
hipCUB Documentation

Release 2.10.9

Advanced Micro Devices

Jun 28, 2022

CONTENTS:

1	Introduction	1
1.1	Overview	1
1.2	rocPRIM backend	1
2	Library API	3
2.1	Full API	3
2.1.1	Namespaces	3
2.1.2	Classes and Structs	9
2.1.3	Enums	136
2.1.4	Unions	141
2.1.5	Functions	143
2.1.6	Defines	164
2.1.7	Typedefs	169
3	Indices and tables	171
	Index	173

INTRODUCTION

1.1 Overview

hipCUB is a thin wrapper library on top of rocPRIM or CUB. It enables developers to port project using CUB library to the HIP layer and to run them on AMD hardware. In the ROCm environment, hipCUB uses rocPRIM library as the backend, however, on CUDA platforms it uses CUB instead.

- When using hipCUB you should only include `<hipcub/hipcub.hpp>` header.
- When rocPRIM is used as backend `HIPCUB_ROCPRIM_API` is defined.
- When CUB is used as backend `HIPCUB_CUB_API` is defined.
- Backends are automatically selected based on platform detected by HIP layer (`__HIP_PLATFORM_AMD__`, `__HIP_PLATFORM_NVIDIA__`).

1.2 rocPRIM backend

hipCUB with rocPRIM backend may not support all function and features CUB has because of the differences between ROCm (HIP) platform and CUDA platform.

Not-supported features and differences:

- Functions, classes and macros which are not in the public API or not documented are not supported.
- Device-wide primitives can't be called from kernels (dynamic parallelism is not supported in HIP on ROCm).
- `DeviceSpmv` is not supported.
- Fancy iterators: `CacheModifiedInputIterator`, `CacheModifiedOutputIterator`, and `TexRefInputIterator` are not supported.
- Thread I/O:
 - `CacheLoadModifier`, `CacheStoreModifier` cache modifiers are not supported.
 - `ThreadLoad`, `ThreadStore` functions are not supported.
- Storage management and debug functions:
 - `Debug`, `PtxVersion`, `SmVersion` functions and `CubDebug`, `CubDebugExit`, `_CubLog` macros are not supported.
- Intrinsic:
 - `ThreadExit`, `ThreadTrap` - not supported.
 - Warp thread masks (when used) are 64-bit unsigned integers.

- member_mask input argument is ignored in WARP_* functions.
- Arguments first_lane, last_lane, and member_mask are ignored in Shuffle* functions.
- Utilities:
 - SwizzleScanOp, ReduceBySegmentOp, ReduceByKeyOp, CastOp - not supported.

LIBRARY API

2.1 Full API

2.1.1 Namespaces

Namespace detail

Contents

- *Functions*

Functions

- *Function detail::to_BlockHistogramAlgorithm_enum*
- *Function detail::to_BlockReduceAlgorithm_enum*

Namespace hipcub

Contents

- *Namespaces*
- *Classes*
- *Enums*
- *Functions*
- *Typedefs*
- *Unions*

Namespaces

- *Namespace hipcub::detail*
- *Namespace hipcub::internal*

Classes

- *Struct ArgMax*
- *Struct ArgMin*
- *Template Struct BaseDigitExtractor*
- *Template Struct BFEDigitExtractor*
- *Template Struct block_raking_layout*
- *Struct block_raking_layout::TempStorage*
- *Struct BlockMergeSortStrategy::TempStorage*
- *Struct BlockRadixRank::PrefixCallBack*
- *Struct BlockRadixRank::TempStorage*
- *Struct BlockRadixRankMatch::TempStorage*
- *Struct BlockRunLengthDecode::TempStorage*
- *Struct CacheModifiedOutputIterator::Reference*
- *Struct CachingDeviceAllocator*
- *Struct CachingDeviceAllocator::BlockDescriptor*
- *Struct DeviceHistogram*
- *Struct DeviceMergeSort*
- *Struct DevicePartition*
- *Struct DeviceRadixSort*
- *Struct DeviceSegmentedRadixSort*
- *Struct DeviceSegmentedReduce*
- *Struct DeviceSegmentedSort*
- *Template Struct DeviceSpmv::SpmvParams*
- *Template Struct DoubleBuffer*
- *Struct Equality*
- *Template Struct GridEvenShare*
- *Template Struct If*
- *Struct Inequality*
- *Template Struct InequalityWrapper*
- *Template Struct Int2Type*
- *Template Struct IsPointer*

- *Template Struct IsVolatile*
- *Template Struct Log2*
- *Struct Max*
- *Struct Min*
- *Template Struct PowerOfTwo*
- *Template Struct RadixSortTwiddle*
- *Template Struct RemoveQualifiers*
- *Template Struct ShiftDigitExtractor*
- *Struct Sum*
- *Template Struct Uninitialized*
- *Struct WarpExchange::TempStorage*
- *Template Struct WarpLoad::LoadInternal*
- *Template Struct WarpLoad::LoadInternal< WARP_LOAD_DIRECT >*
- *Template Struct WarpLoad::LoadInternal< WARP_LOAD_STRIPED >*
- *Template Struct WarpLoad::LoadInternal< WARP_LOAD_TRANSPOSE >*
- *Template Struct WarpLoad::LoadInternal< WARP_LOAD_VECTORIZE >*
- *Struct WarpLoad::TempStorage*
- *Template Struct WarpStore::StoreInternal*
- *Template Struct WarpStore::StoreInternal< WARP_STORE_DIRECT >*
- *Template Struct WarpStore::StoreInternal< WARP_STORE_STRIPED >*
- *Template Struct WarpStore::StoreInternal< WARP_STORE_TRANSPOSE >*
- *Template Struct WarpStore::StoreInternal< WARP_STORE_VECTORIZE >*
- *Struct WarpStore::TempStorage*
- *Template Class BlockAdjacentDifference*
- *Template Class BlockDiscontinuity*
- *Template Class BlockExchange*
- *Template Class BlockLoad*
- *Template Class BlockMergeSort*
- *Template Class BlockMergeSortStrategy*
- *Template Class BlockRadixRank*
- *Template Class BlockRadixRankMatch*
- *Template Class BlockRadixSort*
- *Template Class BlockRunLengthDecode*
- *Template Class BlockScan*
- *Template Class BlockShuffle*
- *Template Class BlockStore*

- *Template Class CacheModifiedInputIterator*
- *Template Class CacheModifiedOutputIterator*
- *Class CachingDeviceAllocator::TotalBytes*
- *Class DeviceReduce*
- *Class DeviceRunLengthEncode*
- *Class DeviceScan*
- *Class DeviceSelect*
- *Class DeviceSpmv*
- *Template Class DiscardOutputIterator*
- *Class GridBarrier*
- *Class GridBarrierLifetime*
- *Template Class GridQueue*
- *Template Class TexObjInputIterator*
- *Template Class TexRefInputIterator*
- *Template Class WarpExchange*
- *Template Class WarpLoad*
- *Template Class WarpMergeSort*
- *Template Class WarpReduce*
- *Template Class WarpScan*
- *Template Class WarpStore*

Enums

- *Enum BlockLoadAlgorithm*
- *Enum BlockScanAlgorithm*
- *Enum BlockStoreAlgorithm*
- *Enum GridMappingStrategy*
- *Enum WarpLoadAlgorithm*
- *Enum WarpStoreAlgorithm*

Functions

- *Function hipcub::BAR*
- *Template Function hipcub::BFE*
- *Function hipcub::BFI*
- *Function hipcub::CTA_SYNC*
- *Function hipcub::Debug*

- *Template Function hipcub::DivideAndRoundUp*
- *Function hipcub::IADD3*
- *Function hipcub::LaneId*
- *Function hipcub::LaneMaskGe*
- *Function hipcub::LaneMaskGt*
- *Function hipcub::LaneMaskLe*
- *Function hipcub::LaneMaskLt*
- *Template Function hipcub::LoadDirectBlocked(int, InputIteratorT, T(&))*
- *Template Function hipcub::LoadDirectBlocked(int, InputIteratorT, T(&), int)*
- *Template Function hipcub::LoadDirectBlocked(int, InputIteratorT, T(&), int, Default)*
- *Template Function hipcub::LoadDirectBlockedVectorized*
- *Template Function hipcub::LoadDirectStriped(int, InputIteratorT, T(&))*
- *Template Function hipcub::LoadDirectStriped(int, InputIteratorT, T(&), int)*
- *Template Function hipcub::LoadDirectStriped(int, InputIteratorT, T(&), int, Default)*
- *Template Function hipcub::LoadDirectWarpStriped(int, InputIteratorT, T(&))*
- *Template Function hipcub::LoadDirectWarpStriped(int, InputIteratorT, T(&), int)*
- *Template Function hipcub::LoadDirectWarpStriped(int, InputIteratorT, T(&), int, Default)*
- *Template Function hipcub::MergePath*
- *Function hipcub::PRMT*
- *Function hipcub::RowMajorTid*
- *Template Function hipcub::SerialMerge*
- *Function hipcub::SHL_ADD*
- *Function hipcub::SHR_ADD*
- *Template Function hipcub::ShuffleDown*
- *Template Function hipcub::ShuffleIndex*
- *Template Function hipcub::ShuffleUp*
- *Template Function hipcub::StableOddEvenSort*
- *Template Function hipcub::StoreDirectBlocked(int, OutputIteratorT, T(&))*
- *Template Function hipcub::StoreDirectBlocked(int, OutputIteratorT, T(&), int)*
- *Template Function hipcub::StoreDirectBlockedVectorized*
- *Template Function hipcub::StoreDirectStriped(int, OutputIteratorT, T(&))*
- *Template Function hipcub::StoreDirectStriped(int, OutputIteratorT, T(&), int)*
- *Template Function hipcub::StoreDirectWarpStriped(int, OutputIteratorT, T(&))*
- *Template Function hipcub::StoreDirectWarpStriped(int, OutputIteratorT, T(&), int)*
- *Template Function hipcub::Swap*
- *Function hipcub::WARP_ALL*

- *Function hipcub::WARP_ANY*
- *Function hipcub::WARP_BALLOT*
- *Function hipcub::WARP_SYNC*
- *Function hipcub::WarpId*
- *Template Function hipcub::WarpMask*

Typedefs

- *Typedef hipcub::FutureValue*

Unions

- *Union BlockMergeSortStrategy::_TempStorage*
- *Union BlockRunLengthDecode::_TempStorage*
- *Union WarpExchange::_TempStorage*

Namespace hipcub::detail

Contents

- *Functions*

Functions

- *Template Function hipcub::detail::convert_result_type*
- *Template Function hipcub::detail::get_lowest_value*
- *Specialized Template Function hipcub::detail::get_lowest_value< __half >*
- *Specialized Template Function hipcub::detail::get_lowest_value< hip_bfloat16 >*
- *Template Function hipcub::detail::get_max_value*
- *Specialized Template Function hipcub::detail::get_max_value< __half >*
- *Specialized Template Function hipcub::detail::get_max_value< hip_bfloat16 >*
- *Function hipcub::detail::to_BlockLoadAlgorithm_enum*
- *Function hipcub::detail::to_BlockScanAlgorithm_enum*
- *Function hipcub::detail::to_BlockStoreAlgorithm_enum*
- *Template Function hipcub::detail::to_double_buffer*
- *Template Function hipcub::detail::unsigned_bit_extract*
- *Template Function hipcub::detail::update_double_buffer*

Namespace `hipcub::internal`

Internal namespace (to prevent ADL mishaps between static functions when mixing different CUB installations)

Contents

- *Functions*

Functions

- *Function `hipcub::internal::ThreadScanExclusive`*
- *Function `hipcub::internal::ThreadScanInclusive`*

Namespace `internal`

Internal namespace (to prevent ADL mishaps between static functions when mixing different CUB installations)

Contents

- *Functions*

Functions

- *Template Function `internal::ThreadReduce(T *, ReductionOp, T)`*
- *Template Function `internal::ThreadReduce(T(&), ReductionOp, T)`*
- *Template Function `internal::ThreadReduce(T(&), ReductionOp)`*

2.1.2 Classes and Structs

Struct `ArgMax`

- Defined in `file_hipcub_backend_rocprim_thread_thread_operators.hpp`

Struct Documentation

```
struct hipcub::ArgMax
```

Public Functions

```
template<class Key, class Value>
__host__ __device__ inline constexpr KeyValuePair<Key, Value> operator() (const KeyValuePair<Key,
                                                                    Value> &a, const
                                                                    KeyValuePair<Key, Value>
                                                                    &b) const
```

Struct ArgMin

- Defined in file `hipcub_backend_rocprim_thread_thread_operators.hpp`

Struct Documentation

```
struct hipcub::ArgMin
```

Public Functions

```
template<class Key, class Value>
__host__ __device__ inline constexpr KeyValuePair<Key, Value> operator() (const KeyValuePair<Key,
                                                                    Value> &a, const
                                                                    KeyValuePair<Key, Value>
                                                                    &b) const
```

Template Struct BaseDigitExtractor

- Defined in file `hipcub_backend_rocprim_block_radix_rank_sort_operations.hpp`

Inheritance Relationships

Derived Types

- `public hipcub::BFEDigitExtractor< KeyT >` (*Template Struct BFEDigitExtractor*)
- `public hipcub::ShiftDigitExtractor< KeyT >` (*Template Struct ShiftDigitExtractor*)

Struct Documentation

```
template<typename KeyT>
```

```
struct hipcub::BaseDigitExtractor
```

Base struct for digit extractor. Contains common code to provide special handling for floating-point -0.0.

Note: This handles correctly both the case when the keys are bitwise-complemented after twiddling for descending sort (in onesweep) as well as when the keys are not bit-negated, but the implementation handles descending sort separately (in other implementations in CUB). Twiddling alone maps -0.0f to 0x7fffffff and +0.0f to 0x80000000 for float, which are subsequent bit patterns and bitwise complements of each other. For onesweep,

both -0.0f and +0.0f are mapped to the bit pattern of +0.0f (0x80000000) for ascending sort, and to the pattern of -0.0f (0x7fffffff) for descending sort. For all other sorting implementations in CUB, both are always mapped to +0.0f. Since bit patterns for both -0.0f and +0.0f are next to each other and only one of them is used, the sorting works correctly. For double, the same applies, but with 64-bit patterns.

Subclassed by *hipcub::BFEDigitExtractor< KeyT >*, *hipcub::ShiftDigitExtractor< KeyT >*

Public Types

enum **[anonymous]**

Values:

enumerator **FLOAT_KEY**

typedef Traits<KeyT> **TraitsT**

typedef TraitsT::UnsignedBits **UnsignedBits**

Public Static Functions

`__device__ __forceinline__ static inline UnsignedBits ProcessFloatMinusZero(UnsignedBits key)`

Template Struct BFEDigitExtractor

- Defined in file `hipcub_backend_rocprim_block_radix_rank_sort_operations.hpp`

Inheritance Relationships

Base Type

- public `hipcub::BaseDigitExtractor< KeyT >` (*Template Struct BaseDigitExtractor*)

Struct Documentation

template<typename **KeyT**>

struct `hipcub::BFEDigitExtractor` : public `hipcub::BaseDigitExtractor<KeyT>`

A wrapper type to extract digits. Uses the BFE intrinsic to extract a key from a digit.

Public Types

enum [anonymous]

Values:

enumerator **FLOAT_KEY**

typedef Traits<*KeyT*> **TraitsT**

typedef *TraitsT*::UnsignedBits **UnsignedBits**

Public Functions

__device__ __forceinline__ inline explicit **BFEDigitExtractor**(uint32_t bit_start = 0, uint32_t num_bits = 0)

__device__ __forceinline__ inline uint32_t **Digit**(*UnsignedBits* key)

Public Members

uint32_t **bit_start**

uint32_t **num_bits**

Public Static Functions

__device__ __forceinline__ static inline *UnsignedBits* **ProcessFloatMinusZero**(*UnsignedBits* key)

Template Struct `block_raking_layout`

- Defined in file `hipcub_backend_rocprim_block_block_raking_layout.hpp`

Nested Relationships

Nested Types

- *Struct* `block_raking_layout::TempStorage`

Struct Documentation

```
template<typename T, int BLOCK_THREADS, int ARCH = HIPCUB_ARCH>
```

```
struct hipcub::block_raking_layout
```

BlockRakingLayout provides a conflict-free shared memory layout abstraction for 1D raking across thread block data.

Overview This type facilitates a shared memory usage pattern where a block of CUDA threads places elements into shared memory and then reduces the active parallelism to one “raking” warp of threads for serially aggregating consecutive sequences of shared items. Padding is inserted to eliminate bank conflicts (for most data types).

Template Parameters

- **T** – The data type to be exchanged.
- **BLOCK_THREADS** – The thread block size in threads.
- **PTX_ARCH** – [optional]

Public Types

```
enum [anonymous]
```

Values:

```
enumerator SHARED_ELEMENTS
```

The total number of elements that need to be cooperatively reduced.

```
enumerator MAX_RAKING_THREADS
```

Maximum number of warp-synchronous raking threads.

```
enumerator SEGMENT_LENGTH
```

Number of raking elements per warp-synchronous raking thread (rounded up)

```
enumerator RAKING_THREADS
```

Never use a raking thread that will have no valid data (e.g., when **BLOCK_THREADS** is 62 and **SEGMENT_LENGTH** is 2, we should only use 31 raking threads)

```
enumerator USE_SEGMENT_PADDING
```

Pad each segment length with one element if segment length is not relatively prime to warp size and can't be optimized as a vector load.

```
enumerator GRID_ELEMENTS
```

Total number of elements in the raking grid.

```
enumerator UNGUARDED
```

Whether or not we need bounds checking during raking (the number of reduction elements is not a multiple of the number of raking threads)

Public Static Functions

`__device__ static inline T *PlacementPtr(TempStorage &temp_storage, unsigned int linear_tid)`

Returns the location for the calling thread to place data into the grid.

`__device__ static inline T *RakingPtr(TempStorage &temp_storage, unsigned int linear_tid)`

Returns the location for the calling thread to begin sequential raking.

struct **TempStorage** : public hipcub::Uninitialized<_TempStorage>

Alias wrapper allowing storage to be unioned.

Public Types

enum [anonymous]

Values:

typedef UnitWord<_TempStorage>::DeviceWord **DeviceWord**

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

`__host__ __device__ __forceinline__ inline _TempStorage &Alias()`

Alias.

Public Members

DeviceWord **storage**[WORDS]

Backing storage.

Struct `block_raking_layout::TempStorage`

- Defined in file `hipcub_backend_rocprim_block_block_raking_layout.hpp`

Nested Relationships

This struct is a nested type of *Template Struct `block_raking_layout`*.

Inheritance Relationships

Base Type

- `public hipcub::Uninitialized< _TempStorage >` (*Template Struct Uninitialized*)

Struct Documentation

struct `hipcub::block_raking_layout::TempStorage` : public `hipcub::Uninitialized<_TempStorage>`
 Alias wrapper allowing storage to be unioned.

Public Types

enum [**anonymous**]

Values:

typedef `UnitWord<_TempStorage>::DeviceWord` **DeviceWord**

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

`__host__ __device__ __forceinline__ inline _TempStorage &Alias()`

Alias.

Public Members

DeviceWord **storage**[WORDS]

Backing storage.

Struct `BlockMergeSortStrategy::TempStorage`

- Defined in file `_hipcub_backend_rocprim_block_block_merge_sort.hpp`

Nested Relationships

This struct is a nested type of *Template Class BlockMergeSortStrategy*.

