036   float2
2037   floorf(float2 v) {
2038     return make_float2(floorf(v.x), floorf(v.y));
2039 }
2040
2041 inline __host__ __device__
2042
2043   float3
2044   floorf(float3 v) {
2045     return make_float3(floorf(v.x), floorf(v.y), floorf(v.z));
2046 }
2047
2048 inline __host__ __device__
2049
2050   float4
2051   floorf(float4 v) {
2052     return make_float4(floorf(v.x), floorf(v.y), floorf(v.z), floorf(v.w));
2053 }
2054
2056 // frac - returns the fractional portion of a scalar or each vector component
2058
2059 inline __host__ __device__
2060
2061   float
2062   fracf(float v) {
2063     return v - floorf(v);
2064 }
2065
2066 inline __host__ __device__
2067
2068   float2
2069   fracf(float2 v) {
2070     return make_float2(fracf(v.x), fracf(v.y));
2071 }
2072
2073 inline __host__ __device__
2074
2075   float3
2076   fracf(float3 v) {
2077     return make_float3(fracf(v.x), fracf(v.y), fracf(v.z));
2078 }
2079
2080 inline __host__ __device__
2081
2082   float4
2083   fracf(float4 v) {
2084     return make_float4(fracf(v.x), fracf(v.y), fracf(v.z), fracf(v.w));
2085 }
```

```
2086
2088 // fmod
2090
2091 inline __host__ __device__
2092
2093   float2
2094   fmodf(float2 a, float2 b) {
2095     return make_float2(fmodf(a.x, b.x), fmodf(a.y, b.y));
2096 }
2097
2098 inline __host__ __device__
2099
2100   float3
2101   fmodf(float3 a, float3 b) {
2102     return make_float3(fmodf(a.x, b.x), fmodf(a.y, b.y), fmodf(a.z, b.z));
2103 }
2104
2105 inline __host__ __device__
2106
2107   float4
2108   fmodf(float4 a, float4 b) {
2109     return make_float4(fmodf(a.x, b.x), fmodf(a.y, b.y), fmodf(a.z, b.z), fmodf(a.w, b.w));
2110 }
2111
2113 // absolute value
2115
2116 inline __host__ __device__
2117
2118   float2
2119   fabs(float2 v) {
2120     return make_float2(fabs(v.x), fabs(v.y));
2121 }
2122
2123 inline __host__ __device__
2124
2125   float3
2126   fabs(float3 v) {
2127     return make_float3(fabs(v.x), fabs(v.y), fabs(v.z));
2128 }
2129
2130 inline __host__ __device__
2131
2132   float4
2133   fabs(float4 v) {
2134     return make_float4(fabs(v.x), fabs(v.y), fabs(v.z), fabs(v.w));
2135 }
2136
2137 inline __host__ __device__
2138
2139   int2
2140   abs(int2 v) {
2141     return make_int2(abs(v.x), abs(v.y));
2142 }
2143
2144 inline __host__ __device__
2145
2146   int3
2147   abs(int3 v) {
2148     return make_int3(abs(v.x), abs(v.y), abs(v.z));
2149 }
2150
2151 inline __host__ __device__
2152
2153   int4
2154   abs(int4 v) {
2155     return make_int4(abs(v.x), abs(v.y), abs(v.z), abs(v.w));
2156 }
2157
2159 // reflect
2160 // - returns reflection of incident ray I around surface normal N
2161 // - N should be normalized, reflected vector's length is equal to length of I
2163
2164 inline __host__ __device__
2165
2166   float3
2167   reflect(float3 i, float3 n) {
2168     return i - 2.0f * n * dot(n, i);
2169 }
2170
2172 // cross product
2174
2175 inline __host__ __device__
2176
2177   float3
2178   cross(float3 a, float3 b) {
2179     return make_float3(a.y * b.z - a.z * b.y, a.z * b.x - a.x * b.z, a.x * b.y - a.y * b.x);
2180 }
```

```
2181
2183 // smoothstep
2184 // - returns 0 if x < a
2185 // - returns 1 if x > b
2186 // - otherwise returns smooth interpolation between 0 and 1 based on x
2188
2189 inline __device__ __host__
2190
2191   float
2192   smoothstep(float a, float b, float x) {
2193     float y = clamp((x - a) / (b - a), 0.0f, 1.0f);
2194     return (y * y * (3.0f - (2.0f * y)));
2195 }
2196
2197 inline __device__ __host__
2198
2199   float2
2200   smoothstep(float2 a, float2 b, float2 x) {
2201     float2 y = clamp((x - a) / (b - a), 0.0f, 1.0f);
2202     return (y * y * (make_float2(3.0f) - (make_float2(2.0f) * y)));
2203 }
2204
2205 inline __device__ __host__
2206
2207   float3
2208   smoothstep(float3 a, float3 b, float3 x) {
2209     float3 y = clamp((x - a) / (b - a), 0.0f, 1.0f);
2210     return (y * y * (make_float3(3.0f) - (make_float3(2.0f) * y)));
2211 }
2212
2213 inline __device__ __host__
2214
2215   float4
2216   smoothstep(float4 a, float4 b, float4 x) {
2217     float4 y = clamp((x - a) / (b - a), 0.0f, 1.0f);
2218     return (y * y * (make_float4(3.0f) - (make_float4(2.0f) * y)));
2219 }
2220
2221 #endif
```

7.175 helper_multiprocess.h

```
1 /* Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
2  *
3  * Redistribution and use in source and binary forms, with or without
4  * modification, are permitted provided that the following conditions
5  * are met:
6  *  * Redistributions of source code must retain the above copyright
7  *    notice, this list of conditions and the following disclaimer.
8  *  * Redistributions in binary form must reproduce the above copyright
9  *    notice, this list of conditions and the following disclaimer in the
10 *    documentation and/or other materials provided with the distribution.
11 *  * Neither the name of NVIDIA CORPORATION nor the names of its
12 *    contributors may be used to endorse or promote products derived
13 *    from this software without specific prior written permission.
14 *
15 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS" AND ANY
16 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
17 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
18 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
19 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
20 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
21 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
22 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
23 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
24 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
25 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26 */
27
28 #ifndef HELPER_MULTIPROCESS_H
29 #define HELPER_MULTIPROCESS_H
30
31 #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
32 #   ifndef WIN32_LEAN_AND_MEAN
33 #       define WIN32_LEAN_AND_MEAN
34 #   endif
35
36 #   include <AclAPI.h>
37 #   include <iostream>
38 #   include <sddl.h>
39 #   include <stdio.h>
40 #   include <strsafe.h>
41 #   include <tchar.h>
```

```
42 #   include <Windows.h>
43 #   include <winternl.h>
44
45 #else
46 #   include <errno.h>
47 #   include <fcntl.h>
48 #   include <memory.h>
49 #   include <stdio.h>
50 #   include <sys/mman.h>
51 #   include <sys/socket.h>
52 #   include <sys/types.h>
53 #   include <sys/un.h>
54 #   include <sys/wait.h>
55 #   include <unistd.h>
56 #endif
57
58 #include <vector>
59
60 typedef struct sharedMemoryInfo_st {
61     void *addr;
62     size_t size;
63 #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
64     HANDLE shmHandle;
65 #else
66     int shmFd;
67 #endif
68 } sharedMemoryInfo;
69
70 int sharedMemoryCreate(const char *name, size_t sz, sharedMemoryInfo *info);
71
72 int sharedMemoryOpen(const char *name, size_t sz, sharedMemoryInfo *info);
73
74 void sharedMemoryClose(sharedMemoryInfo *info);
75
76 #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
77
78 typedef PROCESS_INFORMATION Process;
79
80 #else
81 typedef pid_t Process;
82 #endif
83
84 int spawnProcess(Process *process, const char *app, char *const *args);
85
86 int waitProcess(Process *process);
87
88 #define checkIpcErrors(ipcFuncResult)                                                  \
89     if (ipcFuncResult == -1) {                                                         \
90         fprintf(stderr, "Failure at %u %s\n", __LINE__, __FILE__);                     \
91         exit(EXIT_FAILURE);                                                            \
92     }
93
94 #if defined(__linux__)
95 struct ipcHandle_st {
96     int socket;
97     char *socketName;
98 };
99 typedef int ShareableHandle;
100 #elif defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
101 struct ipcHandle_st {
102     std::vector<HANDLE> hMailslot; // 1 Handle in case of child and `num
103                                    // children` Handles for parent.
104 };
105
106 typedef HANDLE ShareableHandle;
107
108 #endif
109
110 typedef struct ipcHandle_st ipcHandle;
111
112 int ipcCreateSocket(ipcHandle *&handle, const char *name, const std::vector<Process> &processes);
113
114 int ipcOpenSocket(ipcHandle *&handle);
115
116 int ipcCloseSocket(ipcHandle *handle);
117
118 int ipcRecvShareableHandles(ipcHandle *handle, std::vector<ShareableHandle> &shareableHandles);
119
120 int ipcSendShareableHandles(ipcHandle *handle, const std::vector<ShareableHandle> &shareableHandles,
121                             const std::vector<Process> &processes);
122
123 int ipcCloseShareableHandle(ShareableHandle shHandle);
124
125 #endif // HELPER_MULTIPROCESS_H
```

7.176 helper_string.h

```
1 /* Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
2  *
3  * Redistribution and use in source and binary forms, with or without
4  * modification, are permitted provided that the following conditions
5  * are met:
6  *  * Redistributions of source code must retain the above copyright
7  *    notice, this list of conditions and the following disclaimer.
8  *  * Redistributions in binary form must reproduce the above copyright
9  *    notice, this list of conditions and the following disclaimer in the
10 *    documentation and/or other materials provided with the distribution.
11 *  * Neither the name of NVIDIA CORPORATION nor the names of its
12 *    contributors may be used to endorse or promote products derived
13 *    from this software without specific prior written permission.
14 *
15 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS" AND ANY
16 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
17 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
18 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
19 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
20 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
21 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
22 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
23 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
24 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
25 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26 */
27
28 // These are helper functions for the SDK samples (string parsing, timers, etc)
29 #ifndef COMMON_HELPER_STRING_H_
30 #define COMMON_HELPER_STRING_H_
31
32 #include <fstream>
33 #include <stdio.h>
34 #include <stdlib.h>
35 #include <string>
36
37 #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
38 #   ifndef _CRT_SECURE_NO_DEPRECATE
39 #       define _CRT_SECURE_NO_DEPRECATE
40 #   endif
41 #   ifndef STRCASECMP
42 #       define STRCASECMP _stricmp
43 #   endif
44 #   ifndef STRNCASECMP
45 #       define STRNCASECMP _strnicmp
46 #   endif
47 #   ifndef STRCPY
48 #       define STRCPY(sFilePath, nLength, sPath) strcpy_s(sFilePath, nLength, sPath)
49 #   endif
50
51 #   ifndef FOPEN
52 #       define FOPEN(fHandle, filename, mode) fopen_s(&fHandle, filename, mode)
53 #   endif
54 #   ifndef FOPEN_FAIL
55 #       define FOPEN_FAIL(result) (result != 0)
56 #   endif
57 #   ifndef SSCANF
58 #       define SSCANF sscanf_s
59 #   endif
60 #   ifndef SPRINTF
61 #       define SPRINTF sprintf_s
62 #   endif
63 #else // Linux Includes
64 #   include <string.h>
65 #   include <strings.h>
66
67 #   ifndef STRCASECMP
68 #       define STRCASECMP strcasecmp
69 #   endif
70 #   ifndef STRNCASECMP
71 #       define STRNCASECMP strncasecmp
72 #   endif
73 #   ifndef STRCPY
74 #       define STRCPY(sFilePath, nLength, sPath) strcpy(sFilePath, sPath)
75 #   endif
76
77 #   ifndef FOPEN
78 #       define FOPEN(fHandle, filename, mode) (fHandle = fopen(filename, mode))
79 #   endif
80 #   ifndef FOPEN_FAIL
81 #       define FOPEN_FAIL(result) (result == NULL)
82 #   endif
83 #   ifndef SSCANF
84 #       define SSCANF sscanf
85 #   endif
```