Inheritance Relationships

Base Type

- public hipcub::Uninitialized< _TempStorage > (*Template Struct Uninitialized*)

Struct Documentation

```
struct hipcub::BlockMergeSortStrategy::TempStorage : public hipcub::Uninitialized<_TempStorage>
    {BlockMergeSort }
```

Public Types

```
enum [anonymous]
```

Values:

```
typedef UnitWord<_TempStorage>::DeviceWord DeviceWord
```

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

```
__host__ __device__ __forceinline__ inline _TempStorage &Alias()
    Alias.
```

Public Members

```
DeviceWord storage[WORDS]
    Backing storage.
```

Struct BlockRadixRank::PrefixCallback

- Defined in file_hipcub_backend_rocprim_block_block_radix_rank.hpp

Nested Relationships

This struct is a nested type of *Template Class BlockRadixRank*.

Struct Documentation

struct hipcub::BlockRadixRank::PrefixCallback

Block-scan prefix callback

Public Functions

__device__ inline PackedCounter operator() (PackedCounter block_aggregate)

Struct BlockRadixRank::TempStorage

- Defined in file_hipcub_backend_rocprim_block_block_radix_rank.hpp

Nested Relationships

This struct is a nested type of *Template Class BlockRadixRank*.

Inheritance Relationships

Base Type

- public hipcub::Uninitialized<_TempStorage> (*Template Struct Uninitialized*)

Struct Documentation

struct hipcub::BlockRadixRank::TempStorage : public hipcub::Uninitialized<_TempStorage>
{BlockScan}

Public Types

enum [anonymous]

Values:

typedef UnitWord<_TempStorage>::DeviceWord **DeviceWord**

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

`__host__ __device__ __forceinline__ inline _TempStorage &Alias()`
Alias.

Public Members

DeviceWord **storage**[WORDS]
Backing storage.

Struct BlockRadixRankMatch::TempStorage

- Defined in file `hipcub_backend_rocprim_block_block_radix_rank.hpp`

Nested Relationships

This struct is a nested type of *Template Class BlockRadixRankMatch*.

Inheritance Relationships

Base Type

- `public hipcub::Uninitialized< _TempStorage >` (*Template Struct Uninitialized*)

Struct Documentation

```
struct hipcub::BlockRadixRankMatch::TempStorage : public hipcub::Uninitialized<_TempStorage>
{BlockScan}
```

Public Types

enum [**anonymous**]

Values:

typedef UnitWord<_TempStorage>::DeviceWord **DeviceWord**

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

```
__host__ __device__ __forceinline__ inline _TempStorage &Alias()
    Alias.
```

Public Members

```
DeviceWord storage[WORDS]
    Backing storage.
```

Struct BlockRunLengthDecode::TempStorage

- Defined in file `hipcub_backend_rocprim_block_block_run_length_decode.hpp`

Nested Relationships

This struct is a nested type of *Template Class BlockRunLengthDecode*.

Inheritance Relationships

Base Type

- `public hipcub::Uninitialized< _TempStorage >` (*Template Struct Uninitialized*)

Struct Documentation

```
struct hipcub::BlockRunLengthDecode::TempStorage : public hipcub::Uninitialized<_TempStorage>
```

Public Types

```
enum [anonymous]
```

Values:

```
typedef UnitWord<_TempStorage>::DeviceWord DeviceWord
```

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

`__host__ __device__ __forceinline__ inline TempStorage &Alias()`
Alias.

Public Members

DeviceWord **storage**[WORDS]
Backing storage.

Struct CacheModifiedOutputIterator::Reference

- Defined in file `hipcub_backend_rocprim_iterator_cache_modified_output_iterator.hpp`

Nested Relationships

This struct is a nested type of *Template Class CacheModifiedOutputIterator*.

Struct Documentation

struct `hipcub::CacheModifiedOutputIterator::Reference`

Public Functions

`__host__ __device__ __forceinline__ inline Reference(ValueType *ptr)`
Constructor.

`__device__ __forceinline__ inline ValueType operator=(ValueType val)`
Assignment.

Public Members

ValueType ***ptr**

Struct CachingDeviceAllocator

- Defined in file `hipcub_backend_cub_util_allocator.hpp`

Nested Relationships

Nested Types

- *Struct CachingDeviceAllocator::BlockDescriptor*
- *Class CachingDeviceAllocator::TotalBytes*

Inheritance Relationships

Base Type

- public CachingDeviceAllocator

Struct Documentation

```
struct hipcub::CachingDeviceAllocator : public CachingDeviceAllocator
```

Public Types

```
typedef bool (*Compare)(const BlockDescriptor&, const BlockDescriptor&)
    BlockDescriptor comparator function interface.
```

```
typedef std::multiset<BlockDescriptor, Compare> CachedBlocks
    Set type for cached blocks (ordered by size)
```

```
typedef std::multiset<BlockDescriptor, Compare> BusyBlocks
    Set type for live blocks (ordered by ptr)
```

```
typedef std::map<int, TotalBytes> GpuCachedBytes
    Map type of device ordinals to the number of cached bytes cached by each device.
```

Public Functions

```
inline hipError_t SetMaxCachedBytes(size_t max_cached_bytes)
inline hipError_t DeviceAllocate(int device, void **d_ptr, size_t bytes, hipStream_t active_stream = 0)
inline hipError_t DeviceAllocate(void **d_ptr, size_t bytes, hipStream_t active_stream = 0)
inline hipError_t DeviceFree(int device, void *d_ptr)
inline hipError_t DeviceFree(void *d_ptr)
inline hipError_t FreeAllCached()
```

inline void **NearestPowerOf**(unsigned int &power, size_t &rounded_bytes, unsigned int base, size_t value)

Round up to the nearest power-of

inline **CachingDeviceAllocator**(unsigned int bin_growth, unsigned int min_bin = 1, unsigned int max_bin = *INVALID_BIN*, size_t max_cached_bytes = *INVALID_SIZE*, bool skip_cleanup = false, bool debug = false)

Set of live device allocations currently in use.

Constructor.

Parameters

- **bin_growth** – Geometric growth factor for bin-sizes
- **min_bin** – Minimum bin (default is $\text{bin_growth} \wedge 1$)
- **max_bin** – Maximum bin (default is no max bin)
- **max_cached_bytes** – Maximum aggregate cached bytes per device (default is no limit)
- **skip_cleanup** – Whether or not to skip a call to `FreeAllCached()` when the destructor is called (default is to deallocate)
- **debug** – Whether or not to print (de)allocation events to stdout (default is no stderr output)

inline **CachingDeviceAllocator**(bool skip_cleanup = false, bool debug = false)

Default constructor.

Configured with:

which delineates five bin-sizes: 512B, 4KB, 32KB, 256KB, and 2MB and sets a maximum of 6,291,455 cached bytes per device

- $\text{bin_growth} = 8$
- $\text{min_bin} = 3$
- $\text{max_bin} = 7$
- $\text{max_cached_bytes} = (\text{bin_growth} \wedge \text{max_bin}) * 3 - 1 = 6,291,455$ bytes

inline hipError_t **SetMaxCachedBytes**(size_t max_cached_bytes)

Sets the limit on the number bytes this allocator is allowed to cache per device.

Changing the ceiling of cached bytes does not cause any allocations (in-use or cached-in-reserve) to be freed. See `FreeAllCached()`.

inline hipError_t **DeviceAllocate**(int device, void **d_ptr, size_t bytes, hipStream_t active_stream = 0)

Provides a suitable allocation of device memory for the given size on the specified device.

Once freed, the allocation becomes available immediately for reuse within the `active_stream` with which it was associated with during allocation, and it becomes available for reuse within other streams when all prior work submitted to `active_stream` has completed.

Parameters

- **device** – Device on which to place the allocation
- **d_ptr** – Reference to pointer to the allocation
- **bytes** – Minimum number of bytes for the allocation

- **active_stream** – The stream to be associated with this allocation

inline hipError_t **DeviceAllocate**(void **d_ptr, size_t bytes, hipStream_t active_stream = 0)

Provides a suitable allocation of device memory for the given size on the current device.

Once freed, the allocation becomes available immediately for reuse within the `active_stream` with which it was associated with during allocation, and it becomes available for reuse within other streams when all prior work submitted to `active_stream` has completed.

Parameters

- **d_ptr** – Reference to pointer to the allocation
- **bytes** – Minimum number of bytes for the allocation
- **active_stream** – The stream to be associated with this allocation

inline hipError_t **DeviceFree**(int device, void *d_ptr)

Frees a live allocation of device memory on the specified device, returning it to the allocator.

Once freed, the allocation becomes available immediately for reuse within the `active_stream` with which it was associated with during allocation, and it becomes available for reuse within other streams when all prior work submitted to `active_stream` has completed.

inline hipError_t **DeviceFree**(void *d_ptr)

Frees a live allocation of device memory on the current device, returning it to the allocator.

Once freed, the allocation becomes available immediately for reuse within the `active_stream` with which it was associated with during allocation, and it becomes available for reuse within other streams when all prior work submitted to `active_stream` has completed.

inline hipError_t **FreeAllCached**()

Frees all cached device allocations on all devices.

inline virtual **~CachingDeviceAllocator**()

Destructor.

Public Members

std::mutex **mutex**

unsigned int **bin_growth**

Mutex for thread-safety.

unsigned int **min_bin**

Geometric growth factor for bin-sizes.

unsigned int **max_bin**

Minimum bin enumeration.

size_t **min_bin_bytes**

Maximum bin enumeration.

size_t **max_bin_bytes**

Minimum bin size.

size_t **max_cached_bytes**

Maximum bin size.

const bool **skip_cleanup**

Maximum aggregate cached bytes per device.

bool **debug**

Whether or not to skip a call to FreeAllCached() when destructor is called. (The CUDA runtime may have already shut down for statically declared allocators)

GpuCachedBytes **cached_bytes**

Whether or not to print (de)allocation events to stdout.

CachedBlocks **cached_blocks**

Map of device ordinal to aggregate cached bytes on that device.

BusyBlocks **live_blocks**

Set of cached device allocations available for reuse.

Public Static Functions

static inline unsigned int **IntPow**(unsigned int base, unsigned int exp)

Integer pow function for unsigned base and exponent

Public Static Attributes

static const unsigned int **INVALID_BIN** = (unsigned int)-1

Out-of-bounds bin.

static const size_t **INVALID_SIZE** = (size_t)-1

Invalid size.

static const int **INVALID_DEVICE_ORDINAL** = -1

Invalid device ordinal.

struct **BlockDescriptor**

Descriptor for device memory allocations

Public Functions

inline **BlockDescriptor**(void *d_ptr, int device)

inline **BlockDescriptor**(int device)

Public Members

void ***d_ptr**

size_t **bytes**

unsigned int **bin**

int **device**

hipStream_t **associated_stream**

hipEvent_t **ready_event**

Public Static Functions

static inline bool **PtrCompare**(const *BlockDescriptor* &a, const *BlockDescriptor* &b)

static inline bool **SizeCompare**(const *BlockDescriptor* &a, const *BlockDescriptor* &b)

class **TotalBytes**

Public Functions

inline **TotalBytes**()

Public Members

size_t **free**

size_t **live**

Struct `CachingDeviceAllocator::BlockDescriptor`

- Defined in file `hipcub_backend_rocprim_util_allocator.hpp`

Nested Relationships

This struct is a nested type of *Struct `CachingDeviceAllocator`*.

Struct Documentation

struct hipcub:: *CachingDeviceAllocator* : **BlockDescriptor**

Descriptor for device memory allocations

Public Functions

inline **BlockDescriptor**(void *d_ptr, int device)

inline **BlockDescriptor**(int device)

Public Members

void ***d_ptr**

size_t **bytes**

unsigned int **bin**

int **device**

hipStream_t **associated_stream**

hipEvent_t **ready_event**

Public Static Functions

static inline bool **PtrCompare**(const *BlockDescriptor* &a, const *BlockDescriptor* &b)

static inline bool **SizeCompare**(const *BlockDescriptor* &a, const *BlockDescriptor* &b)

Struct DeviceHistogram

- Defined in file `hipcub_backend_cub_device_device_histogram.hpp`

Struct Documentation

struct `hipcub::DeviceHistogram`

Public Static Functions

```
template<typename SampleIteratorT, typename CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t HistogramEven(void *d_temp_storage, size_t &temp_storage_bytes,
SampleIteratorT d_samples, CounterT *d_histogram, int
num_levels, LevelT lower_level, LevelT upper_level,
OffsetT num_samples, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename SampleIteratorT, typename CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t HistogramEven(void *d_temp_storage, size_t &temp_storage_bytes,
SampleIteratorT d_samples, CounterT *d_histogram, int
num_levels, LevelT lower_level, LevelT upper_level,
OffsetT num_row_samples, OffsetT num_rows, size_t
row_stride_bytes, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<int NUM_CHANNELS, int NUM_ACTIVE_CHANNELS, typename SampleIteratorT, typename
CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t MultiHistogramEven(void *d_temp_storage, size_t &temp_storage_bytes,
SampleIteratorT d_samples, CounterT
*d_histogram[NUM_ACTIVE_CHANNELS], int
num_levels[NUM_ACTIVE_CHANNELS], LevelT
lower_level[NUM_ACTIVE_CHANNELS], LevelT
upper_level[NUM_ACTIVE_CHANNELS], OffsetT
num_pixels, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<int NUM_CHANNELS, int NUM_ACTIVE_CHANNELS, typename SampleIteratorT, typename
CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t MultiHistogramEven(void *d_temp_storage, size_t &temp_storage_bytes,
SampleIteratorT d_samples, CounterT
*d_histogram[NUM_ACTIVE_CHANNELS], int
num_levels[NUM_ACTIVE_CHANNELS], LevelT
lower_level[NUM_ACTIVE_CHANNELS], LevelT
upper_level[NUM_ACTIVE_CHANNELS], OffsetT
num_row_pixels, OffsetT num_rows, size_t
row_stride_bytes, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename SampleIteratorT, typename CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t HistogramRange(void *d_temp_storage, size_t &temp_storage_bytes,
SampleIteratorT d_samples, CounterT *d_histogram, int
num_levels, LevelT *d_levels, OffsetT num_samples,
hipStream_t stream = 0, bool debug_synchronous = false)
```

```
template<typename SampleIteratorT, typename CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t HistogramRange(void *d_temp_storage, size_t &temp_storage_bytes,
    SampleIteratorT d_samples, CounterT *d_histogram, int
    num_levels, LevelT *d_levels, OffsetT num_row_samples,
    OffsetT num_rows, size_t row_stride_bytes, hipStream_t
    stream = 0, bool debug_synchronous = false)
```

```
template<int NUM_CHANNELS, int NUM_ACTIVE_CHANNELS, typename SampleIteratorT, typename
CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t MultiHistogramRange(void *d_temp_storage, size_t
    &temp_storage_bytes, SampleIteratorT d_samples,
    CounterT
    *d_histogram[NUM_ACTIVE_CHANNELS], int
    num_levels[NUM_ACTIVE_CHANNELS], LevelT
    *d_levels[NUM_ACTIVE_CHANNELS], OffsetT
    num_pixels, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<int NUM_CHANNELS, int NUM_ACTIVE_CHANNELS, typename SampleIteratorT, typename
CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t MultiHistogramRange(void *d_temp_storage, size_t
    &temp_storage_bytes, SampleIteratorT d_samples,
    CounterT
    *d_histogram[NUM_ACTIVE_CHANNELS], int
    num_levels[NUM_ACTIVE_CHANNELS], LevelT
    *d_levels[NUM_ACTIVE_CHANNELS], OffsetT
    num_row_pixels, OffsetT num_rows, size_t
    row_stride_bytes, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename SampleIteratorT, typename CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t HistogramEven(void *d_temp_storage, size_t &temp_storage_bytes,
    SampleIteratorT d_samples, CounterT *d_histogram, int
    num_levels, LevelT lower_level, LevelT upper_level,
    OffsetT num_samples, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename SampleIteratorT, typename CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t HistogramEven(void *d_temp_storage, size_t &temp_storage_bytes,
    SampleIteratorT d_samples, CounterT *d_histogram, int
    num_levels, LevelT lower_level, LevelT upper_level,
    OffsetT num_row_samples, OffsetT num_rows, size_t
    row_stride_bytes, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<int NUM_CHANNELS, int NUM_ACTIVE_CHANNELS, typename SampleIteratorT, typename
CounterT, typename LevelT, typename OffsetT>
```



```

__host__ static inline hipError_t MultiHistogramEven(void *d_temp_storage, size_t &temp_storage_bytes,
    SampleIteratorT d_samples, CounterT
    *d_histogram[NUM_ACTIVE_CHANNELS], int
    num_levels[NUM_ACTIVE_CHANNELS], LevelT
    lower_level[NUM_ACTIVE_CHANNELS], LevelT
    upper_level[NUM_ACTIVE_CHANNELS], OffsetT
    num_pixels, hipStream_t stream = 0, bool
    debug_synchronous = false)

```

```

template<int NUM_CHANNELS, int NUM_ACTIVE_CHANNELS, typename SampleIteratorT, typename
CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t MultiHistogramEven(void *d_temp_storage, size_t &temp_storage_bytes,
    SampleIteratorT d_samples, CounterT
    *d_histogram[NUM_ACTIVE_CHANNELS], int
    num_levels[NUM_ACTIVE_CHANNELS], LevelT
    lower_level[NUM_ACTIVE_CHANNELS], LevelT
    upper_level[NUM_ACTIVE_CHANNELS], OffsetT
    num_row_pixels, OffsetT num_rows, size_t
    row_stride_bytes, hipStream_t stream = 0, bool
    debug_synchronous = false)

```

```

template<typename SampleIteratorT, typename CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t HistogramRange(void *d_temp_storage, size_t &temp_storage_bytes,
    SampleIteratorT d_samples, CounterT *d_histogram, int
    num_levels, LevelT *d_levels, OffsetT num_samples,
    hipStream_t stream = 0, bool debug_synchronous = false)

```

```

template<typename SampleIteratorT, typename CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t HistogramRange(void *d_temp_storage, size_t &temp_storage_bytes,
    SampleIteratorT d_samples, CounterT *d_histogram, int
    num_levels, LevelT *d_levels, OffsetT num_row_samples,
    OffsetT num_rows, size_t row_stride_bytes, hipStream_t
    stream = 0, bool debug_synchronous = false)

```

```

template<int NUM_CHANNELS, int NUM_ACTIVE_CHANNELS, typename SampleIteratorT, typename
CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t MultiHistogramRange(void *d_temp_storage, size_t
    &temp_storage_bytes, SampleIteratorT d_samples,
    CounterT
    *d_histogram[NUM_ACTIVE_CHANNELS], int
    num_levels[NUM_ACTIVE_CHANNELS], LevelT
    *d_levels[NUM_ACTIVE_CHANNELS], OffsetT
    num_pixels, hipStream_t stream = 0, bool
    debug_synchronous = false)

```

```

template<int NUM_CHANNELS, int NUM_ACTIVE_CHANNELS, typename SampleIteratorT, typename
CounterT, typename LevelT, typename OffsetT>
__host__ static inline hipError_t MultiHistogramRange(void *d_temp_storage, size_t
    &temp_storage_bytes, SampleIteratorT d_samples,
    CounterT
    *d_histogram[NUM_ACTIVE_CHANNELS], int
    num_levels[NUM_ACTIVE_CHANNELS], LevelT
    *d_levels[NUM_ACTIVE_CHANNELS], OffsetT
    num_row_pixels, OffsetT num_rows, size_t
    row_stride_bytes, hipStream_t stream = 0, bool
    debug_synchronous = false)