```
86 #   ifndef SPRINTF
87 #       define SPRINTF sprintf
88 #   endif
89 #endif
90
91 #ifndef EXIT_WAIVED
92 #   define EXIT_WAIVED 2
93 #endif
94
95 // CUDA Utility Helper Functions
96 inline int stringRemoveDelimiter(char delimiter, const char *string) {
97     int string_start = 0;
98
99     while (string[string_start] == delimiter) { string_start++; }
100
101     if (string_start >= static_cast<int>(strlen(string) - 1)) { return 0; }
102
103     return string_start;
104 }
105
106 inline int getFileExtension(char *filename, char **extension) {
107     int string_length = static_cast<int>(strlen(filename));
108
109     while (filename[string_length--] != '.') {
110         if (string_length == 0) break;
111     }
112
113     if (string_length > 0) string_length += 2;
114
115     if (string_length == 0)
116         *extension = NULL;
117     else
118         *extension = &filename[string_length];
119
120     return string_length;
121 }
122
123 inline bool checkCmdLineFlag(const int argc, const char **argv, const char *string_ref) {
124     bool bFound = false;
125
126     if (argc >= 1) {
127         for (int i = 1; i < argc; i++) {
128             int string_start        = stringRemoveDelimiter('-', argv[i]);
129             const char *string_argv = &argv[i][string_start];
130
131             const char *equal_pos = strchr(string_argv, '=');
132             int argv_length =
133               static_cast<int>(equal_pos == 0 ? strlen(string_argv) : equal_pos - string_argv);
134
135             int length = static_cast<int>(strlen(string_ref));
136
137             if (length == argv_length && !STRNCASECMP(string_argv, string_ref, length)) {
138                 bFound = true;
139                 continue;
140             }
141         }
142     }
143
144     return bFound;
145 }
146
147 // This function wraps the CUDA Driver API into a template function
148 template<class T>
149 inline bool getCmdLineArgumentValue(const int argc, const char **argv, const char *string_ref,
150                                     T *value) {
151     bool bFound = false;
152
153     if (argc >= 1) {
154         for (int i = 1; i < argc; i++) {
155             int string_start        = stringRemoveDelimiter('-', argv[i]);
156             const char *string_argv = &argv[i][string_start];
157             int length              = static_cast<int>(strlen(string_ref));
158
159             if (!STRNCASECMP(string_argv, string_ref, length)) {
160                 if (length + 1 <= static_cast<int>(strlen(string_argv))) {
161                     int auto_inc = (string_argv[length] == '=') ? 1 : 0;
162                     *value       = (T)atoi(&string_argv[length + auto_inc]);
163                 }
164
165                 bFound = true;
166                 i      = argc;
167             }
168         }
169     }
170
171     return bFound;
172 }
```

```
173
174 inline int getCmdLineArgumentInt(const int argc, const char **argv, const char *string_ref) {
175     bool bFound = false;
176     int value   = -1;
177
178     if (argc >= 1) {
179         for (int i = 1; i < argc; i++) {
180             int string_start        = stringRemoveDelimiter('-', argv[i]);
181             const char *string_argv = &argv[i][string_start];
182             int length              = static_cast<int>(strlen(string_ref));
183
184             if (!STRNCASECMP(string_argv, string_ref, length)) {
185                 if (length + 1 <= static_cast<int>(strlen(string_argv))) {
186                     int auto_inc = (string_argv[length] == '=') ? 1 : 0;
187                     value        = atoi(&string_argv[length + auto_inc]);
188                 } else {
189                     value = 0;
190                 }
191
192                 bFound = true;
193                 continue;
194             }
195         }
196     }
197
198     if (bFound) {
199         return value;
200     } else {
201         return 0;
202     }
203 }
204
205 inline float getCmdLineArgumentFloat(const int argc, const char **argv, const char *string_ref) {
206     bool bFound = false;
207     float value = -1;
208
209     if (argc >= 1) {
210         for (int i = 1; i < argc; i++) {
211             int string_start        = stringRemoveDelimiter('-', argv[i]);
212             const char *string_argv = &argv[i][string_start];
213             int length              = static_cast<int>(strlen(string_ref));
214
215             if (!STRNCASECMP(string_argv, string_ref, length)) {
216                 if (length + 1 <= static_cast<int>(strlen(string_argv))) {
217                     int auto_inc = (string_argv[length] == '=') ? 1 : 0;
218                     value        = static_cast<float>(atof(&string_argv[length + auto_inc]));
219                 } else {
220                     value = 0.f;
221                 }
222
223                 bFound = true;
224                 continue;
225             }
226         }
227     }
228
229     if (bFound) {
230         return value;
231     } else {
232         return 0;
233     }
234 }
235
236 inline bool getCmdLineArgumentString(const int argc, const char **argv, const char *string_ref,
237                                      char **string_retval) {
238     bool bFound = false;
239
240     if (argc >= 1) {
241         for (int i = 1; i < argc; i++) {
242             int string_start  = stringRemoveDelimiter('-', argv[i]);
243             char *string_argv = const_cast<char *>(&argv[i][string_start]);
244             int length        = static_cast<int>(strlen(string_ref));
245
246             if (!STRNCASECMP(string_argv, string_ref, length)) {
247                 *string_retval = &string_argv[length + 1];
248                 bFound         = true;
249                 continue;
250             }
251         }
252     }
253
254     if (!bFound) { *string_retval = NULL; }
255
256     return bFound;
257 }
258
267 inline char *sdkFindFilePath(const char *filename, const char *executable_path) {
```