```

Struct DeviceMergeSort

- Defined in file `hipcub_backend_cub_device_device_merge_sort.hpp`

Struct Documentation

struct `hipcub::DeviceMergeSort`

Public Static Functions

```
template<typename KeyIteratorT, typename ValueIteratorT, typename OffsetT, typename CompareOpT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, std::size_t &temp_storage_bytes,
                                             KeyIteratorT d_keys, ValueIteratorT d_items, OffsetT
                                             num_items, CompareOpT compare_op, hipStream_t stream = 0,
                                             bool debug_synchronous = false)
```

```
template<typename KeyInputIteratorT, typename ValueInputIteratorT, typename KeyIteratorT,
typename ValueIteratorT, typename OffsetT, typename CompareOpT>
__host__ static inline hipError_t SortPairsCopy(void *d_temp_storage, std::size_t &temp_storage_bytes,
                                                 KeyInputIteratorT d_input_keys, ValueInputIteratorT
                                                 d_input_items, KeyIteratorT d_output_keys,
                                                 ValueIteratorT d_output_items, OffsetT num_items,
                                                 CompareOpT compare_op, hipStream_t stream = 0, bool
                                                 debug_synchronous = false)
```

```
template<typename KeyIteratorT, typename OffsetT, typename CompareOpT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, std::size_t &temp_storage_bytes,
                                             KeyIteratorT d_keys, OffsetT num_items, CompareOpT
                                             compare_op, hipStream_t stream = 0, bool debug_synchronous =
                                             false)
```

```
template<typename KeyInputIteratorT, typename KeyIteratorT, typename OffsetT, typename CompareOpT>
__host__ static inline hipError_t SortKeysCopy(void *d_temp_storage, std::size_t &temp_storage_bytes,
                                                 KeyInputIteratorT d_input_keys, KeyIteratorT
                                                 d_output_keys, OffsetT num_items, CompareOpT
                                                 compare_op, hipStream_t stream = 0, bool
                                                 debug_synchronous = false)
```

```
template<typename KeyIteratorT, typename ValueIteratorT, typename OffsetT, typename CompareOpT>
__host__ static inline hipError_t StableSortPairs(void *d_temp_storage, std::size_t &temp_storage_bytes,
                                                  KeyIteratorT d_keys, ValueIteratorT d_items, OffsetT
                                                  num_items, CompareOpT compare_op, hipStream_t
                                                  stream = 0, bool debug_synchronous = false)
```

```
template<typename KeyIteratorT, typename OffsetT, typename CompareOpT>
```

```

__host__ static inline hipError_t StableSortKeys(void *d_temp_storage, std::size_t &temp_storage_bytes,
KeyIteratorT d_keys, OffsetT num_items, CompareOpT
compare_op, hipStream_t stream = 0, bool
debug_synchronous = false)

```

```

template<typename KeyIteratorT, typename ValueIteratorT, typename OffsetT, typename
CompareOpT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, std::size_t &temp_storage_bytes,
KeyIteratorT d_keys, ValueIteratorT d_items, OffsetT
num_items, CompareOpT compare_op, hipStream_t stream = 0,
bool debug_synchronous = false)

```

```

template<typename KeyInputIteratorT, typename ValueInputIteratorT, typename KeyIteratorT,
typename ValueIteratorT, typename OffsetT, typename CompareOpT>
__host__ static inline hipError_t SortPairsCopy(void *d_temp_storage, std::size_t &temp_storage_bytes,
KeyInputIteratorT d_input_keys, ValueInputIteratorT
d_input_items, KeyIteratorT d_output_keys,
ValueIteratorT d_output_items, OffsetT num_items,
CompareOpT compare_op, hipStream_t stream = 0, bool
debug_synchronous = false)

```

```

template<typename KeyIteratorT, typename OffsetT, typename CompareOpT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, std::size_t &temp_storage_bytes,
KeyIteratorT d_keys, OffsetT num_items, CompareOpT
compare_op, hipStream_t stream = 0, bool debug_synchronous =
false)

```

```

template<typename KeyInputIteratorT, typename KeyIteratorT, typename OffsetT, typename
CompareOpT>
__host__ static inline hipError_t SortKeysCopy(void *d_temp_storage, std::size_t &temp_storage_bytes,
KeyInputIteratorT d_input_keys, KeyIteratorT
d_output_keys, OffsetT num_items, CompareOpT
compare_op, hipStream_t stream = 0, bool
debug_synchronous = false)

```

```

template<typename KeyIteratorT, typename ValueIteratorT, typename OffsetT, typename
CompareOpT>
__host__ static inline hipError_t StableSortPairs(void *d_temp_storage, std::size_t &temp_storage_bytes,
KeyIteratorT d_keys, ValueIteratorT d_items, OffsetT
num_items, CompareOpT compare_op, hipStream_t
stream = 0, bool debug_synchronous = false)

```

```

template<typename KeyIteratorT, typename OffsetT, typename CompareOpT>
__host__ static inline hipError_t StableSortKeys(void *d_temp_storage, std::size_t &temp_storage_bytes,
KeyIteratorT d_keys, OffsetT num_items, CompareOpT
compare_op, hipStream_t stream = 0, bool
debug_synchronous = false)

```

Struct DevicePartition

- Defined in file_hipcub_backend_cub_device_device_partition.hpp

Struct Documentation

struct hipcub::DevicePartition

Public Static Functions

```
template<typename InputIteratorT, typename FlagIterator, typename OutputIteratorT, typename NumSelectedIteratorT>
```

```
__host__ __forceinline__ static inline hipError_t Flagged(void *d_temp_storage, size_t  
    &temp_storage_bytes, InputIteratorT d_in,  
    FlagIterator d_flags, OutputIteratorT d_out,  
    NumSelectedIteratorT d_num_selected_out, int  
    num_items, hipStream_t stream = 0, bool  
    debug_synchronous = false)
```

Parameters

- **d_temp_storage** – Device-accessible allocation of temporary storage. When NULL, the required allocation size is written to **temp_storage_bytes** and no work is done.
- **temp_storage_bytes** – Reference to size in bytes of **d_temp_storage** allocation
- **d_in** – Pointer to the input sequence of data items
- **d_flags** – Pointer to the input sequence of selection flags
- **d_out** – Pointer to the output sequence of partitioned data items
- **d_num_selected_out** – Pointer to the output total number of items selected (i.e., the offset of the unselected partition)
- **num_items** – Total number of items to select from
- **stream** – [optional] hip stream to launch kernels within. Default is stream₀.
- **debug_synchronous** – [optional] Whether or not to synchronize the stream after every kernel launch to check for errors. May cause significant slowdown. Default is false.

```
template<typename InputIteratorT, typename OutputIteratorT, typename NumSelectedIteratorT,  
typename SelectOp>
```

```
__host__ __forceinline__ static inline hipError_t If(void *d_temp_storage, size_t &temp_storage_bytes,  
    InputIteratorT d_in, OutputIteratorT d_out,  
    NumSelectedIteratorT d_num_selected_out, int  
    num_items, SelectOp select_op, hipStream_t stream = 0,  
    bool debug_synchronous = false)
```

Parameters

- **d_temp_storage** – Device-accessible allocation of temporary storage. When NULL, the required allocation size is written to **temp_storage_bytes** and no work is done.
- **temp_storage_bytes** – Reference to size in bytes of **d_temp_storage** allocation
- **d_in** – Pointer to the input sequence of data items

- **d_out** – Pointer to the output sequence of partitioned data items
- **d_num_selected_out** – Pointer to the output total number of items selected (i.e., the offset of the unselected partition)
- **num_items** – Total number of items to select from
- **select_op** – Unary selection operator
- **stream** – [optional] hip stream to launch kernels within. Default is stream₀.
- **debug_synchronous** – [optional] Whether or not to synchronize the stream after every kernel launch to check for errors. May cause significant slowdown. Default is `false`.

```
template<typename InputIteratorT, typename FirstOutputIteratorT, typename
SecondOutputIteratorT, typename UnselectedOutputIteratorT, typename NumSelectedIteratorT,
typename SelectFirstPartOp, typename SelectSecondPartOp>
```

```
__host__ __forceinline__ static inline hipError_t If(void *d_temp_storage, std::size_t &temp_storage_bytes,
    InputIteratorT d_in, FirstOutputIteratorT
    d_first_part_out, SecondOutputIteratorT
    d_second_part_out, UnselectedOutputIteratorT
    d_unselected_out, NumSelectedIteratorT
    d_num_selected_out, int num_items, SelectFirstPartOp
    select_first_part_op, SelectSecondPartOp
    select_second_part_op, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename InputIteratorT, typename FlagIterator, typename OutputIteratorT, typename
NumSelectedIteratorT>
```

```
__host__ __forceinline__ static inline hipError_t Flagged(void *d_temp_storage, size_t
    &temp_storage_bytes, InputIteratorT d_in,
    FlagIterator d_flags, OutputIteratorT d_out,
    NumSelectedIteratorT d_num_selected_out, int
    num_items, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

Parameters

- **d_temp_storage** – Device-accessible allocation of temporary storage. When NULL, the required allocation size is written to `temp_storage_bytes` and no work is done.
- **temp_storage_bytes** – Reference to size in bytes of `d_temp_storage` allocation
- **d_in** – Pointer to the input sequence of data items
- **d_flags** – Pointer to the input sequence of selection flags
- **d_out** – Pointer to the output sequence of partitioned data items
- **d_num_selected_out** – Pointer to the output total number of items selected (i.e., the offset of the unselected partition)
- **num_items** – Total number of items to select from
- **stream** – [optional] hip stream to launch kernels within. Default is stream₀.
- **debug_synchronous** – [optional] Whether or not to synchronize the stream after every kernel launch to check for errors. May cause significant slowdown. Default is `false`.

```
template<typename InputIteratorT, typename OutputIteratorT, typename NumSelectedIteratorT,
typename SelectOp>
```

```
__host__ __forceinline__ static inline hipError_t If(void *d_temp_storage, size_t &temp_storage_bytes,
    InputIteratorT d_in, OutputIteratorT d_out,
    NumSelectedIteratorT d_num_selected_out, int
    num_items, SelectOp select_op, hipStream_t stream = 0,
    bool debug_synchronous = false)
```

Parameters

- **d_temp_storage** – Device-accessible allocation of temporary storage. When NULL, the required allocation size is written to **temp_storage_bytes** and no work is done.
- **temp_storage_bytes** – Reference to size in bytes of **d_temp_storage** allocation
- **d_in** – Pointer to the input sequence of data items
- **d_out** – Pointer to the output sequence of partitioned data items
- **d_num_selected_out** – Pointer to the output total number of items selected (i.e., the offset of the unselected partition)
- **num_items** – Total number of items to select from
- **select_op** – Unary selection operator
- **stream** – [optional] hip stream to launch kernels within. Default is `stream0`.
- **debug_synchronous** – [optional] Whether or not to synchronize the stream after every kernel launch to check for errors. May cause significant slowdown. Default is `false`.

```
template<typename InputIteratorT, typename FirstOutputIteratorT, typename
SecondOutputIteratorT, typename UnselectedOutputIteratorT, typename NumSelectedIteratorT,
typename SelectFirstPartOp, typename SelectSecondPartOp>
__host__ __forceinline__ static inline hipError_t If(void *d_temp_storage, std::size_t &temp_storage_bytes,
    InputIteratorT d_in, FirstOutputIteratorT
    d_first_part_out, SecondOutputIteratorT
    d_second_part_out, UnselectedOutputIteratorT
    d_unselected_out, NumSelectedIteratorT
    d_num_selected_out, int num_items, SelectFirstPartOp
    select_first_part_op, SelectSecondPartOp
    select_second_part_op, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

Struct DeviceRadixSort

- Defined in file `hipcub_backend_cub_device_device_radix_sort.hpp`

Struct Documentation

```
struct hipcub::DeviceRadixSort
```

Public Static Functions

```
template<typename KeyT, typename ValueT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
*d_keys_in, KeyT *d_keys_out, const ValueT *d_values_in,
ValueT *d_values_out, int num_items, int begin_bit = 0, int
end_bit = sizeof(KeyT) * 8, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, DoubleBuffer<ValueT>
&d_values, int num_items, int begin_bit = 0, int end_bit =
sizeof(KeyT) * 8, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT>
__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
&temp_storage_bytes, const KeyT *d_keys_in,
KeyT *d_keys_out, const ValueT *d_values_in,
ValueT *d_values_out, int num_items, int begin_bit
= 0, int end_bit = sizeof(KeyT) * 8, hipStream_t
stream = 0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT>
__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
&temp_storage_bytes, DoubleBuffer<KeyT>
&d_keys, DoubleBuffer<ValueT> &d_values, int
num_items, int begin_bit = 0, int end_bit =
sizeof(KeyT) * 8, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename KeyT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
*d_keys_in, KeyT *d_keys_out, int num_items, int begin_bit = 0,
int end_bit = sizeof(KeyT) * 8, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename KeyT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, int num_items, int begin_bit = 0,
int end_bit = sizeof(KeyT) * 8, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename KeyT>
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
const KeyT *d_keys_in, KeyT *d_keys_out, int
num_items, int begin_bit = 0, int end_bit =
sizeof(KeyT) * 8, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename KeyT>
```

```
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
    DoubleBuffer<KeyT> &d_keys, int num_items, int
    begin_bit = 0, int end_bit = sizeof(KeyT) * 8,
    hipStream_t stream = 0, bool debug_synchronous =
    false)
```

```
template<typename KeyT, typename ValueT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
    *d_keys_in, KeyT *d_keys_out, const ValueT *d_values_in,
    ValueT *d_values_out, int num_items, int begin_bit = 0, int
    end_bit = sizeof(KeyT) * 8, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes,
    DoubleBuffer<KeyT> &d_keys, DoubleBuffer<ValueT>
    &d_values, int num_items, int begin_bit = 0, int end_bit =
    sizeof(KeyT) * 8, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT>
__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
    &temp_storage_bytes, const KeyT *d_keys_in,
    KeyT *d_keys_out, const ValueT *d_values_in,
    ValueT *d_values_out, int num_items, int begin_bit
    = 0, int end_bit = sizeof(KeyT) * 8, hipStream_t
    stream = 0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT>
__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
    &temp_storage_bytes, DoubleBuffer<KeyT>
    &d_keys, DoubleBuffer<ValueT> &d_values, int
    num_items, int begin_bit = 0, int end_bit =
    sizeof(KeyT) * 8, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename KeyT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
    *d_keys_in, KeyT *d_keys_out, int num_items, int begin_bit = 0,
    int end_bit = sizeof(KeyT) * 8, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename KeyT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes,
    DoubleBuffer<KeyT> &d_keys, int num_items, int begin_bit = 0,
    int end_bit = sizeof(KeyT) * 8, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename KeyT>
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
    const KeyT *d_keys_in, KeyT *d_keys_out, int
    num_items, int begin_bit = 0, int end_bit =
    sizeof(KeyT) * 8, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename KeyT>
```



```
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
    DoubleBuffer<KeyT> &d_keys, int num_items, int
    begin_bit = 0, int end_bit = sizeof(KeyT) * 8,
    hipStream_t stream = 0, bool debug_synchronous =
    false)
```

Struct DeviceSegmentedRadixSort

- Defined in file `hipcub_backend_cub_device_device_segmented_radix_sort.hpp`

Struct Documentation

```
struct hipcub::DeviceSegmentedRadixSort
```

Public Static Functions

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
    *d_keys_in, KeyT *d_keys_out, const ValueT *d_values_in,
    ValueT *d_values_out, int num_items, int num_segments,
    OffsetIteratorT d_begin_offsets, OffsetIteratorT d_end_offsets,
    int begin_bit = 0, int end_bit = sizeof(KeyT) * 8, hipStream_t
    stream = 0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes,
    DoubleBuffer<KeyT> &d_keys, DoubleBuffer<ValueT>
    &d_values, int num_items, int num_segments, OffsetIteratorT
    d_begin_offsets, OffsetIteratorT d_end_offsets, int begin_bit =
    0, int end_bit = sizeof(KeyT) * 8, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
    &temp_storage_bytes, const KeyT *d_keys_in,
    KeyT *d_keys_out, const ValueT *d_values_in,
    ValueT *d_values_out, int num_items, int
    num_segments, OffsetIteratorT d_begin_offsets,
    OffsetIteratorT d_end_offsets, int begin_bit = 0, int
    end_bit = sizeof(KeyT) * 8, hipStream_t stream =
    0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
    &temp_storage_bytes, DoubleBuffer<KeyT>
    &d_keys, DoubleBuffer<ValueT> &d_values, int
    num_items, int num_segments, OffsetIteratorT
    d_begin_offsets, OffsetIteratorT d_end_offsets, int
    begin_bit = 0, int end_bit = sizeof(KeyT) * 8,
    hipStream_t stream = 0, bool debug_synchronous =
    false)
```

```
template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
*d_keys_in, KeyT *d_keys_out, int num_items, int
num_segments, OffsetIteratorT d_begin_offsets, OffsetIteratorT
d_end_offsets, int begin_bit = 0, int end_bit = sizeof(KeyT) * 8,
hipStream_t stream = 0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, int num_items, int
num_segments, OffsetIteratorT d_begin_offsets, OffsetIteratorT
d_end_offsets, int begin_bit = 0, int end_bit = sizeof(KeyT) * 8,
hipStream_t stream = 0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
const KeyT *d_keys_in, KeyT *d_keys_out, int
num_items, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, int
begin_bit = 0, int end_bit = sizeof(KeyT) * 8,
hipStream_t stream = 0, bool debug_synchronous =
false)
```

```
template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, int num_items, int
num_segments, OffsetIteratorT d_begin_offsets,
OffsetIteratorT d_end_offsets, int begin_bit = 0, int
end_bit = sizeof(KeyT) * 8, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
*d_keys_in, KeyT *d_keys_out, const ValueT *d_values_in,
ValueT *d_values_out, int num_items, int num_segments,
OffsetIteratorT d_begin_offsets, OffsetIteratorT d_end_offsets,
int begin_bit = 0, int end_bit = sizeof(KeyT) * 8, hipStream_t
stream = 0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, DoubleBuffer<ValueT>
&d_values, int num_items, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, int begin_bit =
0, int end_bit = sizeof(KeyT) * 8, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
```

```

__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
&temp_storage_bytes, const KeyT *d_keys_in,
KeyT *d_keys_out, const ValueT *d_values_in,
ValueT *d_values_out, int num_items, int
num_segments, OffsetIteratorT d_begin_offsets,
OffsetIteratorT d_end_offsets, int begin_bit = 0, int
end_bit = sizeof(KeyT) * 8, hipStream_t stream =
0, bool debug_synchronous = false)

```

```

template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
&temp_storage_bytes, DoubleBuffer<KeyT>
&d_keys, DoubleBuffer<ValueT> &d_values, int
num_items, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, int
begin_bit = 0, int end_bit = sizeof(KeyT) * 8,
hipStream_t stream = 0, bool debug_synchronous =
false)

```

```

template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
*d_keys_in, KeyT *d_keys_out, int num_items, int
num_segments, OffsetIteratorT d_begin_offsets, OffsetIteratorT
d_end_offsets, int begin_bit = 0, int end_bit = sizeof(KeyT) * 8,
hipStream_t stream = 0, bool debug_synchronous = false)

```

```

template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, int num_items, int
num_segments, OffsetIteratorT d_begin_offsets, OffsetIteratorT
d_end_offsets, int begin_bit = 0, int end_bit = sizeof(KeyT) * 8,
hipStream_t stream = 0, bool debug_synchronous = false)

```

```

template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
const KeyT *d_keys_in, KeyT *d_keys_out, int
num_items, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, int
begin_bit = 0, int end_bit = sizeof(KeyT) * 8,
hipStream_t stream = 0, bool debug_synchronous =
false)

```

```

template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, int num_items, int
num_segments, OffsetIteratorT d_begin_offsets,
OffsetIteratorT d_end_offsets, int begin_bit = 0, int
end_bit = sizeof(KeyT) * 8, hipStream_t stream = 0,
bool debug_synchronous = false)

```

Struct DeviceSegmentedReduce

- Defined in file `hipcub_backend_cub_device_device_segmented_reduce.hpp`

Struct Documentation

struct hipcub::DeviceSegmentedReduce

Public Static Functions

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT,
typename ReductionOp, typename T>
__host__ static inline hipError_t Reduce(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, ReductionOp
reduction_op, T initial_value, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT>
__host__ static inline hipError_t Sum(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT>
__host__ static inline hipError_t Min(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT>
__host__ static inline hipError_t ArgMin(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, hipStream_t stream
= 0, bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT>
__host__ static inline hipError_t Max(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT>
__host__ static inline hipError_t ArgMax(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, hipStream_t stream
= 0, bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT,
typename ReductionOp, typename T>
```

```
__host__ static inline hipError_t Reduce(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, ReductionOp
reduction_op, T initial_value, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT>
__host__ static inline hipError_t Sum(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT>
__host__ static inline hipError_t Min(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT>
__host__ static inline hipError_t ArgMin(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, hipStream_t stream
= 0, bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT>
__host__ static inline hipError_t Max(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename OffsetIteratorT>
__host__ static inline hipError_t ArgMax(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets, hipStream_t stream
= 0, bool debug_synchronous = false)
```

Struct DeviceSegmentedSort

- Defined in file_hipcub_backend_cub_device_device_segmented_sort.hpp

Struct Documentation

```
struct hipcub::DeviceSegmentedSort
```

Public Static Functions

```
template<typename KeyT, typename BeginOffsetIteratorT, typename EndOffsetIteratorT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
*d_keys_in, KeyT *d_keys_out, int num_items, int
num_segments, BeginOffsetIteratorT d_begin_offsets,
EndOffsetIteratorT d_end_offsets, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename KeyT, typename BeginOffsetIteratorT, typename EndOffsetIteratorT>
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
const KeyT *d_keys_in, KeyT *d_keys_out, int
num_items, int num_segments, BeginOffsetIteratorT
d_begin_offsets, EndOffsetIteratorT d_end_offsets,
hipStream_t stream = 0, bool debug_synchronous =
false)
```

```
template<typename KeyT, typename BeginOffsetIteratorT, typename EndOffsetIteratorT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, int num_items, int
num_segments, BeginOffsetIteratorT d_begin_offsets,
EndOffsetIteratorT d_end_offsets, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename KeyT, typename BeginOffsetIteratorT, typename EndOffsetIteratorT>
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, int num_items, int
num_segments, BeginOffsetIteratorT
d_begin_offsets, EndOffsetIteratorT d_end_offsets,
hipStream_t stream = 0, bool debug_synchronous =
false)
```

```
template<typename KeyT, typename BeginOffsetIteratorT, typename EndOffsetIteratorT>
__host__ static inline hipError_t StableSortKeys(void *d_temp_storage, size_t &temp_storage_bytes, const
KeyT *d_keys_in, KeyT *d_keys_out, int num_items, int
num_segments, BeginOffsetIteratorT d_begin_offsets,
EndOffsetIteratorT d_end_offsets, hipStream_t stream =
0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename BeginOffsetIteratorT, typename EndOffsetIteratorT>
__host__ static inline hipError_t StableSortKeysDescending(void *d_temp_storage, size_t
&temp_storage_bytes, const KeyT
*d_keys_in, KeyT *d_keys_out, int
num_items, int num_segments,
BeginOffsetIteratorT d_begin_offsets,
EndOffsetIteratorT d_end_offsets,
hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename KeyT, typename BeginOffsetIteratorT, typename EndOffsetIteratorT>
__host__ static inline hipError_t StableSortKeys(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, int num_items, int
num_segments, BeginOffsetIteratorT d_begin_offsets,
EndOffsetIteratorT d_end_offsets, hipStream_t stream =
0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename BeginOffsetIteratorT, typename EndOffsetIteratorT>
__host__ static inline hipError_t StableSortKeysDescending(void *d_temp_storage, size_t
&temp_storage_bytes, DoubleBuffer<KeyT>
&d_keys, int num_items, int num_segments,
BeginOffsetIteratorT d_begin_offsets,
EndOffsetIteratorT d_end_offsets,
hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename BeginOffsetIteratorT, typename
EndOffsetIteratorT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
*d_keys_in, KeyT *d_keys_out, const ValueT *d_values_in,
ValueT *d_values_out, int num_items, int num_segments,
BeginOffsetIteratorT d_begin_offsets, EndOffsetIteratorT
d_end_offsets, hipStream_t stream = 0, bool debug_synchronous
= false)
```

```
template<typename KeyT, typename ValueT, typename BeginOffsetIteratorT, typename
EndOffsetIteratorT>
__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
&temp_storage_bytes, const KeyT *d_keys_in,
KeyT *d_keys_out, const ValueT *d_values_in,
ValueT *d_values_out, int num_items, int
num_segments, BeginOffsetIteratorT
d_begin_offsets, EndOffsetIteratorT d_end_offsets,
hipStream_t stream = 0, bool debug_synchronous =
false)
```

```
template<typename KeyT, typename ValueT, typename BeginOffsetIteratorT, typename
EndOffsetIteratorT>
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, DoubleBuffer<ValueT>
&d_values, int num_items, int num_segments,
BeginOffsetIteratorT d_begin_offsets, EndOffsetIteratorT
d_end_offsets, hipStream_t stream = 0, bool debug_synchronous
= false)
```

```
template<typename KeyT, typename ValueT, typename BeginOffsetIteratorT, typename
EndOffsetIteratorT>
__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
&temp_storage_bytes, DoubleBuffer<KeyT>
&d_keys, DoubleBuffer<ValueT> &d_values, int
num_items, int num_segments,
BeginOffsetIteratorT d_begin_offsets,
EndOffsetIteratorT d_end_offsets, hipStream_t
stream = 0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename BeginOffsetIteratorT, typename
EndOffsetIteratorT>
```



```
__host__ static inline hipError_t StableSortPairs(void *d_temp_storage, size_t &temp_storage_bytes,
    const KeyT *d_keys_in, KeyT *d_keys_out, const
    ValueT *d_values_in, ValueT *d_values_out, int
    num_items, int num_segments, BeginOffsetIteratorT
    d_begin_offsets, EndOffsetIteratorT d_end_offsets,
    hipStream_t stream = 0, bool debug_synchronous =
    false)
```

```
template<typename KeyT, typename ValueT, typename BeginOffsetIteratorT, typename
EndOffsetIteratorT>
```

```
__host__ static inline hipError_t StableSortPairsDescending(void *d_temp_storage, size_t
    &temp_storage_bytes, const KeyT
    *d_keys_in, KeyT *d_keys_out, const
    ValueT *d_values_in, ValueT
    *d_values_out, int num_items, int
    num_segments, BeginOffsetIteratorT
    d_begin_offsets, EndOffsetIteratorT
    d_end_offsets, hipStream_t stream = 0,
    bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename BeginOffsetIteratorT, typename
EndOffsetIteratorT>
```

```
__host__ static inline hipError_t StableSortPairs(void *d_temp_storage, size_t &temp_storage_bytes,
    DoubleBuffer<KeyT> &d_keys, DoubleBuffer<ValueT>
    &d_values, int num_items, int num_segments,
    BeginOffsetIteratorT d_begin_offsets,
    EndOffsetIteratorT d_end_offsets, hipStream_t stream =
    0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename BeginOffsetIteratorT, typename
EndOffsetIteratorT>
```

```
__host__ static inline hipError_t StableSortPairsDescending(void *d_temp_storage, size_t
    &temp_storage_bytes,
    DoubleBuffer<KeyT> &d_keys,
    DoubleBuffer<ValueT> &d_values, int
    num_items, int num_segments,
    BeginOffsetIteratorT d_begin_offsets,
    EndOffsetIteratorT d_end_offsets,
    hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
```

```
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
    *d_keys_in, KeyT *d_keys_out, const ValueT *d_values_in,
    ValueT *d_values_out, int num_items, int num_segments,
    OffsetIteratorT d_begin_offsets, OffsetIteratorT d_end_offsets,
    hipStream_t stream = 0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
```

```
__host__ static inline hipError_t SortPairs(void *d_temp_storage, size_t &temp_storage_bytes,
    DoubleBuffer<KeyT> &d_keys, DoubleBuffer<ValueT>
    &d_values, int num_items, int num_segments, OffsetIteratorT
    d_begin_offsets, OffsetIteratorT d_end_offsets, hipStream_t
    stream = 0, bool debug_synchronous = false)
```



```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
&temp_storage_bytes, const KeyT *d_keys_in,
KeyT *d_keys_out, const ValueT *d_values_in,
ValueT *d_values_out, int num_items, int
num_segments, OffsetIteratorT d_begin_offsets,
OffsetIteratorT d_end_offsets, hipStream_t stream
= 0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t SortPairsDescending(void *d_temp_storage, size_t
&temp_storage_bytes, DoubleBuffer<KeyT>
&d_keys, DoubleBuffer<ValueT> &d_values, int
num_items, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets,
hipStream_t stream = 0, bool debug_synchronous =
false)
```

```
template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes, const KeyT
*d_keys_in, KeyT *d_keys_out, int num_items, int
num_segments, OffsetIteratorT d_begin_offsets, OffsetIteratorT
d_end_offsets, hipStream_t stream = 0, bool debug_synchronous
= false)
```

```
template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeys(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, int num_items, int
num_segments, OffsetIteratorT d_begin_offsets, OffsetIteratorT
d_end_offsets, hipStream_t stream = 0, bool debug_synchronous
= false)
```

```
template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
const KeyT *d_keys_in, KeyT *d_keys_out, int
num_items, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets,
hipStream_t stream = 0, bool debug_synchronous =
false)
```

```
template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t SortKeysDescending(void *d_temp_storage, size_t &temp_storage_bytes,
DoubleBuffer<KeyT> &d_keys, int num_items, int
num_segments, OffsetIteratorT d_begin_offsets,
OffsetIteratorT d_end_offsets, hipStream_t stream =
0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t StableSortPairs(void *d_temp_storage, size_t &temp_storage_bytes,
const KeyT *d_keys_in, KeyT *d_keys_out, const
ValueT *d_values_in, ValueT *d_values_out, int
num_items, int num_segments, OffsetIteratorT
d_begin_offsets, OffsetIteratorT d_end_offsets,
hipStream_t stream = 0, bool debug_synchronous =
false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t StableSortPairs(void *d_temp_storage, size_t &temp_storage_bytes,
    DoubleBuffer<KeyT> &d_keys, DoubleBuffer<ValueT>
    &d_values, int num_items, int num_segments,
    OffsetIteratorT d_begin_offsets, OffsetIteratorT
    d_end_offsets, hipStream_t stream = 0, bool
    debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t StableSortPairsDescending(void *d_temp_storage, size_t
    &temp_storage_bytes, const KeyT
    *d_keys_in, KeyT *d_keys_out, const
    ValueT *d_values_in, ValueT
    *d_values_out, int num_items, int
    num_segments, OffsetIteratorT
    d_begin_offsets, OffsetIteratorT
    d_end_offsets, hipStream_t stream = 0,
    bool debug_synchronous = false)
```

```
template<typename KeyT, typename ValueT, typename OffsetIteratorT>
__host__ static inline hipError_t StableSortPairsDescending(void *d_temp_storage, size_t
    &temp_storage_bytes,
    DoubleBuffer<KeyT> &d_keys,
    DoubleBuffer<ValueT> &d_values, int
    num_items, int num_segments,
    OffsetIteratorT d_begin_offsets,
    OffsetIteratorT d_end_offsets, hipStream_t
    stream = 0, bool debug_synchronous =
    false)
```

```
template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t StableSortKeys(void *d_temp_storage, size_t &temp_storage_bytes, const
    KeyT *d_keys_in, KeyT *d_keys_out, int num_items, int
    num_segments, OffsetIteratorT d_begin_offsets,
    OffsetIteratorT d_end_offsets, hipStream_t stream = 0,
    bool debug_synchronous = false)
```

```
template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t StableSortKeys(void *d_temp_storage, size_t &temp_storage_bytes,
    DoubleBuffer<KeyT> &d_keys, int num_items, int
    num_segments, OffsetIteratorT d_begin_offsets,
    OffsetIteratorT d_end_offsets, hipStream_t stream = 0,
    bool debug_synchronous = false)
```

```
template<typename KeyT, typename OffsetIteratorT>
__host__ static inline hipError_t StableSortKeysDescending(void *d_temp_storage, size_t
    &temp_storage_bytes, const KeyT
    *d_keys_in, KeyT *d_keys_out, int
    num_items, int num_segments,
    OffsetIteratorT d_begin_offsets,
    OffsetIteratorT d_end_offsets, hipStream_t
    stream = 0, bool debug_synchronous = false)
```

```
template<typename KeyT, typename OffsetIteratorT>
```

```

__host__ static inline hipError_t StableSortKeysDescending(void *d_temp_storage, size_t
&temp_storage_bytes, DoubleBuffer<KeyT>
&d_keys, int num_items, int num_segments,
OffsetIteratorT d_begin_offsets,
OffsetIteratorT d_end_offsets, hipStream_t
stream = 0, bool debug_synchronous = false)

```

Template Struct DeviceSpmv::SpmvParams

- Defined in file `hipcub_backend_cub_device_device_spmv.hpp`

Nested Relationships

This struct is a nested type of *Class DeviceSpmv*.

Struct Documentation

```
template<typename ValueT, typename OffsetT>
```

```
struct hipcub::DeviceSpmv::SpmvParams
```

```
< Signed integer type for sequence offsets
```

Public Members

ValueT ***d_values**

Pointer to the array of `num_nonzeros` values of the corresponding nonzero elements of matrix **A**.

OffsetT ***d_row_end_offsets**

Pointer to the array of `m` offsets demarcating the end of every row in `d_column_indices` and `d_values`.

OffsetT ***d_column_indices**

Pointer to the array of `num_nonzeros` column-indices of the corresponding nonzero elements of matrix **A**. (Indices are zero-valued.)

ValueT ***d_vector_x**

Pointer to the array of `num_cols` values corresponding to the dense input vector `x`

ValueT ***d_vector_y**

Pointer to the array of `num_rows` values corresponding to the dense output vector `y`

int **num_rows**

Number of rows of matrix **A**.

int **num_cols**

Number of columns of matrix **A**.

int **num_nonzeros**

Number of nonzero elements of matrix **A**.

ValueT **alpha**

Alpha multiplicand.

ValueT **beta**

Beta addend-multiplicand.

`::cub::TexRefInputIterator<ValueT, 66778899, OffsetT> t_vector_x`

`::hipcub::TexRefInputIterator<ValueT, 66778899, OffsetT> t_vector_x`

Template Struct DoubleBuffer

- Defined in file `hipcub_backend_rocprim_util_type.hpp`

Struct Documentation

```
template<typename T>
```

```
struct hipcub::DoubleBuffer
```

Public Functions

```
__host__ __device__ inline DoubleBuffer()
```

```
__host__ __device__ inline DoubleBuffer(T *d_current, T *d_alternate)
```

```
__host__ __device__ inline T *Current()
```

```
__host__ __device__ inline T *Alternate()
```

Public Members

```
T *d_buffers[2]
```

```
int selector
```

Struct Equality

- Defined in file `hipcub_backend_rocprim_thread_thread_operators.hpp`

Struct Documentation

```
struct hipcub::Equality
```

Public Functions

```
template<class T>
__host__ __device__ inline constexpr bool operator() (const T &a, const T &b) const
```

Template Struct GridEvenShare

- Defined in file `hipcub_backend_rocprim_grid_grid_even_share.hpp`

Struct Documentation

```
template<typename OffsetT>
```

```
struct hipcub::GridEvenShare
```

GridEvenShare is a descriptor utility for distributing input among CUDA thread blocks in an “even-share” fashion. Each thread block gets roughly the same number of input tiles.

Overview Each thread block is assigned a consecutive sequence of input tiles. To help preserve alignment and eliminate the overhead of guarded loads for all but the last thread block, *GridEvenShare* assigns one of three different amounts of work to a given thread block: “big”, “normal”, or “last”. The “big” workloads are one scheduling grain larger than “normal”. The “last” work unit for the last thread block may be partially-full if the input is not an even multiple of the scheduling grain size.

Before invoking a child grid, a parent thread will typically construct an instance of *GridEvenShare*. The instance can be passed to child thread blocks which can initialize their per-thread block offsets using *BlockInit()*.

Public Functions

```
__host__ __device__ __forceinline__ inline GridEvenShare()
```

Constructor.

```
__host__ __device__ __forceinline__ inline void DispatchInit(OffsetT num_items_, int max_grid_size, int
                                                                    tile_items)
```

Dispatch initializer. To be called prior to kernel launch.

Parameters

- **num_items_** – Total number of input items
- **max_grid_size** – Maximum grid size allowable (actual grid size may be less if not warranted by the the number of input items)

- **tile_items** – Number of data items per input tile

```
template<int TILE_ITEMS>
```

```
__device__ __forceinline__ inline void BlockInit(int block_id, Int2Type<GRID_MAPPING_RAKE>)
```

Initializes ranges for the specified thread block index. Specialized for a “raking” access pattern in which each thread block is assigned a consecutive sequence of input tiles.

```
template<int TILE_ITEMS>
```

```
__device__ __forceinline__ inline void BlockInit(int block_id,  
Int2Type<GRID_MAPPING_STRIP_MINE>)
```

Block-initialization, specialized for a “raking” access pattern in which each thread block is assigned a consecutive sequence of input tiles.

```
template<int TILE_ITEMS, GridMappingStrategy STRATEGY>
```

```
__device__ __forceinline__ inline void BlockInit()
```

Block-initialization, specialized for “strip mining” access pattern in which the input tiles assigned to each thread block are separated by a stride equal to the the extent of the grid.

```
template<int TILE_ITEMS>
```

```
__device__ __forceinline__ inline void BlockInit(OffsetT block_offset, OffsetT block_end)
```

Block-initialization, specialized for a “raking” access pattern in which each thread block is assigned a consecutive sequence of input tiles.

Parameters

- **block_offset** – Threadblock begin offset (inclusive)
- **block_end** – Threadblock end offset (exclusive)

Public Members

OffsetT **num_items**

Total number of input items.

int **grid_size**

Grid size in thread blocks.

OffsetT **block_offset**

OffsetT into input marking the beginning of the owning thread block’s segment of input tiles.

OffsetT **block_end**

OffsetT into input of marking the end (one-past) of the owning thread block’s segment of input tiles.

OffsetT **block_stride**

Stride between input tiles.

Template Struct If

- Defined in file_hipcub_backend_rocprim_util_type.hpp

Struct Documentation

```
template<bool B, typename T, typename F>
```

```
struct hipcub::If
```

Public Types

```
using Type = typename std::conditional<B, T, F>::type
```

Struct Inequality

- Defined in file_hipcub_backend_rocprim_thread_thread_operators.hpp

Struct Documentation

```
struct hipcub::Inequality
```

Public Functions

```
template<class T>
__host__ __device__ inline constexpr bool operator() (const T &a, const T &b) const
```

Template Struct InequalityWrapper

- Defined in file_hipcub_backend_rocprim_thread_thread_operators.hpp

Struct Documentation

```
template<class EqualityOp>
```

```
struct hipcub::InequalityWrapper
```

Public Functions

```
__host__ __device__ inline InequalityWrapper(EqualityOp op)  
template<class T>  
__host__ __device__ inline bool operator()(const T &a, const T &b)
```

Public Members

EqualityOp **op**

Template Struct Int2Type

- Defined in file `hipcub_backend_rocprim_util_type.hpp`

Struct Documentation

```
template<int A>  
struct hipcub::Int2Type
```

Public Types

```
enum [anonymous]  
    Values:  
    enumerator VALUE
```

Template Struct IsPointer

- Defined in file `hipcub_backend_rocprim_util_type.hpp`

Struct Documentation

```
template<typename T>  
struct hipcub::IsPointer
```


Public Static Attributes

```
static constexpr bool VALUE = std::is_pointer<T>::value
```

Template Struct IsVolatile

- Defined in file_hipcub_backend_rocprim_util_type.hpp

Struct Documentation

```
template<typename T>
```

```
struct hipcub::IsVolatile
```

Public Static Attributes

```
static constexpr bool VALUE = std::is_volatile<T>::value
```

Template Struct Log2

- Defined in file_hipcub_backend_rocprim_util_type.hpp

Struct Documentation

```
template<int N>
```

```
struct hipcub::Log2
```

Public Static Attributes

```
static constexpr int VALUE = detail::Log2Impl<N>::VALUE
```

Struct Max

- Defined in file_hipcub_backend_rocprim_thread_thread_operators.hpp

Struct Documentation

struct hipcub: **Max**

Public Functions

```
template<class T>
__host__ __device__ inline constexpr T operator()(const T &a, const T &b) const
```

Struct Min

- Defined in file_hipcub_backend_rocprim_thread_thread_operators.hpp

Struct Documentation

struct hipcub: **Min**

Public Functions

```
template<class T>
__host__ __device__ inline constexpr T operator()(const T &a, const T &b) const
```

Template Struct PowerOfTwo

- Defined in file_hipcub_backend_rocprim_util_type.hpp

Struct Documentation

```
template<int N>
```

```
struct hipcub: PowerOfTwo
```

Public Static Attributes

```
static constexpr bool VALUE = ::rocprim::detail::is_power_of_two(N)
```

Template Struct RadixSortTwiddle

- Defined in file_hipcub_backend_rocprim_block_radix_rank_sort_operations.hpp

Struct Documentation

```
template<bool IS_DESCENDING, typename KeyT>
```

```
struct hipcub::RadixSortTwiddle
```

Twiddling keys for radix sort.