```
268     // <executable_name> defines a variable that is replaced with the name of
269     // the executable
270
271     // Typical relative search paths to locate needed companion files (e.g.
272     // sample input data, or JIT source files) The origin for the relative
273     // search may be the .exe file, a .bat file launching an .exe, a browser
274     // .exe launching the .exe or .bat, etc
275     const char *searchPath[] = {
276       "./",                                          // same dir
277       "./data/",                                     // same dir
278       "../../../../Samples/<executable_name>/",      // up 4 in tree
279       "../../../Samples/<executable_name>/",         // up 3 in tree
280       "../../Samples/<executable_name>/",            // up 2 in tree
281       "../../../../Samples/<executable_name>/data/", // up 4 in tree
282       "../../../Samples/<executable_name>/data/",    // up 3 in tree
283       "../../Samples/<executable_name>/data/",       // up 2 in tree
284       "../../../../Common/data/",                    // up 4 in tree
285       "../../../Common/data/",                       // up 3 in tree
286       "../../Common/data/"                           // up 2 in tree
287     };
288
289     // Extract the executable name
290     std::string executable_name;
291
292     if (executable_path != 0) {
293         executable_name = std::string(executable_path);
294
295 #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
296         // Windows path delimiter
297         size_t delimiter_pos = executable_name.find_last_of('\\');
298         executable_name.erase(0, delimiter_pos + 1);
299
300         if (executable_name.rfind(".exe") != std::string::npos) {
301             // we strip .exe, only if the .exe is found
302             executable_name.resize(executable_name.size() - 4);
303         }
304
305 #else
306         // Linux & OSX path delimiter
307         size_t delimiter_pos = executable_name.find_last_of('/');
308         executable_name.erase(0, delimiter_pos + 1);
309 #endif
310     }
311
312     // Loop over all search paths and return the first hit
313     for (unsigned int i = 0; i < sizeof(searchPath) / sizeof(char *); ++i) {
314         std::string path(searchPath[i]);
315         size_t executable_name_pos = path.find("<executable_name>");
316
317         // If there is executable_name variable in the searchPath
318         // replace it with the value
319         if (executable_name_pos != std::string::npos) {
320             if (executable_path != 0) {
321                 path.replace(executable_name_pos, strlen("<executable_name>"), executable_name);
322             } else {
323                 // Skip this path entry if no executable argument is given
324                 continue;
325             }
326         }
327
328 #ifdef _DEBUG
329         printf("sdkFindFilePath <%s> in %s\n", filename, path.c_str());
330 #endif
331
332         // Test if the file exists
333         path.append(filename);
334         FILE *fp;
335         FOPEN(fp, path.c_str(), "rb");
336
337         if (fp != NULL) {
338             fclose(fp);
339             // File found
340             // returning an allocated array here for backwards compatibility
341             // reasons
342             char *file_path = reinterpret_cast<char *>(malloc(path.length() + 1));
343             STRCPY(file_path, path.length() + 1, path.c_str());
344             return file_path;
345         }
346
347         if (fp) { fclose(fp); }
348     }
349
350     // File not found
351     return 0;
352 }
353
354 #endif // COMMON_HELPER_STRING_H_
```

7.177 helper_timer.h

```
1 /* Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
2  *
3  * Redistribution and use in source and binary forms, with or without
4  * modification, are permitted provided that the following conditions
5  * are met:
6  *  * Redistributions of source code must retain the above copyright
7  *    notice, this list of conditions and the following disclaimer.
8  *  * Redistributions in binary form must reproduce the above copyright
9  *    notice, this list of conditions and the following disclaimer in the
10 *    documentation and/or other materials provided with the distribution.
11 *  * Neither the name of NVIDIA CORPORATION nor the names of its
12 *    contributors may be used to endorse or promote products derived
13 *    from this software without specific prior written permission.
14 *
15 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS" AND ANY
16 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
17 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
18 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
19 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
20 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
21 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
22 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
23 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
24 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
25 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26 */
27
28 // Helper Timing Functions
29 #ifndef COMMON_HELPER_TIMER_H_
30 #define COMMON_HELPER_TIMER_H_
31
32 #ifndef EXIT_WAIVED
33 #   define EXIT_WAIVED 2
34 #endif
35
36 // includes, system
37 #include <vector>
38
39 // includes, project
40 #include "exception.h"
41
42 // Definition of the StopWatch Interface, this is used if we don't want to use
43 // the CUT functions But rather in a self contained class interface
44 class StopWatchInterface {
45 public:
46     StopWatchInterface() {}
47
48     virtual ~StopWatchInterface() {}
49
50 public:
52     virtual void start() = 0;
53
55     virtual void stop() = 0;
56
58     virtual void reset() = 0;
59
63     virtual float getTime() = 0;
64
67     virtual float getAverageTime() = 0;
68 };
69
71 // Begin Stopwatch timer class definitions for all OS platforms //
73 #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
74 // includes, system
75 #   define WINDOWS_LEAN_AND_MEAN
76
77 #   include <Windows.h>
78
79 #   undef min
80 #   undef max
81
83 class StopWatchWin : public StopWatchInterface {
84 public:
86     StopWatchWin() :
87             start_time(), end_time(), diff_time(0.0f), total_time(0.0f), running(false),
88             clock_sessions(0), freq(0), freq_set(false) {
89         if (!freq_set) {
90             // helper variable
91             LARGE_INTEGER temp;
92
93             // get the tick frequency from the OS
94             QueryPerformanceFrequency(reinterpret_cast<LARGE_INTEGER *>(&temp));
95
96             // convert to type in which it is needed
97             freq = (static_cast<double>(temp.QuadPart)) / 1000.0;
```

```
98
99              // rememeber query
100              freq_set = true;
101          }
102      }
103
104      // Destructor
105      ~StopWatchWin() {}
106
107  public:
109      inline void start();
110
112      inline void stop();
113
115      inline void reset();
116
120      inline float getTime();
121
124      inline float getAverageTime();
125
126  private:
127      // member variables
128
130      LARGE_INTEGER start_time;
132      LARGE_INTEGER end_time;
133
135      float diff_time;
136
138      float total_time;
139
141      bool running;
142
145      int clock_sessions;
146
148      double freq;
149
151      bool freq_set;
152  };
153
154  // functions, inlined
155
159  inline void StopWatchWin::start() {
160      QueryPerformanceCounter(reinterpret_cast<LARGE_INTEGER *>(&start_time));
161      running = true;
162  }
163
168  inline void StopWatchWin::stop() {
169      QueryPerformanceCounter(reinterpret_cast<LARGE_INTEGER *>(&end_time));
170      diff_time = static_cast<float>(
171        ((static_cast<double>(end_time.QuadPart) - static_cast<double>(start_time.QuadPart)) / freq));
172
173      total_time += diff_time;
174      clock_sessions++;
175      running = false;
176  }
177
182  inline void StopWatchWin::reset() {
183      diff_time      = 0;
184      total_time     = 0;
185      clock_sessions = 0;
186
187      if (running) { QueryPerformanceCounter(reinterpret_cast<LARGE_INTEGER *>(&start_time)); }
188  }
189
196  inline float StopWatchWin::getTime() {
197      // Return the TOTAL time to date
198      float retval = total_time;
199
200      if (running) {
201          LARGE_INTEGER temp;
202          QueryPerformanceCounter(reinterpret_cast<LARGE_INTEGER *>(&temp));
203          retval += static_cast<float>(
204            ((static_cast<double>(temp.QuadPart) - static_cast<double>(start_time.QuadPart)) / freq));
205      }
206
207      return retval;
208  }
209
214  inline float StopWatchWin::getAverageTime() {
215      return (clock_sessions > 0) ? (total_time / clock_sessions) : 0.0f;
216  }
217
218  #else
219  // Declarations for Stopwatch on Linux and Mac OSX
220  // includes, system
221  #   include <ctime>
222  #   include <sys/time.h>
```