Public Types

```
typedef Traits<KeyT> TraitsT
```

```
typedef TraitsT::UnsignedBits UnsignedBits
```

Public Static Functions

```
__host__ __device__ __forceinline__ static inline UnsignedBits In(UnsignedBits key)
```

```
__host__ __device__ __forceinline__ static inline UnsignedBits Out(UnsignedBits key)
```

```
__host__ __device__ __forceinline__ static inline UnsignedBits DefaultKey()
```

Template Struct RemoveQualifiers

- Defined in file_hipcub_backend_rocprim_util_type.hpp

Struct Documentation

```
template<typename T>
```

```
struct hipcub::RemoveQualifiers
```

Public Types

```
using Type = typename std::remove_cv<T>::type
```

Template Struct ShiftDigitExtractor

- Defined in file_hipcub_backend_rocprim_block_radix_rank_sort_operations.hpp

Inheritance Relationships

Base Type

- public hipcub::BaseDigitExtractor< KeyT > (*Template Struct BaseDigitExtractor*)

Struct Documentation

```
template<typename KeyT>
```

```
struct hipcub::ShiftDigitExtractor : public hipcub::BaseDigitExtractor<KeyT>
```

A wrapper type to extract digits. Uses a combination of shift and bitwise and to extract digits.

Public Types

```
enum [anonymous]
```

Values:

```
enumerator FLOAT_KEY
```

```
typedef Traits<KeyT> TraitsT
```

```
typedef TraitsT::UnsignedBits UnsignedBits
```

Public Functions

```
__device__ __forceinline__ inline explicit ShiftDigitExtractor(uint32_t bit_start = 0, uint32_t num_bits = 0)
```

```
__device__ __forceinline__ inline uint32_t Digit(UnsignedBits key)
```

Public Members

```
uint32_t bit_start
```

```
uint32_t mask
```

Public Static Functions

```
__device__ __forceinline__ static inline UnsignedBits ProcessFloatMinusZero(UnsignedBits key)
```

Struct Sum

- Defined in file_hipcub_backend_rocprim_thread_thread_operators.hpp

Struct Documentation

```
struct hipcub::Sum
```

Public Functions

```
template<class T>
__host__ __device__ inline constexpr T operator() (const T &a, const T &b) const
```

Template Struct Uninitialized

- Defined in file_hipcub_backend_rocprim_util_type.hpp

Struct Documentation

```
template<typename T>
```

```
struct hipcub::Uninitialized
```

A storage-backing wrapper that allows types with non-trivial constructors to be aliased in unions.

Public Types

```
enum [anonymous]
```

Values:

```
enumerator WORDS
```

```
typedef UnitWord<T>::DeviceWord DeviceWord
```

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

`__host__ __device__ __forceinline__ inline T &Alias()`
Alias.

Public Members

DeviceWord **storage**[*WORDS*]
Backing storage.

Struct WarpExchange::TempStorage

- Defined in file `hipcub_backend_rocprim_warp_warp_exchange.hpp`

Nested Relationships

This struct is a nested type of *Template Class WarpExchange*.

Inheritance Relationships

Base Type

- `public hipcub::Uninitialized<_TempStorage >` (*Template Struct Uninitialized*)

Struct Documentation

```
struct hipcub::WarpExchange::TempStorage : public hipcub::Uninitialized<_TempStorage>
```

Public Types

enum [**anonymous**]

Values:

typedef UnitWord<_TempStorage>::DeviceWord **DeviceWord**

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

```
__host__ __device__ __forceinline__ inline TempStorage &Alias()
    Alias.
```

Public Members

```
DeviceWord storage[WORDS]
    Backing storage.
```

Template Struct WarpLoad::LoadInternal

- Defined in file `hipcub_backend_rocprim_warp_warp_load.hpp`

Nested Relationships

This struct is a nested type of *Template Class WarpLoad*.

Struct Documentation

```
template<WarpLoadAlgorithm _POLICY>
struct LoadInternal
```

Template Struct WarpLoad::LoadInternal< WARP_LOAD_DIRECT >

- Defined in file `hipcub_backend_rocprim_warp_warp_load.hpp`

Nested Relationships

This struct is a nested type of *Template Class WarpLoad*.

Struct Documentation

```
template<>
struct hipcub::WarpLoad::LoadInternal<WARP_LOAD_DIRECT>
```

Public Types

```
using TempStorage = NullType
```

Public Functions

```
__device__ __forceinline__ inline LoadInternal(TempStorage&, int linear_tid)
```

```
template<typename InputIteratorT>  
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT  
(&items)[ITEMS_PER_THREAD])
```

```
template<typename InputIteratorT>  
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT  
(&items)[ITEMS_PER_THREAD], int valid_items)
```

```
template<typename InputIteratorT, typename DefaultT>  
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT  
(&items)[ITEMS_PER_THREAD], int valid_items, DefaultT  
oob_default)
```

Public Members

```
int linear_tid
```

Template Struct **WarpLoad::LoadInternal< WARP_LOAD_STRIPED >**

- Defined in file `hipcub_backend_rocprim_warp_warp_load.hpp`

Nested Relationships

This struct is a nested type of *Template Class WarpLoad*.

Struct Documentation

```
template<>  
struct hipcub::WarpLoad::LoadInternal<WARP_LOAD_STRIPED>
```


Public Types

```
using TempStorage = NullType
```

Public Functions

```
__device__ __forceinline__ inline LoadInternal(TempStorage&, int linear_tid)
```

```
template<typename InputIteratorT>
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT
                                             (&items)[ITEMS_PER_THREAD])
```

```
template<typename InputIteratorT>
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT
                                             (&items)[ITEMS_PER_THREAD], int valid_items)
```

```
template<typename InputIteratorT, typename DefaultT>
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT
                                             (&items)[ITEMS_PER_THREAD], int valid_items, DefaultT
                                             oob_default)
```

Public Members

```
int linear_tid
```

Template Struct `WarpLoad::LoadInternal<WARP_LOAD_TRANSPOSE>` >

- Defined in file `hipcub_backend_rocprim_warp_warp_load.hpp`

Nested Relationships

This struct is a nested type of *Template Class WarpLoad*.

Struct Documentation

```
template<>
struct hipcub::WarpLoad::LoadInternal<WARP_LOAD_TRANSPOSE>
```

Public Types

```
using WarpExchangeT = WarpExchange<InputT, ITEMS_PER_THREAD, LOGICAL_WARP_THREADS, ARCH>
```

```
using TempStorage = typename WarpExchangeT::TempStorage
```

Public Functions

```
__device__ __forceinline__ inline LoadInternal(TempStorage &temp_storage, int linear_tid)
```

```
template<typename InputIteratorT>  
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT  
                                             (&items)[ITEMS_PER_THREAD])
```

```
template<typename InputIteratorT>  
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT  
                                             (&items)[ITEMS_PER_THREAD], int valid_items)
```

```
template<typename InputIteratorT, typename DefaultT>  
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT  
                                             (&items)[ITEMS_PER_THREAD], int valid_items, DefaultT  
                                             oob_default)
```

Public Members

```
TempStorage &temp_storage
```

```
int linear_tid
```

Template Struct `WarpLoad::LoadInternal<WARP_LOAD_VECTORIZE>`

- Defined in file `hipcub_backend_rocprim_warp_warp_load.hpp`

Nested Relationships

This struct is a nested type of *Template Class WarpLoad*.

Struct Documentation

```
template<>  
struct hipcub::WarpLoad::LoadInternal<WARP_LOAD_VECTORIZE>
```

Public Types

```
using TempStorage = NullType
```

Public Functions

```
__device__ __forceinline__ inline LoadInternal(TempStorage&, int linear_tid)

template<typename InputIteratorT>
__device__ __forceinline__ inline void Load(InputT *block_ptr, InputT (&items)[ITEMS_PER_THREAD])

template<typename InputIteratorT>
__device__ __forceinline__ inline void Load(const InputT *block_ptr, InputT
                                             (&items)[ITEMS_PER_THREAD])

template<CacheLoadModifier MODIFIER, typename ValueType, typename OffsetT>
__device__ __forceinline__ inline void Load(CacheModifiedInputIterator<MODIFIER, ValueType, OffsetT>
                                             block_itr, InputT (&items)[ITEMS_PER_THREAD])

template<typename _InputIteratorT>
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT
                                             (&items)[ITEMS_PER_THREAD])

template<typename InputIteratorT>
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT
                                             (&items)[ITEMS_PER_THREAD], int valid_items)

template<typename InputIteratorT, typename DefaultT>
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT
                                             (&items)[ITEMS_PER_THREAD], int valid_items, DefaultT
                                             oob_default)
```

Public Members

```
int linear_tid
```

Struct WarpLoad::TempStorage

- Defined in file `hipcub_backend_rocprim_warp_warp_load.hpp`

Nested Relationships

This struct is a nested type of *Template Class WarpLoad*.

Inheritance Relationships

Base Type

- public hipcub::Uninitialized< _TempStorage > (*Template Struct Uninitialized*)

Struct Documentation

```
struct hipcub::WarpLoad::TempStorage : public hipcub::Uninitialized<_TempStorage>
```

Public Types

```
enum [anonymous]
```

Values:

```
typedef UnitWord<_TempStorage>::DeviceWord DeviceWord
```

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

```
__host__ __device__ __forceinline__ inline _TempStorage &Alias()
```

Alias.

Public Members

```
DeviceWord storage[WORDS]
```

Backing storage.

Template Struct WarpStore::StoreInternal

- Defined in file_hipcub_backend_rocprim_warp_warp_store.hpp

Nested Relationships

This struct is a nested type of *Template Class WarpStore*.

Struct Documentation

```
template<WarpStoreAlgorithm _POLICY>
struct StoreInternal
```

Template Struct WarpStore::StoreInternal< WARP_STORE_DIRECT >

- Defined in file_hipcub_backend_rocprim_warp_warp_store.hpp

Nested Relationships

This struct is a nested type of *Template Class WarpStore*.

Struct Documentation

```
template<>
struct hipcub::WarpStore::StoreInternal<WARP_STORE_DIRECT>
```

Public Types

```
using TempStorage = NullType
```

Public Functions

```
__device__ __forceinline__ inline StoreInternal(TempStorage&, int linear_tid)
```

```
template<typename OutputIteratorT>
__device__ __forceinline__ inline void Store(OutputIteratorT block_itr, T
                                             (&items)[ITEMS_PER_THREAD])
```

```
template<typename OutputIteratorT>
__device__ __forceinline__ inline void Store(OutputIteratorT block_itr, T
                                             (&items)[ITEMS_PER_THREAD], int valid_items)
```

Public Members

```
int linear_tid
```

Template Struct WarpStore::StoreInternal< WARP_STORE_STRIPED >

- Defined in file_hipcub_backend_rocprim_warp_warp_store.hpp

Nested Relationships

This struct is a nested type of *Template Class WarpStore*.

Struct Documentation

```
template<>
```

```
struct hipcub::WarpStore::StoreInternal<WARP_STORE_STRIPED>
```

Public Types

```
using TempStorage = NullType
```

Public Functions

```
__device__ __forceinline__ inline StoreInternal(TempStorage&, int linear_tid)
```

```
template<typename OutputIteratorT>  
__device__ __forceinline__ inline void Store(OutputIteratorT block_itr, T  
                                             (&items)[ITEMS_PER_THREAD])
```

```
template<typename OutputIteratorT>  
__device__ __forceinline__ inline void Store(OutputIteratorT block_itr, T  
                                             (&items)[ITEMS_PER_THREAD], int valid_items)
```

Public Members

```
int linear_tid
```

Template Struct WarpStore::StoreInternal< WARP_STORE_TRANSPOSE >

- Defined in file_hipcub_backend_rocprim_warp_warp_store.hpp

Nested Relationships

This struct is a nested type of *Template Class WarpStore*.

Struct Documentation

```
template<>
```

```
struct hipcub::WarpStore::StoreInternal<WARP_STORE_TRANSPOSE>
```

Public Types

```
using WarpExchangeT = WarpExchange<T, ITEMS_PER_THREAD, LOGICAL_WARP_THREADS, ARCH>
```

```
using TempStorage = typename WarpExchangeT::TempStorage
```

Public Functions

```
__device__ __forceinline__ inline StoreInternal(TempStorage &temp_storage, int linear_tid)
```

```
template<typename OutputIteratorT>
__device__ __forceinline__ inline void Store(OutputIteratorT block_itr, T
                                             (&items)[ITEMS_PER_THREAD])
```

```
template<typename OutputIteratorT>
__device__ __forceinline__ inline void Store(OutputIteratorT block_itr, T
                                             (&items)[ITEMS_PER_THREAD], int valid_items)
```

Public Members

```
TempStorage &temp_storage
```

```
int linear_tid
```

Template Struct WarpStore::StoreInternal< WARP_STORE_VECTORIZE >

- Defined in file `hipcub_backend_rocprim_warp_warp_store.hpp`

Nested Relationships

This struct is a nested type of *Template Class WarpStore*.

Struct Documentation

```
template<>
```

```
struct hipcub::WarpStore::StoreInternal<WARP_STORE_VECTORIZE>
```

Public Types

```
using TempStorage = NullType
```

Public Functions

```
__device__ __forceinline__ inline StoreInternal(TempStorage&, int linear_tid)
```

```
template<typename OutputIteratorT>  
__device__ __forceinline__ inline void Store(T *block_ptr, T (&items)[ITEMS_PER_THREAD])
```

```
template<typename _OutputIteratorT>  
__device__ __forceinline__ inline void Store(_OutputIteratorT block_itr, T  
(&items)[ITEMS_PER_THREAD])
```

```
template<typename OutputIteratorT>  
__device__ __forceinline__ inline void Store(OutputIteratorT block_itr, T  
(&items)[ITEMS_PER_THREAD], int valid_items)
```

Public Members

```
int linear_tid
```

Struct WarpStore::TempStorage

- Defined in file `hipcub_backend_rocprim_warp_warp_store.hpp`

Nested Relationships

This struct is a nested type of *Template Class WarpStore*.

Inheritance Relationships

Base Type

- `public hipcub::Uninitialized< _TempStorage >` (*Template Struct Uninitialized*)

Struct Documentation

```
struct hipcub::WarpStore::TempStorage : public hipcub::Uninitialized<_TempStorage>
```

Public Types

```
enum [anonymous]
```

Values:

```
typedef UnitWord<_TempStorage>::DeviceWord DeviceWord
```

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

```
__host__ __device__ __forceinline__ inline _TempStorage &Alias()
```

Alias.

Public Members

```
DeviceWord storage[WORDS]
```

Backing storage.

Template Class BlockHistogram

- Defined in file `_hipcub_backend_rocprim_block_block_histogram.hpp`

Inheritance Relationships

Base Type

- `private rocprim::block_histogram< T, BLOCK_DIM_X, ITEMS_PER_THREAD, BINS, static_cast<::rocprim::block_histogram_algorithm >(ALGORITHM), BLOCK_DIM_Y, BLOCK_DIM_Z >`

Class Documentation

```
template<typename T, int BLOCK_DIM_X, int ITEMS_PER_THREAD, int BINS, BlockHistogramAlgorithm ALGORITHM = BLOCK_HISTO_SORT, int BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int ARCH = HIPCUB_ARCH>
class BlockHistogram : private rocprim::block_histogram<T, BLOCK_DIM_X, ITEMS_PER_THREAD, BINS,
static_cast<::rocprim::block_histogram_algorithm>(ALGORITHM), BLOCK_DIM_Y, BLOCK_DIM_Z>
```

Public Types

```
using TempStorage = typename base_type::storage_type
```

Public Functions

```
__device__ inline BlockHistogram()

__device__ inline BlockHistogram(TempStorage &temp_storage)

template<class CounterT>
__device__ inline void InitHistogram(CounterT histogram[BINS])

template<class CounterT>
__device__ inline void Composite(T (&items)[ITEMS_PER_THREAD], CounterT histogram[BINS])

template<class CounterT>
__device__ inline void Histogram(T (&items)[ITEMS_PER_THREAD], CounterT histogram[BINS])
```

Template Class BlockReduce

- Defined in file `hipcub_backend_rocprim_block_block_reduce.hpp`

Inheritance Relationships

Base Type

- private rocprim::block_reduce< T, BLOCK_DIM_X, static_cast<::rocprim::block_reduce_algorithm >(ALGORITHM), BLOCK_DIM_Y, BLOCK_DIM_Z >

Class Documentation

```
template<typename T, int BLOCK_DIM_X, BlockReduceAlgorithm ALGORITHM = BLOCK_REDUCE_WARP_REDUCTIONS, int BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int ARCH = HIPCUB_ARCH>
class BlockReduce : private rocprim::block_reduce<T, BLOCK_DIM_X,
static_cast<::rocprim::block_reduce_algorithm>(ALGORITHM), BLOCK_DIM_Y, BLOCK_DIM_Z>
```

Public Types

```
using TempStorage = typename base_type::storage_type
```

Public Functions

```
__device__ inline BlockReduce()
__device__ inline BlockReduce(TempStorage &temp_storage)
__device__ inline T Sum(T input)
__device__ inline T Sum(T input, int valid_items)
template<int ITEMS_PER_THREAD>
__device__ inline T Sum(T (&input)[ITEMS_PER_THREAD])
template<typename ReduceOp>
__device__ inline T Reduce(T input, ReduceOp reduce_op)
template<typename ReduceOp>
__device__ inline T Reduce(T input, ReduceOp reduce_op, int valid_items)
template<int ITEMS_PER_THREAD, typename ReduceOp>
__device__ inline T Reduce(T (&input)[ITEMS_PER_THREAD], ReduceOp reduce_op)
```

Template Class BlockAdjacentDifference

- Defined in file_hipcup_backend_rocprim_block_block_adjacent_difference.hpp

Inheritance Relationships

Base Type

- private rocprim::block_adjacent_difference< T, BLOCK_DIM_X, BLOCK_DIM_Y, BLOCK_DIM_Z >

Class Documentation

```
template<typename T, int BLOCK_DIM_X, int BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int ARCH = HIPCUB_ARCH>
class hipcup:: BlockAdjacentDifference : private rocprim::block_adjacent_difference<T, BLOCK_DIM_X,
BLOCK_DIM_Y, BLOCK_DIM_Z>
```

Public Types

using **TempStorage** = typename base_type::storage_type

Public Functions

```
__device__ inline BlockAdjacentDifference()
```

```
__device__ inline BlockAdjacentDifference(TempStorage &temp_storage)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>  
__device__ inline void FlagHeads(FlagT (&head_flags)[ITEMS_PER_THREAD], T  
    (&input)[ITEMS_PER_THREAD], FlagOp flag_op)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>  
__device__ inline void FlagHeads(FlagT (&head_flags)[ITEMS_PER_THREAD], T  
    (&input)[ITEMS_PER_THREAD], FlagOp flag_op, T  
    tile_predecessor_item)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>  
__device__ inline void FlagTails(FlagT (&tail_flags)[ITEMS_PER_THREAD], T  
    (&input)[ITEMS_PER_THREAD], FlagOp flag_op)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>  
__device__ inline void FlagTails(FlagT (&tail_flags)[ITEMS_PER_THREAD], T  
    (&input)[ITEMS_PER_THREAD], FlagOp flag_op, T  
    tile_successor_item)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>  
__device__ inline void FlagHeadsAndTails(FlagT (&head_flags)[ITEMS_PER_THREAD], FlagT  
    (&tail_flags)[ITEMS_PER_THREAD], T  
    (&input)[ITEMS_PER_THREAD], FlagOp flag_op)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>  
__device__ inline void FlagHeadsAndTails(FlagT (&head_flags)[ITEMS_PER_THREAD], FlagT  
    (&tail_flags)[ITEMS_PER_THREAD], T tile_successor_item,  
    T (&input)[ITEMS_PER_THREAD], FlagOp flag_op)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>  
__device__ inline void FlagHeadsAndTails(FlagT (&head_flags)[ITEMS_PER_THREAD], T  
    tile_predecessor_item, FlagT  
    (&tail_flags)[ITEMS_PER_THREAD], T  
    (&input)[ITEMS_PER_THREAD], FlagOp flag_op)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>  
__device__ inline void FlagHeadsAndTails(FlagT (&head_flags)[ITEMS_PER_THREAD], T  
    tile_predecessor_item, FlagT  
    (&tail_flags)[ITEMS_PER_THREAD], T tile_successor_item,  
    T (&input)[ITEMS_PER_THREAD], FlagOp flag_op)
```

Template Class BlockDiscontinuity

- Defined in file_hipcub_backend_rocprim_block_block_discontinuity.hpp

Inheritance Relationships

Base Type

- private rocprim::block_discontinuity< T, BLOCK_DIM_X, BLOCK_DIM_Y, BLOCK_DIM_Z >

Class Documentation

```
template<typename T, int BLOCK_DIM_X, int BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int ARCH = HIPCUB_ARCH>
class hipcub::BlockDiscontinuity : private rocprim::block_discontinuity<T, BLOCK_DIM_X,
BLOCK_DIM_Y, BLOCK_DIM_Z>
```

Public Types

```
using TempStorage = typename base_type::storage_type
```

Public Functions

```
__device__ inline BlockDiscontinuity()
```

```
__device__ inline BlockDiscontinuity(TempStorage &temp_storage)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>
__device__ inline void FlagHeads(FlagT (&head_flags)[ITEMS_PER_THREAD], T
(&input)[ITEMS_PER_THREAD], FlagOp flag_op)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>
__device__ inline void FlagHeads(FlagT (&head_flags)[ITEMS_PER_THREAD], T
(&input)[ITEMS_PER_THREAD], FlagOp flag_op, T
tile_predecessor_item)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>
__device__ inline void FlagTails(FlagT (&tail_flags)[ITEMS_PER_THREAD], T
(&input)[ITEMS_PER_THREAD], FlagOp flag_op)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>
__device__ inline void FlagTails(FlagT (&tail_flags)[ITEMS_PER_THREAD], T
(&input)[ITEMS_PER_THREAD], FlagOp flag_op, T
tile_successor_item)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>
__device__ inline void FlagHeadsAndTails(FlagT (&head_flags)[ITEMS_PER_THREAD], FlagT
(&tail_flags)[ITEMS_PER_THREAD], T
(&input)[ITEMS_PER_THREAD], FlagOp flag_op)
```

```
template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>
```

```
__device__ inline void FlagHeadsAndTails(FlagT (&head_flags)[ITEMS_PER_THREAD], FlagT
(&tail_flags)[ITEMS_PER_THREAD], T tile_successor_item,
T (&input)[ITEMS_PER_THREAD], FlagOp flag_op)

template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>
__device__ inline void FlagHeadsAndTails(FlagT (&head_flags)[ITEMS_PER_THREAD], T
tile_predecessor_item, FlagT
(&tail_flags)[ITEMS_PER_THREAD], T
(&input)[ITEMS_PER_THREAD], FlagOp flag_op)

template<int ITEMS_PER_THREAD, typename FlagT, typename FlagOp>
__device__ inline void FlagHeadsAndTails(FlagT (&head_flags)[ITEMS_PER_THREAD], T
tile_predecessor_item, FlagT
(&tail_flags)[ITEMS_PER_THREAD], T tile_successor_item,
T (&input)[ITEMS_PER_THREAD], FlagOp flag_op)
```

Template Class BlockExchange

- Defined in file_hipcube_backend_rocprim_block_block_exchange.hpp

Inheritance Relationships

Base Type

- private rocprim::block_exchange< InputT, BLOCK_DIM_X, ITEMS_PER_THREAD, BLOCK_DIM_Y, BLOCK_DIM_Z >

Class Documentation

```
template<typename InputT, int BLOCK_DIM_X, int ITEMS_PER_THREAD, bool WARP_TIME_SLICING = false, int
BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int ARCH = HIPCUB_ARCH>
class hipcube::BlockExchange : private rocprim::block_exchange<InputT, BLOCK_DIM_X,
ITEMS_PER_THREAD, BLOCK_DIM_Y, BLOCK_DIM_Z>
```

Public Types

```
using TempStorage = typename base_type::storage_type
```

Public Functions

```
__device__ inline BlockExchange()

__device__ inline BlockExchange(TempStorage &temp_storage)

template<typename OutputT>
__device__ inline void StripedToBlocked(InputT (&input_items)[ITEMS_PER_THREAD], OutputT
(&output_items)[ITEMS_PER_THREAD])

template<typename OutputT>
```

```

__device__ inline void BlockedToStriped(InputT (&input_items)[ITEMS_PER_THREAD], OutputT
    (&output_items)[ITEMS_PER_THREAD])

template<typename OutputT>
__device__ inline void WarpStripedToBlocked(InputT (&input_items)[ITEMS_PER_THREAD], OutputT
    (&output_items)[ITEMS_PER_THREAD])

template<typename OutputT>
__device__ inline void BlockedToWarpStriped(InputT (&input_items)[ITEMS_PER_THREAD], OutputT
    (&output_items)[ITEMS_PER_THREAD])

template<typename OutputT, typename OffsetT>
__device__ inline void ScatterToBlocked(InputT (&input_items)[ITEMS_PER_THREAD], OutputT
    (&output_items)[ITEMS_PER_THREAD], OffsetT
    (&ranks)[ITEMS_PER_THREAD])

template<typename OutputT, typename OffsetT>
__device__ inline void ScatterToStriped(InputT (&input_items)[ITEMS_PER_THREAD], OutputT
    (&output_items)[ITEMS_PER_THREAD], OffsetT
    (&ranks)[ITEMS_PER_THREAD])

template<typename OutputT, typename OffsetT>
__device__ inline void ScatterToStripedGuarded(InputT (&input_items)[ITEMS_PER_THREAD],
    OutputT (&output_items)[ITEMS_PER_THREAD],
    OffsetT (&ranks)[ITEMS_PER_THREAD])

template<typename OutputT, typename OffsetT, typename ValidFlag>
__device__ inline void ScatterToStripedFlagged(InputT (&input_items)[ITEMS_PER_THREAD],
    OutputT (&output_items)[ITEMS_PER_THREAD],
    OffsetT (&ranks)[ITEMS_PER_THREAD], ValidFlag
    (&is_valid)[ITEMS_PER_THREAD])

```

Template Class BlockLoad

- Defined in file `hipcub_backend_rocprim_block_block_load.hpp`

Inheritance Relationships

Base Type

- `private rocprim::block_load< T, BLOCK_DIM_X, ITEMS_PER_THREAD, static_cast<::rocprim::block_load_method >(ALGORITHM), BLOCK_DIM_Y, BLOCK_DIM_Z >`

Class Documentation

```
template<typename T, int BLOCK_DIM_X, int ITEMS_PER_THREAD, BlockLoadAlgorithm ALGORITHM =
BLOCK_LOAD_DIRECT, int BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int ARCH = HIPCUB_ARCH>
class hipcub::BlockLoad : private rocprim::block_load<T, BLOCK_DIM_X, ITEMS_PER_THREAD,
static_cast<::rocprim::block_load_method>(ALGORITHM), BLOCK_DIM_Y, BLOCK_DIM_Z>
```

Public Types

```
using TempStorage = typename base_type::storage_type
```

Public Functions

```
__device__ inline BlockLoad()
```

```
__device__ inline BlockLoad(TempStorage &temp_storage)
```

```
template<class InputIteratorT>
```

```
__device__ inline void Load(InputIteratorT block_iter, T (&items)[ITEMS_PER_THREAD])
```

```
template<class InputIteratorT>
```

```
__device__ inline void Load(InputIteratorT block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items)
```

```
template<class InputIteratorT, class Default>
```

```
__device__ inline void Load(InputIteratorT block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items,
Default oob_default)
```

Template Class BlockMergeSort

- Defined in file_hipcub_backend_rocprim_block_block_merge_sort.hpp

Inheritance Relationships

Base Type

- public hipcub::BlockMergeSortStrategy< KeyT, ValueT, BLOCK_DIM_X *BLOCK_DIM_Y *BLOCK_DIM_Z, ITEMS_PER_THREAD, BlockMergeSort< KeyT, BLOCK_DIM_X, ITEMS_PER_THREAD, ValueT, BLOCK_DIM_Y, BLOCK_DIM_Z > > (*Template Class BlockMergeSortStrategy*)

Class Documentation

```
template<typename KeyT, int BLOCK_DIM_X, int ITEMS_PER_THREAD, typename ValueT = NullType, int
BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1>
class hipcub::BlockMergeSort : public hipcub::BlockMergeSortStrategy<KeyT, ValueT, BLOCK_DIM_X *
BLOCK_DIM_Y * BLOCK_DIM_Z, ITEMS_PER_THREAD, BlockMergeSort<KeyT, BLOCK_DIM_X,
ITEMS_PER_THREAD, ValueT, BLOCK_DIM_Y, BLOCK_DIM_Z>>
```


The *BlockMergeSort* class provides methods for sorting items partitioned across a CUDA thread block using a merge sorting method.

This example can be easily adapted to the storage required by *BlockMergeSort*.

Overview *BlockMergeSort* arranges items into ascending order using a comparison functor with less-than semantics. Merge sort can handle arbitrary types and comparison functors, but is slower than *BlockRadixSort* when sorting arithmetic types into ascending/descending order.

A Simple Example {*BlockMergeSort*}

The code snippet below illustrates a sort of 512 integer keys that are partitioned across 128 threads * where each thread owns 4 consecutive items.

```
#include <hipcub/hipcub.hpp> // or equivalently <hipcub/block/block_merge_sort.
↳hpp>

struct CustomLess
{
    template <typename DataType>
    __device__ bool operator()(const DataType &lhs, const DataType &rhs)
    {
        return lhs < rhs;
    }
};

__global__ void ExampleKernel(...)
{
    // Specialize BlockMergeSort for a 1D block of 128 threads owning 4 integer
    ↳items each
    typedef hipcub::BlockMergeSort<int, 128, 4> BlockMergeSort;

    // Allocate shared memory for BlockMergeSort
    __shared__ typename BlockMergeSort::TempStorage temp_storage_shuffle;

    // Obtain a segment of consecutive items that are blocked across threads
    int thread_keys[4];
    ...

    BlockMergeSort(temp_storage_shuffle).Sort(thread_keys, CustomLess());
    ...
}
```

Suppose the set of input `thread_keys` across the block of threads is { [0,511,1,510], [2,509,3,508], [4,507,5,506], ..., [254,257,255,256] }. The corresponding output `thread_keys` in those threads will be { [0,1,2,3], [4,5,6,7], [8,9,10,11], ..., [508,509,510,511] }.

Re-using dynamically allocating shared memory The following example under the examples/block folder illustrates usage of dynamically shared memory with *BlockReduce* and how to re-purpose the same memory region: `example_block_reduce_dyn_smem.cu`

Template Parameters

- **KeyT** – KeyT type
- **BLOCK_DIM_X** – The thread block length in threads along the X dimension
- **ITEMS_PER_THREAD** – The number of items per thread
- **ValueT** – [optional] ValueT type (default: `hipcub::NullType`, which indicates a keys-only sort)
- **BLOCK_DIM_Y** – [optional] The thread block length in threads along the Y dimension (default: 1)
- **BLOCK_DIM_Z** – [optional] The thread block length in threads along the Z dimension (default: 1)

Public Functions

```
__device__ __forceinline__ inline BlockMergeSort()  
  
__device__ __forceinline__ inline explicit BlockMergeSort(typename  
BlockMergeSortStrategyT::TempStorage  
&temp_storage)  
  
__device__ __forceinline__ inline unsigned int get_linear_tid() const  
  
__device__ __forceinline__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp  
compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

Sort is not guaranteed to be stable. That is, suppose that *i* and *j* are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.

Template Parameters **CompareOp** – functor type having member `bool operator() (KeyT lhs, KeyT rhs)`. **CompareOp** is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
__device__ __forceinline__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp  
compare_op, int valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- Sort is not guaranteed to be stable. That is, suppose that *i* and *j* are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.
- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. If there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters CompareOp – functor type having member `bool operator() (KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

```
__device__ __forceinline__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], ValueT
                                             (&items)[ITEMS_PER_THREAD], CompareOp compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

Sort is not guaranteed to be stable. That is, suppose that `i` and `j` are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.

Template Parameters CompareOp – functor type having member `bool operator() (KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
__device__ __forceinline__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], ValueT
                                             (&items)[ITEMS_PER_THREAD], CompareOp compare_op, int
                                             valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- Sort is not guaranteed to be stable. That is, suppose that `i` and `j` are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.
- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. If there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters

- **CompareOp** – functor type having member `bool operator() (KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).
- **IS_LAST_TILE** – True if `valid_items` isn't equal to the `ITEMS_PER_TILE`

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

```
__device__ __forceinline__ inline void StableSort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp  
compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

`StableSort` is stable: it preserves the relative ordering of equivalent elements. That is, if x and y are elements such that x precedes y , and if the two elements are equivalent (neither $x < y$ nor $y < x$) then a postcondition of `StableSort` is that x still precedes y .

Template Parameters `CompareOp` – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
__device__ __forceinline__ inline void StableSort(KeyT (&keys)[ITEMS_PER_THREAD], ValueT  
(&items)[ITEMS_PER_THREAD], CompareOp  
compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

`StableSort` is stable: it preserves the relative ordering of equivalent elements. That is, if x and y are elements such that x precedes y , and if the two elements are equivalent (neither $x < y$ nor $y < x$) then a postcondition of `StableSort` is that x still precedes y .

Template Parameters `CompareOp` – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
__device__ __forceinline__ inline void StableSort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp  
compare_op, int valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- StableSort is stable: it preserves the relative ordering of equivalent elements. That is, if x and y are elements such that x precedes y , and if the two elements are equivalent (neither $x < y$ nor $y < x$) then a postcondition of StableSort is that x still precedes y .
- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. If there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters `CompareOp` – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

```
__device__ __forceinline__ inline void StableSort(KeyT (&keys)[ITEMS_PER_THREAD], ValueT
(&items)[ITEMS_PER_THREAD], CompareOp
compare_op, int valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- StableSort is stable: it preserves the relative ordering of equivalent elements. That is, if x and y are elements such that x precedes y , and if the two elements are equivalent (neither $x < y$ nor $y < x$) then a postcondition of StableSort is that x still precedes y .
- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. If there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters

- **CompareOp** – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).
- **IS_LAST_TILE** – True if `valid_items` isn't equal to the `ITEMS_PER_TILE`

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

Template Class BlockMergeSortStrategy

- Defined in file `hipcub_backend_rocprim_block_block_merge_sort.hpp`

Nested Relationships

Nested Types

- *Struct* `BlockMergeSortStrategy::TempStorage`
- *Union* `BlockMergeSortStrategy::_TempStorage`

Class Documentation

template<typename **KeyT**, typename **ValueT**, int **NUM_THREADS**, int **ITEMS_PER_THREAD**, typename **SynchronizationPolicy**>

class `hipcub::BlockMergeSortStrategy`

Generalized merge sort algorithm.

This class is used to reduce code duplication. Warp and Block merge sort differ only in how they compute thread index and how they synchronize threads. Since synchronization might require access to custom data (like member mask), CRTP is used.

The code snippet below illustrates the way this class can be used.

```
#include <hipcub/hipcub.hpp> // or equivalently <hipcub/block/block_merge_sort.
↳hpp>

constexpr int BLOCK_THREADS = 256;
constexpr int ITEMS_PER_THREAD = 9;

class BlockMergeSort : public BlockMergeSortStrategy<int,
                                                    hipcub::NullType,
                                                    BLOCK_THREADS,
                                                    ITEMS_PER_THREAD,
                                                    BlockMergeSort>
{
    using BlockMergeSortStrategyT =
        BlockMergeSortStrategy<int,
                                hipcub::NullType,
                                BLOCK_THREADS,
                                ITEMS_PER_THREAD,
                                BlockMergeSort>;
public:
    __device__ __forceinline__ explicit BlockMergeSort(
        typename BlockMergeSortStrategyT::TempStorage &temp_storage)
        : BlockMergeSortStrategyT(temp_storage, threadIdx.x)
    {}
};
```

(continues on next page)

(continued from previous page)

```

__device__ __forceinline__ void SyncImplementation() const
{
    __syncthreads();
}
};

```

Template Parameters

- **KeyT** – KeyT type
- **ValueT** – ValueT type. hipcub::NullType indicates a keys-only sort
- **SynchronizationPolicy** – Provides a way of synchronizing threads. Should be derived from *BlockMergeSortStrategy*.

Public Functions**BlockMergeSortStrategy**() = delete

```
__device__ __forceinline__ inline explicit BlockMergeSortStrategy(unsigned int linear_tid)
```

```
__device__ __forceinline__ inline BlockMergeSortStrategy(TempStorage &temp_storage, unsigned int
linear_tid)
```

```
__device__ __forceinline__ inline unsigned int get_linear_tid() const
```

```
template<typename CompareOp>
```

```
__device__ __forceinline__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp
compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

Sort is not guaranteed to be stable. That is, suppose that *i* and *j* are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.

Template Parameters **CompareOp** – functor type having member `bool operator() (KeyT lhs, KeyT rhs)`. **CompareOp** is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
template<typename CompareOp>
```

```
__device__ __forceinline__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp
compare_op, int valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- Sort is not guaranteed to be stable. That is, suppose that *i* and *j* are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.

- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. *If* there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters `CompareOp` – functor type having member `bool operator() (KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

```
template<typename CompareOp>
__device__ __forceinline__ inline void Sort (KeyT (&keys)[ITEMS_PER_THREAD], ValueT
                                             (&items)[ITEMS_PER_THREAD], CompareOp compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

Sort is not guaranteed to be stable. That is, suppose that `i` and `j` are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.