```
223
225 class StopWatchLinux : public StopWatchInterface {
226 public:
228     StopWatchLinux() :
229             start_time(), diff_time(0.0), total_time(0.0), running(false), clock_sessions(0) {}
230
231     // Destructor
232     virtual ~StopWatchLinux() {}
233
234 public:
236     inline void start();
237
239     inline void stop();
240
242     inline void reset();
243
247     inline float getTime();
248
251     inline float getAverageTime();
252
253 private:
254     // helper functions
255
257     inline float getDiffTime();
258
259 private:
260     // member variables
261
263     struct timeval start_time;
264
266     float diff_time;
267
269     float total_time;
270
272     bool running;
273
276     int clock_sessions;
277 };
278
279 // functions, inlined
280
284 inline void StopWatchLinux::start() {
285     gettimeofday(&start_time, 0);
286     running = true;
287 }
288
293 inline void StopWatchLinux::stop() {
294     diff_time = getDiffTime();
295     total_time += diff_time;
296     running = false;
297     clock_sessions++;
298 }
299
304 inline void StopWatchLinux::reset() {
305     diff_time      = 0;
306     total_time     = 0;
307     clock_sessions = 0;
308
309     if (running) { gettimeofday(&start_time, 0); }
310 }
311
318 inline float StopWatchLinux::getTime() {
319     // Return the TOTAL time to date
320     float retval = total_time;
321
322     if (running) { retval += getDiffTime(); }
323
324     return retval;
325 }
326
331 inline float StopWatchLinux::getAverageTime() {
332     return (clock_sessions > 0) ? (total_time / clock_sessions) : 0.0f;
333 }
335
337 inline float StopWatchLinux::getDiffTime() {
338     struct timeval t_time;
339     gettimeofday(&t_time, 0);
340
341     // time difference in milli-seconds
342     return static_cast<float>(1000.0 * (t_time.tv_sec - start_time.tv_sec) +
343                               (0.001 * (t_time.tv_usec - start_time.tv_usec)));
344 }
345 #endif // WIN32
346
349
355 inline bool sdkCreateTimer(StopWatchInterface **timer_interface) {
356 // printf("sdkCreateTimer called object %08x\n", (void *)*timer_interface);
```

```
357 #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
358     *timer_interface = reinterpret_cast<StopWatchInterface *>(new StopWatchWin());
359 #else
360     *timer_interface = reinterpret_cast<StopWatchInterface *>(new StopWatchLinux());
361 #endif
362     return (*timer_interface != NULL) ? true : false;
363 }
364
370 inline bool sdkDeleteTimer(StopWatchInterface **timer_interface) {
371     // printf("sdkDeleteTimer called object %08x\n", (void *)*timer_interface);
372     if (*timer_interface) {
373         delete *timer_interface;
374         *timer_interface = NULL;
375     }
376
377     return true;
378 }
379
384 inline bool sdkStartTimer(StopWatchInterface **timer_interface) {
385     // printf("sdkStartTimer called object %08x\n", (void *)*timer_interface);
386     if (*timer_interface) { (*timer_interface)->start(); }
387
388     return true;
389 }
390
395 inline bool sdkStopTimer(StopWatchInterface **timer_interface) {
396     // printf("sdkStopTimer called object %08x\n", (void *)*timer_interface);
397     if (*timer_interface) { (*timer_interface)->stop(); }
398
399     return true;
400 }
401
406 inline bool sdkResetTimer(StopWatchInterface **timer_interface) {
407     // printf("sdkResetTimer called object %08x\n", (void *)*timer_interface);
408     if (*timer_interface) { (*timer_interface)->reset(); }
409
410     return true;
411 }
412
420 inline float sdkGetAverageTimerValue(StopWatchInterface **timer_interface) {
421     //  printf("sdkGetAverageTimerValue called object %08x\n", (void
422     //  *)*timer_interface);
423     if (*timer_interface) {
424         return (*timer_interface)->getAverageTime();
425     } else {
426         return 0.0f;
427     }
428 }
429
435 inline float sdkGetTimerValue(StopWatchInterface **timer_interface) {
436     // printf("sdkGetTimerValue called object %08x\n", (void
437     // *)*timer_interface);
438     if (*timer_interface) {
439         return (*timer_interface)->getTime();
440     } else {
441         return 0.0f;
442     }
443 }
444
445 #endif // COMMON_HELPER_TIMER_H_
```

7.178 kernel_header.h

```
1 #pragma once
2
3 #include <string>
4
5 namespace librapid::imp {
6     inline const jitify::detail::vector<std::string> cudaHeaders = { // CUDA_INCLUDE_DIRS,
7       CUDA_INCLUDE_DIRS + std::string("/curand.h"),
8       CUDA_INCLUDE_DIRS + std::string("/curand_kernel.h"),
9       CUDA_INCLUDE_DIRS + std::string("/cublas_v2.h"),
10       CUDA_INCLUDE_DIRS + std::string("/cublas_api.h"),
11       CUDA_INCLUDE_DIRS + std::string("/cuda_fp16.h"),
12       CUDA_INCLUDE_DIRS + std::string("/cuda_bf16.h")};
13
14     inline const std::vector<std::string> cudaParams = {
15       "--disable-warnings", "-std=c++17", std::string("-I") + CUDA_INCLUDE_DIRS};
16
17     inline std::string genKernelHeader() {
18         return fmt::format(R"V0G0N(
19 #include <"{0}/curand_kernel.h>
20 #include <"{0}"/curand.h>
```