Template Parameters `CompareOp` – functor type having member `bool operator() (KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
template<typename CompareOp, bool IS_LAST_TILE = true>
__device__ __forceinline__ inline void Sort (KeyT (&keys)[ITEMS_PER_THREAD], ValueT
                                             (&items)[ITEMS_PER_THREAD], CompareOp compare_op, int
                                             valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- Sort is not guaranteed to be stable. That is, suppose that `i` and `j` are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.
- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. *If* there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters

- **CompareOp** – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).
- **IS_LAST_TILE** – True if `valid_items` isn't equal to the `ITEMS_PER_TILE`

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

```
template<typename CompareOp>
__device__ __forceinline__ inline void StableSort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp
                                                    compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

`StableSort` is stable: it preserves the relative ordering of equivalent elements. That is, if `x` and `y` are elements such that `x` precedes `y`, and if the two elements are equivalent (neither `x < y` nor `y < x`) then a postcondition of `StableSort` is that `x` still precedes `y`.

Template Parameters **CompareOp** – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
template<typename CompareOp>
__device__ __forceinline__ inline void StableSort(KeyT (&keys)[ITEMS_PER_THREAD], ValueT
                                                    (&items)[ITEMS_PER_THREAD], CompareOp
                                                    compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

`StableSort` is stable: it preserves the relative ordering of equivalent elements. That is, if `x` and `y` are elements such that `x` precedes `y`, and if the two elements are equivalent (neither `x < y` nor `y < x`) then a postcondition of `StableSort` is that `x` still precedes `y`.

Template Parameters **CompareOp** – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort

- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
template<typename CompareOp>
__device__ __forceinline__ inline void StableSort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp
compare_op, int valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- StableSort is stable: it preserves the relative ordering of equivalent elements. That is, if x and y are elements such that x precedes y , and if the two elements are equivalent (neither $x < y$ nor $y < x$) then a postcondition of StableSort is that x still precedes y .
- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. *If* there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters CompareOp – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. CompareOp is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

```
template<typename CompareOp, bool IS_LAST_TILE = true>
__device__ __forceinline__ inline void StableSort(KeyT (&keys)[ITEMS_PER_THREAD], ValueT
(&items)[ITEMS_PER_THREAD], CompareOp
compare_op, int valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- StableSort is stable: it preserves the relative ordering of equivalent elements. That is, if x and y are elements such that x precedes y , and if the two elements are equivalent (neither $x < y$ nor $y < x$) then a postcondition of StableSort is that x still precedes y .
- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. *If* there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters

- **CompareOp** – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. CompareOp is a model of [Strict Weak Ordering](#).

- **IS_LAST_TILE** – True if `valid_items` isn't equal to the `ITEMS_PER_TILE`

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

```
struct TempStorage : public hipcub::Uninitialized<_TempStorage>
    {BlockMergeSort}
```

Public Types

```
enum [anonymous]
```

Values:

```
typedef UnitWord<_TempStorage>::DeviceWord DeviceWord
```

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

```
__host__ __device__ __forceinline__ inline _TempStorage &Alias()
```

Alias.

Public Members

```
DeviceWord storage[WORDS]
```

Backing storage.

Template Class **BlockRadixRank**

- Defined in file `hipcub_backend_rocprim_block_block_radix_rank.hpp`

Nested Relationships

Nested Types

- *Struct* `BlockRadixRank::PrefixCallback`
- *Struct* `BlockRadixRank::TempStorage`

Class Documentation

```
template<int BLOCK_DIM_X, int RADIX_BITS, bool IS_DESCENDING, bool MEMOIZE_OUTER_SCAN = false,
BlockScanAlgorithm INNER_SCAN_ALGORITHM = BLOCK_SCAN_WARP_SCANS, hipSharedMemConfig
SMEM_CONFIG = hipSharedMemBankSizeFourByte, int BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int ARCH = 1>
class hipcub::BlockRadixRank
```

BlockRadixRank provides operations for ranking unsigned integer types within a CUDA thread block.

Overview Blah...

- Keys must be in a form suitable for radix ranking (i.e., unsigned bits).
-

Performance Considerations

-

Examples

- **Example 1:** Simple radix rank of 32-bit integer keys

```
#include <hipcub/hipcub.hpp>

template <int BLOCK_THREADS>
__global__ void ExampleKernel(...)
{
```

Template Parameters

- **BLOCK_DIM_X** – The thread block length in threads along the X dimension
- **RADIX_BITS** – The number of radix bits per digit place
- **IS_DESCENDING** – Whether or not the sorted-order is high-to-low
- **MEMOIZE_OUTER_SCAN** – [optional] Whether or not to buffer outer raking scan partials to incur fewer shared memory reads at the expense of higher register pressure (default: true for architectures SM35 and newer, false otherwise). See `BlockScanAlgorithm::BLOCK_SCAN_RAKING_MEMOIZE` for more details.
- **INNER_SCAN_ALGORITHM** – [optional] The `hipcub::BlockScanAlgorithm` algorithm to use (default: `hipcub::BLOCK_SCAN_WARP_SCANS`)
- **SMEM_CONFIG** – [optional] Shared memory bank mode (default: `hipSharedMemBankSizeFourByte`)
- **BLOCK_DIM_Y** – [optional] The thread block length in threads along the Y dimension (default: 1)
- **BLOCK_DIM_Z** – [optional] The thread block length in threads along the Z dimension (default: 1)
- **ARCH** – [optional]

Collective constructors

```
__device__ inline BlockRadixRank()
```

Collective constructor using a private static allocation of shared memory as temporary storage.

```
__device__ inline BlockRadixRank(TempStorage &temp_storage)
```

Collective constructor using the specified memory allocation as temporary storage.

Parameters `temp_storage` – Reference to memory allocation having layout type *TempStorage*

Raking

```
template<typename UnsignedBits, int KEYS_PER_THREAD, typename DigitExtractorT>
__device__ inline void RankKeys(UnsignedBits (&keys)[KEYS_PER_THREAD], int
                                (&ranks)[KEYS_PER_THREAD], DigitExtractorT digit_extractor)
```

Rank keys.

Parameters

- **keys** – Keys for this tile
- **ranks** – For each key, the local rank within the tile
- **digit_extractor** – The digit extractor

```
template<typename UnsignedBits, int KEYS_PER_THREAD, typename DigitExtractorT>
__device__ inline void RankKeys(UnsignedBits (&keys)[KEYS_PER_THREAD], int
                                (&ranks)[KEYS_PER_THREAD], DigitExtractorT digit_extractor, int
                                (&exclusive_digit_prefix)[BINS_TRACKED_PER_THREAD])
```

Rank keys. For the lower `RADIX_DIGITS` threads, digit counts for each digit are provided for the corresponding thread.

Parameters

- **keys** – Keys for this tile
- **ranks** – For each key, the local rank within the tile (out parameter)
- **digit_extractor** – The digit extractor
- **exclusive_digit_prefix** – The exclusive prefix sum for the digits $[(\text{threadIdx.x} * \text{BINS_TRACKED_PER_THREAD}) \dots (\text{threadIdx.x} * \text{BINS_TRACKED_PER_THREAD}) + \text{BINS_TRACKED_PER_THREAD} - 1]$

Public Types

```
enum [anonymous]
```

Values:

```
enumerator BINS_TRACKED_PER_THREAD
```

Number of bin-starting offsets tracked per thread.

```
struct TempStorage : public hipcub::Uninitialized<_TempStorage>
{ BlockScan }
```

Public Types

enum [**anonymous**]

Values:

typedef UnitWord<_TempStorage>::DeviceWord **DeviceWord**

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

__host__ __device__ __forceinline__ inline _TempStorage &**Alias**()

Alias.

Public Members

DeviceWord **storage**[WORDS]

Backing storage.

Template Class BlockRadixRankMatch

- Defined in file `hipcub_backend_rocprim_block_block_radix_rank.hpp`

Nested Relationships

Nested Types

- *Struct* `BlockRadixRankMatch::TempStorage`

Class Documentation

```
template<int BLOCK_DIM_X, int RADIX_BITS, bool IS_DESCENDING, BlockScanAlgorithm INNER_SCAN_ALGORITHM  
= BLOCK_SCAN_WARP_SCANS, int BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int ARCH = 1>
```

```
class hipcub:: BlockRadixRankMatch
```

Radix-rank using match.any

Collective constructors

```
__device__ inline BlockRadixRankMatch(TempStorage &temp_storage)
```

Collective constructor using the specified memory allocation as temporary storage.

Parameters `temp_storage` – Reference to memory allocation having layout type *TempStorage*

Raking

```
template<typename UnsignedBits, int KEYS_PER_THREAD, typename DigitExtractorT>
__device__ __forceinline__ inline void RankKeys(UnsignedBits (&keys)[KEYS_PER_THREAD], int
                                                (&ranks)[KEYS_PER_THREAD], DigitExtractorT
                                                digit_extractor)
```

Rank keys.

Parameters

- **keys** – Keys for this tile
- **ranks** – For each key, the local rank within the tile
- **digit_extractor** – The digit extractor

```
template<typename UnsignedBits, int KEYS_PER_THREAD, typename DigitExtractorT>
__device__ __forceinline__ inline void RankKeys(UnsignedBits (&keys)[KEYS_PER_THREAD], int
                                                (&ranks)[KEYS_PER_THREAD], DigitExtractorT
                                                digit_extractor, int (&exclu-
                                                sive_digit_prefix)[BINS_TRACKED_PER_THREAD])
```

Rank keys. For the lower RADIX_DIGITS threads, digit counts for each digit are provided for the corresponding thread.

Parameters

- **keys** – Keys for this tile
- **ranks** – For each key, the local rank within the tile (out parameter)
- **digit_extractor** – The digit extractor
- **exclusive_digit_prefix** – The exclusive prefix sum for the digits $[(\text{threadIdx.x} * \text{BINS_TRACKED_PER_THREAD}) \dots (\text{threadIdx.x} * \text{BINS_TRACKED_PER_THREAD}) + \text{BINS_TRACKED_PER_THREAD} - 1]$

Public Types

```
enum [anonymous]
```

Values:

```
enumerator BINS_TRACKED_PER_THREAD
```

Number of bin-starting offsets tracked per thread.

```
struct TempStorage : public hipcub::Uninitialized<_TempStorage>
{BlockScan}
```

Public Types

enum [**anonymous**]

Values:

typedef UnitWord<_TempStorage>::DeviceWord **DeviceWord**

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

`__host__ __device__ __forceinline__ inline` _TempStorage &**Alias**()

Alias.

Public Members

DeviceWord **storage**[WORDS]

Backing storage.

Template Class BlockRadixSort

- Defined in file `_hipcub_backend_rocprim_block_block_radix_sort.hpp`

Inheritance Relationships

Base Type

- `private rocprim::block_radix_sort< KeyT, BLOCK_DIM_X, ITEMS_PER_THREAD, ValueT, BLOCK_DIM_Y, BLOCK_DIM_Z >`

Class Documentation

```
template<typename KeyT, int BLOCK_DIM_X, int ITEMS_PER_THREAD, typename ValueT = NullType, int
RADIX_BITS = 4, bool MEMOIZE_OUTER_SCAN = true, BlockScanAlgorithm INNER_SCAN_ALGORITHM =
BLOCK_SCAN_WARP_SCANS, hipSharedMemConfig SMEM_CONFIG = hipSharedMemBankSizeFourByte, int
BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int PTX_ARCH = HIPCUB_ARCH>
class hipcub:: BlockRadixSort : private rocprim::block_radix_sort<KeyT, BLOCK_DIM_X,
ITEMS_PER_THREAD, ValueT, BLOCK_DIM_Y, BLOCK_DIM_Z>
```


Public Types

using **TempStorage** = typename base_type::storage_type

Public Functions

`__device__ inline BlockRadixSort()`

`__device__ inline BlockRadixSort(TempStorage &temp_storage)`

`__device__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], int begin_bit = 0, int end_bit = sizeof(KeyT) * 8)`

`__device__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], ValueT (&values)[ITEMS_PER_THREAD], int begin_bit = 0, int end_bit = sizeof(KeyT) * 8)`

`__device__ inline void SortDescending(KeyT (&keys)[ITEMS_PER_THREAD], int begin_bit = 0, int end_bit = sizeof(KeyT) * 8)`

`__device__ inline void SortDescending(KeyT (&keys)[ITEMS_PER_THREAD], ValueT (&values)[ITEMS_PER_THREAD], int begin_bit = 0, int end_bit = sizeof(KeyT) * 8)`

`__device__ inline void SortBlockedToStriped(KeyT (&keys)[ITEMS_PER_THREAD], int begin_bit = 0, int end_bit = sizeof(KeyT) * 8)`

`__device__ inline void SortBlockedToStriped(KeyT (&keys)[ITEMS_PER_THREAD], ValueT (&values)[ITEMS_PER_THREAD], int begin_bit = 0, int end_bit = sizeof(KeyT) * 8)`

`__device__ inline void SortDescendingBlockedToStriped(KeyT (&keys)[ITEMS_PER_THREAD], int begin_bit = 0, int end_bit = sizeof(KeyT) * 8)`

`__device__ inline void SortDescendingBlockedToStriped(KeyT (&keys)[ITEMS_PER_THREAD], ValueT (&values)[ITEMS_PER_THREAD], int begin_bit = 0, int end_bit = sizeof(KeyT) * 8)`

Template Class BlockRunLengthDecode

- Defined in file `hipcub_backend_rocprim_block_block_run_length_decode.hpp`

Nested Relationships

Nested Types

- *Struct* `BlockRunLengthDecode::TempStorage`
- *Union* `BlockRunLengthDecode::_TempStorage`

Class Documentation

```
template<typename ItemT, int BLOCK_DIM_X, int RUNS_PER_THREAD, int DECODED_ITEMS_PER_THREAD, typename
DecodedOffsetT = uint32_t, int BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1>
```

```
class hipcub::BlockRunLengthDecode
```

The *BlockRunLengthDecode* class supports decoding a run-length encoded array of items. That is, given the two arrays `run_value[N]` and `run_lengths[N]`, `run_value[i]` is repeated `run_lengths[i]` many times in the output array. Due to the nature of the run-length decoding algorithm (“decompression”), the output size of the run-length decoded array is runtime-dependent and potentially without any upper bound. To address this, *BlockRunLengthDecode* allows retrieving a “window” from the run-length decoded array. The window’s offset can be specified and `BLOCK_THREADS * DECODED_ITEMS_PER_THREAD` (i.e., referred to as `window_size`) decoded items from the specified window will be returned.

```
__global__ void ExampleKernel(...)
{
    // Specialising BlockRunLengthDecode to run-length decode items of type
    ↪uint64_t
    using RunItemT = uint64_t;
    // Type large enough to index into the run-length decoded array
    using RunLengthT = uint32_t;

    // Specialising BlockRunLengthDecode for a 1D block of 128 threads
    constexpr int BLOCK_DIM_X = 128;
    // Specialising BlockRunLengthDecode to have each thread contribute 2 run-
    ↪length encoded runs
    constexpr int RUNS_PER_THREAD = 2;
    // Specialising BlockRunLengthDecode to have each thread hold 4 run-length
    ↪decoded items
    constexpr int DECODED_ITEMS_PER_THREAD = 4;

    // Specialize BlockRadixSort for a 1D block of 128 threads owning 4 integer
    ↪items each
    using BlockRunLengthDecodeT =
        hipcub::BlockRunLengthDecode<RunItemT, BLOCK_DIM_X, RUNS_PER_THREAD,
    ↪DECODED_ITEMS_PER_THREAD>;

    // Allocate shared memory for BlockRunLengthDecode
    __shared__ typename BlockRunLengthDecodeT::TempStorage temp_storage;

    // The run-length encoded items and how often they shall be repeated in the
    ↪run-length decoded output
    RunItemT run_values[RUNS_PER_THREAD];
    RunLengthT run_lengths[RUNS_PER_THREAD];
    ...

    // Initialize the BlockRunLengthDecode with the runs that we want to run-
    ↪length decode
    uint32_t total_decoded_size = 0;
    BlockRunLengthDecodeT block_rld(temp_storage, run_values, run_lengths, total_
    ↪decoded_size);
```

(continues on next page)

(continued from previous page)

```

// Run-length decode ("decompress") the runs into a window buffer of limited_
↳size. This is repeated until all runs
// have been decoded.
uint32_t decoded_window_offset = 0U;
while (decoded_window_offset < total_decoded_size)
{
    RunLengthT relative_offsets[DECODED_ITEMS_PER_THREAD];
    RunItemT decoded_items[DECODED_ITEMS_PER_THREAD];

    // The number of decoded items that are valid within this window (aka pass)_
↳of run-length decoding
    uint32_t num_valid_items = total_decoded_size - decoded_window_offset;
    block_rld.RunLengthDecode(decoded_items, relative_offsets, decoded_window_
↳offset);

    decoded_window_offset += BLOCK_DIM_X * DECODED_ITEMS_PER_THREAD;

    ...
}
}

```

Suppose the set of input `run_values` across the block of threads is { [0, 1], [2, 3], [4, 5], [6, 7], ..., [254, 255] } and `run_lengths` is { [1, 2], [3, 4], [5, 1], [2, 3], ..., [5, 1] }. The corresponding output `decoded_items` in those threads will be { [0, 1, 1, 2], [2, 2, 3, 3], [3, 3, 4, 4], [4, 4, 4, 5], ..., [169, 169, 170, 171] } and `relative_offsets` will be { [0, 0, 1, 0], [1, 2, 0, 1], [2, 3, 0, 1], [2, 3, 4, 0], ..., [3, 4, 0, 0] } during the first iteration of the while loop.

Note: : Trailing runs of length 0 are supported (i.e., they may only appear at the end of the `run_lengths` array). A run of length zero may not be followed by a run length that is not zero.

Template Parameters

- **ItemT** – The data type of the items being run-length decoded
- **BLOCK_DIM_X** – The thread block length in threads along the X dimension
- **RUNS_PER_THREAD** – The number of consecutive runs that each thread contributes
- **DECODED_ITEMS_PER_THREAD** – The maximum number of decoded items that each thread holds
- **DecodedOffsetT** – Type used to index into the block’s decoded items (large enough to hold the sum over all the runs’ lengths)
- **BLOCK_DIM_Y** – The thread block length in threads along the Y dimension
- **BLOCK_DIM_Z** – The thread block length in threads along the Z dimension

Public Functions

```
template<typename RunLengthT, typename TotalDecodedSizeT>
__device__ __forceinline__ inline BlockRunLengthDecode(TempStorage &temp_storage, ItemT
    (&run_values)[RUNS_PER_THREAD],
    RunLengthT
    (&run_lengths)[RUNS_PER_THREAD],
    TotalDecodedSizeT &total_decoded_size)
```

Constructor specialised for user-provided temporary storage, initializing using the runs' lengths. The algorithm's temporary storage may not be repurposed between the constructor call and subsequent **RunLengthDecode** calls.

```
template<typename UserRunOffsetT>
__device__ __forceinline__ inline BlockRunLengthDecode(TempStorage &temp_storage, ItemT
    (&run_values)[RUNS_PER_THREAD],
    UserRunOffsetT
    (&run_offsets)[RUNS_PER_THREAD])
```

Constructor specialised for user-provided temporary storage, initializing using the runs' offsets. The algorithm's temporary storage may not be repurposed between the constructor call and subsequent **RunLengthDecode** calls.

```
template<typename RunLengthT, typename TotalDecodedSizeT>
__device__ __forceinline__ inline BlockRunLengthDecode(ItemT (&run_values)[RUNS_PER_THREAD],
    RunLengthT
    (&run_lengths)[RUNS_PER_THREAD],
    TotalDecodedSizeT &total_decoded_size)
```

Constructor specialised for static temporary storage, initializing using the runs' lengths.

```
template<typename UserRunOffsetT>
__device__ __forceinline__ inline BlockRunLengthDecode(ItemT (&run_values)[RUNS_PER_THREAD],
    UserRunOffsetT
    (&run_offsets)[RUNS_PER_THREAD])
```

Constructor specialised for static temporary storage, initializing using the runs' offsets.

```
template<typename RelativeOffsetT>
__device__ __forceinline__ inline void RunLengthDecode(ItemT (&de-
    coded_items)[DECODED_ITEMS_PER_THREAD],
    RelativeOffsetT
    (&item_offsets)[DECODED_ITEMS_PER_THREAD],
    DecodedOffsetT from_decoded_offset = 0)
```

Run-length decodes the runs previously passed via a call to `Init(...)` and returns the run-length decoded items in a blocked arrangement to `decoded_items`. If the number of run-length decoded items exceeds the run-length decode buffer (i.e., `DECODED_ITEMS_PER_THREAD * BLOCK_THREADS`), only the items that fit within the buffer are returned. Subsequent calls to **RunLengthDecode** adjusting `from_decoded_offset` can be used to retrieve the remaining run-length decoded items. Calling `__syncthreads()` between any two calls to **RunLengthDecode** is not required. `item_offsets` can be used to retrieve each run-length decoded item's relative index within its run. E.g., the run-length encoded array of 3, 1, 4 with the respective run lengths of 2, 1, 3 would yield the run-length decoded array of 3, 3, 1, 4, 4, 4 with the relative offsets of 0, 1, 0, 0, 1, 2.