```
21 #include <stdint.h>
22 #include <type_traits>
23
24 #ifndef LIBRAPID_CUSTOM_COMPLEX
25 #define LIBRAPID_CUSTOM_COMPLEX
26
27 namespace librapid {{
28
29     template<class T>
30     class Complex {{
31     public:
32         Complex(const T &real_val = T(), const T &imag_val = T())
33                 : m_real(real_val), m_imag(imag_val) {{}}
34
35         Complex &operator=(const T &val) {{
36             m_real = val;
37             m_imag = 0;
38             return *this;
39         }}
40
41         template<class V>
42         Complex(const Complex<V> &other)
43                 : Complex(static_cast<T>(other.real()), static_cast<T>(other.imag())) {{}}
44
45         template<class V>
46         Complex &operator=(const Complex<V> &other) {{
47             m_real = static_cast<T>(other.real());
48             m_imag = static_cast<T>(other.imag());
49             return *this;
50         }}
51
52         Complex copy() const {{
53             return Complex<T>(m_real, m_imag);
54         }}
55
56         inline Complex operator-() const {{
57             return Complex<T>(-m_real, -m_imag);
58         }}
59
60         template<typename V, typename std::enable_if<std::is_scalar<V>::value, int>::type = 0>
61         inline Complex operator+(const V &other) const {{
62             return Complex<T>(m_real + other, m_imag);
63         }}
64
65         template<typename V, typename std::enable_if<std::is_scalar<V>::value, int>::type = 0>
66         inline Complex operator-(const V &other) const {{
67             return Complex<T>(m_real - other, m_imag);
68         }}
69
70         template<typename V, typename std::enable_if<std::is_scalar<V>::value, int>::type = 0>
71         inline Complex operator*(const V &other) const {{
72             return Complex<T>(m_real * other, m_imag * other);
73         }}
74
75         template<typename V, typename std::enable_if<std::is_scalar<V>::value, int>::type = 0>
76         inline Complex operator/(const V &other) const {{
77             return Complex<T>(m_real / other, m_imag / other);
78         }}
79
80         template<typename V, typename std::enable_if<std::is_scalar<V>::value, int>::type = 0>
81         inline Complex &operator+=(const V &other) {{
82             m_real += other;
83             return *this;
84         }}
85
86         template<typename V, typename std::enable_if<std::is_scalar<V>::value, int>::type = 0>
87         inline Complex &operator-=(const V &other) {{
88             m_real -= other;
89             return *this;
90         }}
91
92         template<typename V, typename std::enable_if<std::is_scalar<V>::value, int>::type = 0>
93         inline Complex &operator*=(const V &other) {{
94             m_real *= other;
95             m_imag *= other;
96             return *this;
97         }}
98
99         template<typename V, typename std::enable_if<std::is_scalar<V>::value, int>::type = 0>
100         inline Complex &operator/=(const V &other) {{
101             m_real /= other;
102             m_imag /= other;
103             return *this;
104         }}
105
106         template<typename V>
107         inline Complex operator+(const Complex<V> &other) const {{
```

```
108             return Complex(m_real + other.real(),
109                            m_imag + other.imag());
110         }}
111 
112         template<typename V>
113         inline Complex operator-(const Complex<V> &other) const {{
114             return Complex(m_real - other.real(),
115                            m_imag - other.imag());
116         }}
117 
118         template<typename V>
119         inline Complex operator*(const Complex<V> &other) const {{
120             return Complex((m_real * other.real()) - (m_imag * other.imag()),
121                            (m_real * other.imag()) + (m_imag * other.real()));
122         }}
123 
124         template<typename V>
125         inline Complex operator/(const Complex<V> &other) const {{
126             return Complex((m_real * other.real()) + (m_imag * other.imag()) /
127                                                      ((other.real() * other.real()) + (other.imag() *
    other.imag())),
128                            (m_real * other.real()) - (m_imag * other.imag()) /
129                                                      ((other.real() * other.real()) + (other.imag() *
    other.imag())));
130         }}
131 
132         template<typename V>
133         inline Complex &operator+=(const Complex<V> &other) {{
134             m_real = m_real + other.real();
135             m_imag = m_imag + other.imag();
136             return *this;
137         }}
138 
139         template<typename V>
140         inline Complex &operator-=(const Complex<V> &other) {{
141             m_real = m_real - other.real();
142             m_imag = m_imag - other.imag();
143             return *this;
144         }}
145 
146         template<typename V>
147         inline Complex &operator*=(const Complex<V> &other) {{
148             m_real = (m_real * other.real()) - (m_imag * other.imag());
149             m_imag = (m_real * other.imag()) + (imag() * other.real());
150             return *this;
151         }}
152 
153         template<typename V>
154         inline Complex &operator/=(const Complex<V> &other) {{
155             m_real = (m_real * other.real()) + (m_imag * other.imag()) /
156                                                ((other.real() * other.real()) + (other.imag() *
    other.imag()));
157             m_imag = (m_real * other.real()) - (m_imag * other.imag()) /
158                                                ((other.real() * other.real()) + (other.imag() *
    other.imag()));
159             return *this;
160         }}
161 
162         template<typename V>
163         inline bool operator==(const Complex<V> &other) const {{
164             return m_real == other.real() && m_imag == other.imag();
165         }}
166 
167         template<typename V>
168         inline bool operator!=(const Complex<V> &other) const {{
169             return !(*this == other);
170         }}
171 
172         template<typename V>
173         inline bool operator==(const V &other) const {{
174             return m_real == other && m_imag == 0;
175         }}
176 
177         template<typename V>
178         inline bool operator!=(const V &other) const {{
179             return !(*this == other);
180         }}
181 
182         inline T mag() const {{
183             return std::sqrt(m_real * m_real + m_imag * m_imag);
184         }}
185 
186         inline T angle() const {{
187             return std::atan2(m_real, m_imag);
188         }}
189 
190         inline Complex<T> log() const {{
```

```
191             return Complex<T>(std::log(mag()), angle());
192         }}
193
194         inline Complex<T> conjugate() const {{
195             return Complex<T>(m_real, -m_imag);
196         }}
197
198         inline Complex<T> reciprocal() const {{
199             return Complex<T>((m_real) / (m_real * m_real + m_imag * m_imag),
200                               -(m_imag) / (m_real * m_real + m_imag * m_imag));
201         }}
202
203         inline const T &real() const {{
204             return m_real;
205         }}
206
207         inline T &real() {{
208             return m_real;
209         }}
210
211         inline const T &imag() const {{
212             return m_imag;
213         }}
214
215         inline T &imag() {{
216             return m_imag;
217         }}
218
219         inline explicit operator std::string() const {{
220             return str();
221         }}
222
223         template<typename V>
224         inline operator V() const {{
225             return m_real;
226         }}
227
228         template<typename V>
229         inline explicit operator std::complex<V>() const {{
230             return std::complex<V>(m_real, m_imag);
231         }}
232
233     private:
234         T m_real = 0;
235         T m_imag = 0;
236     }};
237
238     template<typename A, typename B, typename std::enable_if<std::is_scalar<A>::value, int>::type = 0>
239     Complex<B> operator+(const A &a, const Complex<B> &b) {{
240         return Complex<B>(a) + b;
241     }}
242
243     template<typename A, typename B, typename std::enable_if<std::is_scalar<A>::value, int>::type = 0>
244     inline Complex<B> operator-(const A &a, const Complex<B> &b) {{
245         return Complex<B>(a) - b;
246     }}
247
248     template<typename A, typename B, typename std::enable_if<std::is_scalar<A>::value, int>::type = 0>
249     inline Complex<B> operator*(const A &a, const Complex<B> &b) {{
250         return Complex<B>(a) * b;
251     }}
252
253     template<typename A, typename B, typename std::enable_if<std::is_scalar<A>::value, int>::type = 0>
254     inline Complex<B> operator/(const A &a, const Complex<B> &b) {{
255         return Complex<B>(a) / b;
256     }}
257
258     template<typename A, typename B, typename std::enable_if<std::is_scalar<A>::value, int>::type = 0>
259     inline A &operator+=(A &a, const Complex<B> &b) {{
260         a += b.real();
261         return a;
262     }}
263
264     template<typename A, typename B, typename std::enable_if<std::is_scalar<A>::value, int>::type = 0>
265     inline A &operator-=(A &a, const Complex<B> &b) {{
266         a -= b.real();
267         return a;
268     }}
269
270     template<typename A, typename B, typename std::enable_if<std::is_scalar<A>::value, int>::type = 0>
271     inline A &operator*=(A &a, const Complex<B> &b) {{
272         a *= b.real();
273         return a;
274     }}
275
276     template<typename A, typename B, typename std::enable_if<std::is_scalar<A>::value, int>::type = 0>
277     inline A &operator/=(A &a, const Complex<B> &b) {{
```

```
278         a /= b.real();
279         return a;
280     }}
281
282 }}
283
284 #endif // LIBRAPID_CUSTOM_COMPLEX
285         )V0G0N",
286                             CUDA_INCLUDE_DIRS);
287     }
288 } // namespace librapid::imp
```

7.179 nvrtc_helper.h

```
1 /* Copyright (c) 2019, NVIDIA CORPORATION. All rights reserved.
2  *
3  * Redistribution and use in source and binary forms, with or without
4  * modification, are permitted provided that the following conditions
5  * are met:
6  *  * Redistributions of source code must retain the above copyright
7  *    notice, this list of conditions and the following disclaimer.
8  *  * Redistributions in binary form must reproduce the above copyright
9  *    notice, this list of conditions and the following disclaimer in the
10 *    documentation and/or other materials provided with the distribution.
11 *  * Neither the name of NVIDIA CORPORATION nor the names of its
12 *    contributors may be used to endorse or promote products derived
13 *    from this software without specific prior written permission.
14 *
15 * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS "AS IS" AND ANY
16 * EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
17 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
18 * PURPOSE ARE DISCLAIMED.  IN NO EVENT SHALL THE COPYRIGHT OWNER OR
19 * CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
20 * EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
21 * PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
22 * PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY
23 * OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
24 * (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
25 * OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
26 */
27
28 #ifndef COMMON_NVRTC_HELPER_H_
29
30 #define COMMON_NVRTC_HELPER_H_ 1
31
32 #include <cuda.h>
33 #include <fstream>
34 #include "helper_cuda_drvapi.h"
35 #include <iostream>
36 #include <nvrtc.h>
37 #include <sstream>
38 #include <string>
39
40 #define NVRTC_SAFE_CALL(Name, x)                                                \
41     do {                                                                        \
42         nvrtcResult result = x;                                                 \
43         if (result != NVRTC_SUCCESS) {                                          \
44             std::cerr « "\nerror: " « Name « " failed with error "             \
45                       « nvrtcGetErrorString(result);                            \
46             exit(1);                                                            \
47         }                                                                       \
48     } while (0)
49
50 void compileFileToCUBIN(char *filename, int argc, char **argv, char **cubinResult,
51                         size_t *cubinResultSize, int requiresCGheaders) {
52     std::ifstream inputFile(filename, std::ios::in | std::ios::binary | std::ios::ate);
53
54     if (!inputFile.is_open()) {
55         std::cerr « "\nerror: unable to open " « filename « " for reading!\n";
56         exit(1);
57     }
58
59     std::streampos pos = inputFile.tellg();
60     size_t inputSize   = (size_t)pos;
61     char *memBlock     = new char[inputSize + 1];
62
63     inputFile.seekg(0, std::ios::beg);
64     inputFile.read(memBlock, inputSize);
65     inputFile.close();
66     memBlock[inputSize] = '\x0';
67
68     int numCompileOptions = 0;
69
```

```
70     char *compileParams[2];
71
72     int major = 0, minor = 0;
73     char deviceName[256];
74
75     // Picks the best CUDA device available
76     CUdevice cuDevice = findCudaDeviceDRV(argc, (const char **)argv);
77
78     // get compute capabilities and the devicename
79     checkCudaErrors(
80       cuDeviceGetAttribute(&major, CU_DEVICE_ATTRIBUTE_COMPUTE_CAPABILITY_MAJOR, cuDevice));
81     checkCudaErrors(
82       cuDeviceGetAttribute(&minor, CU_DEVICE_ATTRIBUTE_COMPUTE_CAPABILITY_MINOR, cuDevice));
83
84     {
85         // Compile cubin for the GPU arch on which are going to run cuda kernel.
86         std::string compileOptions;
87         compileOptions = "--gpu-architecture=sm_";
88
89         compileParams[numCompileOptions] =
90           reinterpret_cast<char *>(malloc(sizeof(char) * (compileOptions.length() + 10)));
91 #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
92         sprintf_s(compileParams[numCompileOptions],
93                   sizeof(char) * (compileOptions.length() + 10),
94                   "%s%d%d",
95                   compileOptions.c_str(),
96                   major,
97                   minor);
98 #else
99         snprintf(compileParams[numCompileOptions],
100                  compileOptions.size() + 10,
101                  "%s%d%d",
102                  compileOptions.c_str(),
103                  major,
104                  minor);
105 #endif
106     }
107
108     numCompileOptions++;
109
110     if (requiresCGheaders) {
111         std::string compileOptions;
112         char HeaderNames[256];
113 #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
114         sprintf_s(HeaderNames, sizeof(HeaderNames), "%s", "cooperative_groups.h");
115 #else
116         snprintf(HeaderNames, sizeof(HeaderNames), "%s", "cooperative_groups.h");
117 #endif
118
119         compileOptions = "--include-path=";
120
121         std::string path = sdkFindFilePath(HeaderNames, argv[0]);
122         if (!path.empty()) {
123             std::size_t found = path.find(HeaderNames);
124             path.erase(found);
125         } else {
126             printf(
127               "\nCooperativeGroups headers not found, please install it in %s "
128               "sample directory..\n Exiting..\n",
129               argv[0]);
130         }
131         compileOptions += path.c_str();
132         compileParams[numCompileOptions] =
133           reinterpret_cast<char *>(malloc(sizeof(char) * (compileOptions.length() + 1)));
134 #if defined(WIN32) || defined(_WIN32) || defined(WIN64) || defined(_WIN64)
135         sprintf_s(compileParams[numCompileOptions],
136                   sizeof(char) * (compileOptions.length() + 1),
137                   "%s",
138                   compileOptions.c_str());
139 #else
140         snprintf(
141           compileParams[numCompileOptions], compileOptions.size(), "%s", compileOptions.c_str());
142 #endif
143         numCompileOptions++;
144     }
145
146     // compile
147     nvrtcProgram prog;
148     NVRTC_SAFE_CALL("nvrtcCreateProgram",
149                     nvrtcCreateProgram(&prog, memBlock, filename, 0, NULL, NULL));
150
151     nvrtcResult res = nvrtcCompileProgram(prog, numCompileOptions, compileParams);
152
153     // dump log
154     size_t logSize;
155     NVRTC_SAFE_CALL("nvrtcGetProgramLogSize", nvrtcGetProgramLogSize(prog, &logSize));
156     char *log = reinterpret_cast<char *>(malloc(sizeof(char) * logSize + 1));
```

```
157     NVRTC_SAFE_CALL("nvrtcGetProgramLog", nvrtcGetProgramLog(prog, log));
158     log[logSize] = '\x0';
159 
160     if (strlen(log) >= 2) {
161         std::cerr << "\n compilation log ---\n";
162         std::cerr << log;
163         std::cerr << "\n end log ---\n";
164     }
165 
166     free(log);
167 
168     NVRTC_SAFE_CALL("nvrtcCompileProgram", res);
169 
170     size_t codeSize;
171     NVRTC_SAFE_CALL("nvrtcGetCUBINSize", nvrtcGetCUBINSize(prog, &codeSize));
172     char *code = new char[codeSize];
173     NVRTC_SAFE_CALL("nvrtcGetCUBIN", nvrtcGetCUBIN(prog, code));
174     *cubinResult     = code;
175     *cubinResultSize = codeSize;
176 
177     for (int i = 0; i < numCompileOptions; i++) { free(compileParams[i]); }
178 }
179 
180 CUmodule loadCUBIN(char *cubin, int argc, char **argv) {
181     CUmodule module;
182     CUcontext context;
183     int major = 0, minor = 0;
184     char deviceName[256];
185 
186     // Picks the best CUDA device available
187     CUdevice cuDevice = findCudaDeviceDRV(argc, (const char **)argv);
188 
189     // get compute capabilities and the devicename
190     checkCudaErrors(
191       cuDeviceGetAttribute(&major, CU_DEVICE_ATTRIBUTE_COMPUTE_CAPABILITY_MAJOR, cuDevice));
192     checkCudaErrors(
193       cuDeviceGetAttribute(&minor, CU_DEVICE_ATTRIBUTE_COMPUTE_CAPABILITY_MINOR, cuDevice));
194     checkCudaErrors(cuDeviceGetName(deviceName, 256, cuDevice));
195     printf("> GPU Device has SM %d.%d compute capability\n", major, minor);
196 
197     checkCudaErrors(cuInit(0));
198     checkCudaErrors(cuCtxCreate(&context, 0, cuDevice));
199 
200     checkCudaErrors(cuModuleLoadData(&module, cubin));
201     free(cubin);
202 
203     return module;
204 }
205 
206 #endif // COMMON_NVRTC_HELPER_H_
```

7.180 librapid.hpp

```
1 #ifndef LIBRAPID_HPP
2 #define LIBRAPID_HPP
3 
4 #include "core/core.hpp"
5 #include "array/array.hpp"
6 
7 #endif // LIBRAPID_HPP
```

7.181 coreMath.hpp

```
1 #ifndef LIBRAPID_MATH_CORE_MATH_HPP
2 #define LIBRAPID_MATH_CORE_MATH_HPP
3 
4 /*
5  * This file defines a wide range of core operations on many data types.
6  * Many of these functions will end up calling the C++ STL function for
7  * primitive types, though for types defined by LibRapid, custom implementations
8  * will be required.
9  */
10 
11 namespace librapid {
16     template<typename T>
17     T min(const T &val);
18 
23     template<typename First, typename... Rest>
24     auto min(const First &first, const Rest &...rest);
```

```
25 } // namespace librapid
26
27 #endif // LIBRAPID_MATH_CORE_MATH_HPP
```

Index