Parameters

- **decoded_items** – [out] The run-length decoded items to be returned in a blocked arrangement

- **item_offsets** – [out] The run-length decoded items' relative offset within the run they belong to
- **from_decoded_offset** – [in] *If* invoked with `from_decoded_offset` that is larger than `total_decoded_size` results in undefined behavior.

```
__device__ __forceinline__ inline void RunLengthDecode(ItemT (&de-
    coded_items)[DECODED_ITEMS_PER_THREAD],
    DecodedOffsetT from_decoded_offset = 0)
```

Run-length decodes the runs previously passed via a call to `Init(...)` and returns the run-length decoded items in a blocked arrangement to `decoded_items`. *If* the number of run-length decoded items exceeds the run-length decode buffer (i.e., `DECODED_ITEMS_PER_THREAD * BLOCK_THREADS`), only the items that fit within the buffer are returned. Subsequent calls to `RunLengthDecode` adjusting `from_decoded_offset` can be used to retrieve the remaining run-length decoded items. Calling `__sync_threads()` between any two calls to `RunLengthDecode` is not required.

Parameters

- **decoded_items** – [out] The run-length decoded items to be returned in a blocked arrangement
- **from_decoded_offset** – [in] *If* invoked with `from_decoded_offset` that is larger than `total_decoded_size` results in undefined behavior.

```
struct TempStorage : public hipcub::Uninitialized<_TempStorage>
```

Public Types

```
enum [anonymous]
```

Values:

```
typedef UnitWord<_TempStorage>::DeviceWord DeviceWord
```

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

```
__host__ __device__ __forceinline__ inline _TempStorage &Alias()
```

Alias.

Public Members

```
DeviceWord storage[WORDS]
```

Backing storage.

Template Class BlockScan

- Defined in file_hipcub_backend_rocprim_block_block_scan.hpp

Inheritance Relationships

Base Type

- private rocprim::block_scan< T, BLOCK_DIM_X, static_cast<::rocprim::block_scan_algorithm>(ALGORITHM), BLOCK_DIM_Y, BLOCK_DIM_Z >

Class Documentation

```
template<typename T, int BLOCK_DIM_X, BlockScanAlgorithm ALGORITHM = BLOCK_SCAN_RAKING, int
BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int ARCH = HIPCUB_ARCH>
class hipcub::BlockScan : private rocprim::block_scan<T, BLOCK_DIM_X,
static_cast<::rocprim::block_scan_algorithm>(ALGORITHM), BLOCK_DIM_Y, BLOCK_DIM_Z>
```

Public Types

```
using TempStorage = typename base_type::storage_type
```

Public Functions

```
__device__ inline BlockScan()
```

```
__device__ inline BlockScan(TempStorage &temp_storage)
```

```
__device__ inline void InclusiveSum(T input, T &output)
```

```
__device__ inline void InclusiveSum(T input, T &output, T &block_aggregate)
```

```
template<typename BlockPrefixCallbackOp>
```

```
__device__ inline void InclusiveSum(T input, T &output, BlockPrefixCallbackOp
&block_prefix_callback_op)
```

```
template<int ITEMS_PER_THREAD>
```

```
__device__ inline void InclusiveSum(T (&input)[ITEMS_PER_THREAD], T
(&output)[ITEMS_PER_THREAD])
```

```
template<int ITEMS_PER_THREAD>
```

```
__device__ inline void InclusiveSum(T (&input)[ITEMS_PER_THREAD], T
(&output)[ITEMS_PER_THREAD], T &block_aggregate)
```

```
template<int ITEMS_PER_THREAD, typename BlockPrefixCallbackOp>
```

```
__device__ inline void InclusiveSum(T (&input)[ITEMS_PER_THREAD], T
(&output)[ITEMS_PER_THREAD], BlockPrefixCallbackOp
&block_prefix_callback_op)
```

```
template<typename ScanOp>
```

```

__device__ inline void InclusiveScan(T input, T &output, ScanOp scan_op)

template<typename ScanOp>
__device__ inline void InclusiveScan(T input, T &output, ScanOp scan_op, T &block_aggregate)

template<typename ScanOp, typename BlockPrefixCallbackOp>
__device__ inline void InclusiveScan(T input, T &output, ScanOp scan_op, BlockPrefixCallbackOp
&block_prefix_callback_op)

template<int ITEMS_PER_THREAD, typename ScanOp>
__device__ inline void InclusiveScan(T (&input)[ITEMS_PER_THREAD], T
(&output)[ITEMS_PER_THREAD], ScanOp scan_op)

template<int ITEMS_PER_THREAD, typename ScanOp>
__device__ inline void InclusiveScan(T (&input)[ITEMS_PER_THREAD], T
(&output)[ITEMS_PER_THREAD], ScanOp scan_op, T
&block_aggregate)

template<int ITEMS_PER_THREAD, typename ScanOp, typename BlockPrefixCallbackOp>
__device__ inline void InclusiveScan(T (&input)[ITEMS_PER_THREAD], T
(&output)[ITEMS_PER_THREAD], ScanOp scan_op,
BlockPrefixCallbackOp &block_prefix_callback_op)

__device__ inline void ExclusiveSum(T input, T &output)

__device__ inline void ExclusiveSum(T input, T &output, T &block_aggregate)

template<typename BlockPrefixCallbackOp>
__device__ inline void ExclusiveSum(T input, T &output, BlockPrefixCallbackOp
&block_prefix_callback_op)

template<int ITEMS_PER_THREAD>
__device__ inline void ExclusiveSum(T (&input)[ITEMS_PER_THREAD], T
(&output)[ITEMS_PER_THREAD])

template<int ITEMS_PER_THREAD>
__device__ inline void ExclusiveSum(T (&input)[ITEMS_PER_THREAD], T
(&output)[ITEMS_PER_THREAD], T &block_aggregate)

template<int ITEMS_PER_THREAD, typename BlockPrefixCallbackOp>
__device__ inline void ExclusiveSum(T (&input)[ITEMS_PER_THREAD], T
(&output)[ITEMS_PER_THREAD], BlockPrefixCallbackOp
&block_prefix_callback_op)

template<typename ScanOp>
__device__ inline void ExclusiveScan(T input, T &output, T initial_value, ScanOp scan_op)

template<typename ScanOp>
__device__ inline void ExclusiveScan(T input, T &output, T initial_value, ScanOp scan_op, T
&block_aggregate)

template<typename ScanOp, typename BlockPrefixCallbackOp>
__device__ inline void ExclusiveScan(T input, T &output, ScanOp scan_op, BlockPrefixCallbackOp
&block_prefix_callback_op)

template<int ITEMS_PER_THREAD, typename ScanOp>

```

```
__device__ inline void ExclusiveScan(T (&input)[ITEMS_PER_THREAD], T
                                       (&output)[ITEMS_PER_THREAD], T initial_value, ScanOp
                                       scan_op)

template<int ITEMS_PER_THREAD, typename ScanOp>
__device__ inline void ExclusiveScan(T (&input)[ITEMS_PER_THREAD], T
                                       (&output)[ITEMS_PER_THREAD], T initial_value, ScanOp
                                       scan_op, T &block_aggregate)

template<int ITEMS_PER_THREAD, typename ScanOp, typename BlockPrefixCallbackOp>
__device__ inline void ExclusiveScan(T (&input)[ITEMS_PER_THREAD], T
                                       (&output)[ITEMS_PER_THREAD], ScanOp scan_op,
                                       BlockPrefixCallbackOp &block_prefix_callback_op)
```

Template Class BlockShuffle

- Defined in file `hipcub_backend_rocprim_block_block_shuffle.hpp`

Inheritance Relationships

Base Type

- `public rocprim::block_shuffle< T, BLOCK_DIM_X, BLOCK_DIM_Y, BLOCK_DIM_Z >`

Class Documentation

```
template<typename T, int BLOCK_DIM_X, int BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int ARCH = HIPCUB_ARCH>
class hipcub::BlockShuffle : public rocprim::block_shuffle<T, BLOCK_DIM_X, BLOCK_DIM_Y,
BLOCK_DIM_Z>
```

Public Types

```
using TempStorage = typename base_type::storage_type
```

Public Functions

```
__device__ inline BlockShuffle()
__device__ inline BlockShuffle(TempStorage &temp_storage)
```

Parameters `temp_storage` – Reference to memory allocation having layout type `TempStorage`

```
__device__ inline void Offset(T input, T &output, int distance = 1)
```

Each *thread* obtains the input provided by *thread*₊. The offset distance may be negative.

•

Parameters

- **input** – The input item from the calling thread (*thread*)
- **output** – The **input** item from the successor (or predecessor) thread *thread₊* (may be aliased to **input**). This value is only updated for for *thread* when $0 \leq (i + \text{distance}) < \text{BLOCK_THREADS} - 1$
- **distance** – Offset distance (may be negative)

```
__device__ inline void Rotate(T input, T &output, unsigned int distance = 1)
```

Each *thread* obtains the input provided by *thread₊*.

•

Parameters

- **input** – The calling thread's input item
- **output** – The **input** item from thread *thread_{(+)%}* (may be aliased to **input**). This value is not updated for *thread_{BLOCK_THREADS-1}*
- **distance** – Offset distance ($0 < \text{distance} < \text{BLOCK_THREADS}$)

```
template<int ITEMS_PER_THREAD>
```

```
__device__ inline void Up(T (&input)[ITEMS_PER_THREAD], T (&prev)[ITEMS_PER_THREAD])
```

The thread block rotates its of **input** items, shifting it up by one item.

•

•

•

Parameters

- **input** – The calling thread's input items
- **prev** – The corresponding predecessor items (may be aliased to **input**). The item `prev[0]` is not updated for *thread₀*.

```
template<int ITEMS_PER_THREAD>
```

```
__device__ inline void Up(T (&input)[ITEMS_PER_THREAD], T (&prev)[ITEMS_PER_THREAD], T  
    &block_suffix)
```

The thread block rotates its of **input** items, shifting it up by one item. All threads receive the **input** provided by *thread*.

•

•

•

Parameters

- **input** – The calling thread’s input items
- **prev** – The corresponding predecessor items (may be aliased to **input**). The item `prev[0]` is not updated for *thread*₀.
- **block_suffix** – The item `input[ITEMS_PER_THREAD-1]` from *thread*, provided to all threads

```
template<int ITEMS_PER_THREAD>
__device__ inline void Down(T (&input)[ITEMS_PER_THREAD], T (&next)[ITEMS_PER_THREAD])
```

The thread block rotates its of `input` items, shifting it down by one item.

-
-
-

Parameters

- **input** – The calling thread’s input items
- **next** – The corresponding predecessor items (may be aliased to **input**). The value `next[0]` is not updated for *thread*_{BLOCK_THREADS-1}.

```
template<int ITEMS_PER_THREAD>
__device__ inline void Down(T (&input)[ITEMS_PER_THREAD], T (&next)[ITEMS_PER_THREAD], T
&block_prefix)
```

The thread block rotates its of `input` items, shifting it down by one item. All threads receive `input[0]` provided by *thread*.

-
-
-

Parameters

- **input** – The calling thread’s input items
- **next** – The corresponding predecessor items (may be aliased to **input**). The value `next[0]` is not updated for *thread*_{BLOCK_THREADS-1}.
- **block_prefix** – The item `input[0]` from *thread*, provided to all threads

Template Class BlockStore

- Defined in file_hipcub_backend_rocprim_block_block_store.hpp

Inheritance Relationships

Base Type

- `private rocprim::block_store< T, BLOCK_DIM_X, ITEMS_PER_THREAD, static_cast<::rocprim::block_store_method >(ALGORITHM), BLOCK_DIM_Y, BLOCK_DIM_Z >`

Class Documentation

```
template<typename T, int BLOCK_DIM_X, int ITEMS_PER_THREAD, BlockStoreAlgorithm ALGORITHM =
BLOCK_STORE_DIRECT, int BLOCK_DIM_Y = 1, int BLOCK_DIM_Z = 1, int ARCH = HIPCUB_ARCH>
class hipcub::BlockStore : private rocprim::block_store<T, BLOCK_DIM_X, ITEMS_PER_THREAD,
static_cast<::rocprim::block_store_method>(ALGORITHM), BLOCK_DIM_Y, BLOCK_DIM_Z>
```

Public Types

```
using TempStorage = typename base_type::storage_type
```

Public Functions

```
__device__ inline BlockStore()
```

```
__device__ inline BlockStore(TempStorage &temp_storage)
```

```
template<class OutputIteratorT>
__device__ inline void Store(OutputIteratorT block_iter, T (&items)[ITEMS_PER_THREAD])
```

```
template<class OutputIteratorT>
__device__ inline void Store(OutputIteratorT block_iter, T (&items)[ITEMS_PER_THREAD], int
valid_items)
```

Template Class CacheModifiedInputIterator

- Defined in file_hipcub_backend_rocprim_iterator_cache_modified_input_iterator.hpp

Class Documentation

```
template<CacheLoadModifier MODIFIER, typename ValueType, typename OffsetT = ptrdiff_t>
```

```
class hipcub::CacheModifiedInputIterator
```

Public Types

```
typedef CacheModifiedInputIterator self_type
```

My own type.

```
typedef OffsetT difference_type
```

Type to express the result of subtracting one iterator from another.

```
typedef ValueType value_type
```

The type of the element the iterator can point to.

```
typedef ValueType *pointer
```

The type of a pointer to an element the iterator can point to.

```
typedef ValueType reference
```

The type of a reference to an element the iterator can point to.

```
typedef std::random_access_iterator_tag iterator_category
```

The iterator category.

Public Functions

```
__host__ __device__ __forceinline__ inline CacheModifiedInputIterator(ValueType *ptr)
```

Constructor.

Parameters *ptr* – Native pointer to wrap

```
__host__ __device__ __forceinline__ inline self_type operator++(int)
```

Postfix increment.

```
__host__ __device__ __forceinline__ inline self_type operator++()
```

Prefix increment.

```
__device__ __forceinline__ inline reference operator*() const
```

Indirection.

```
template<typename Distance>
```

```
__host__ __device__ __forceinline__ inline self_type operator+(Distance n) const
```

Addition.

```
template<typename Distance>
```

```
__host__ __device__ __forceinline__ inline self_type &operator+=(Distance n)
```

Addition assignment.

```
template<typename Distance>
```

```
__host__ __device__ __forceinline__ inline self_type operator-(Distance n) const
```

Subtraction.

```
template<typename Distance>
__host__ __device__ __forceinline__ inline self_type &operator-=(Distance n)
```

Subtraction assignment.

```
__host__ __device__ __forceinline__ inline difference_type operator-(self_type other) const
```

Distance.

```
template<typename Distance>
__device__ __forceinline__ inline reference operator[](Distance n) const
```

Array subscript.

```
__device__ __forceinline__ inline pointer operator->()
```

Structure dereference.

```
__host__ __device__ __forceinline__ inline bool operator==(const self_type &rhs)
```

Equal to.

```
__host__ __device__ __forceinline__ inline bool operator!=(const self_type &rhs)
```

Not equal to.

Public Members

ValueType ***ptr**

Wrapped native pointer.

Template Class CacheModifiedOutputIterator

- Defined in file `hipcub_backend_rocprim_iterator_cache_modified_output_iterator.hpp`

Nested Relationships

Nested Types

- *Struct* `CacheModifiedOutputIterator::Reference`

Class Documentation

```
template<CacheStoreModifier MODIFIER, typename ValueType, typename OffsetT = ptrdiff_t>
class hipcub::CacheModifiedOutputIterator
```

Public Types

typedef *CacheModifiedOutputIterator* **self_type**

My own type.

typedef *OffsetT* **difference_type**

Type to express the result of subtracting one iterator from another.

typedef void **value_type**

The type of the element the iterator can point to.

typedef void **pointer**

The type of a pointer to an element the iterator can point to.

typedef Reference **reference**

The type of a reference to an element the iterator can point to.

typedef std::random_access_iterator_tag **iterator_category**

The iterator category.

Public Functions

template<typename **QualifiedValueType**>

`__host__ __device__ __forceinline__ inline CacheModifiedOutputIterator(QualifiedValueType *ptr)`

Constructor.

Parameters *ptr* – Native pointer to wrap

`__host__ __device__ __forceinline__ inline self_type operator++(int)`

Postfix increment.

`__host__ __device__ __forceinline__ inline self_type operator++()`

Prefix increment.

`__host__ __device__ __forceinline__ inline reference operator*() const`

Indirection.

template<typename **Distance**>

`__host__ __device__ __forceinline__ inline self_type operator+(Distance n) const`

Addition.

template<typename **Distance**>

`__host__ __device__ __forceinline__ inline self_type &operator+=(Distance n)`

Addition assignment.

template<typename **Distance**>

`__host__ __device__ __forceinline__ inline self_type operator-(Distance n) const`

Subtraction.

template<typename **Distance**>

```

__host__ __device__ __forceinline__ inline self_type &operator==(Distance n)
    Subtraction assignment.

__host__ __device__ __forceinline__ inline difference_type operator-(self_type other) const
    Distance.

template<typename Distance>
__host__ __device__ __forceinline__ inline reference operator[](Distance n) const
    Array subscript.

__host__ __device__ __forceinline__ inline bool operator==(const self_type &rhs)
    Equal to.

__host__ __device__ __forceinline__ inline bool operator!=(const self_type &rhs)
    Not equal to.

```

Class `CachingDeviceAllocator::TotalBytes`

- Defined in file `hipcub_backend_rocprim_util_allocator.hpp`

Nested Relationships

This class is a nested type of *Struct CachingDeviceAllocator*.

Class Documentation

```
class hipcub::CachingDeviceAllocator::TotalBytes
```

Public Functions

```
inline TotalBytes()
```

Public Members

```
size_t free
```

```
size_t live
```

Class `DeviceReduce`

- Defined in file `hipcub_backend_cub_device_device_reduce.hpp`

Class Documentation

class hipcub::DeviceReduce

Public Static Functions

```
template<typename InputIteratorT, typename OutputIteratorT, typename ReduceOpT, typename T>
__host__ static inline hipError_t Reduce(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, ReduceOpT
reduction_op, T init, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t Sum(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t Min(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t ArgMin(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t Max(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t ArgMax(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename KeysInputIteratorT, typename UniqueOutputIteratorT, typename
ValuesInputIteratorT, typename AggregatesOutputIteratorT, typename
NumRunsOutputIteratorT, typename ReductionOpT>
__host__ static inline hipError_t ReduceByKey(void *d_temp_storage, size_t &temp_storage_bytes,
KeysInputIteratorT d_keys_in, UniqueOutputIteratorT
d_unique_out, ValuesInputIteratorT d_values_in,
AggregatesOutputIteratorT d_aggregates_out,
NumRunsOutputIteratorT d_num_runs_out, ReductionOpT
reduction_op, int num_items, hipStream_t stream = 0, bool
debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename ReduceOpT, typename T>
__host__ static inline hipError_t Reduce(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, ReduceOpT
reduction_op, T init, hipStream_t stream = 0, bool
debug_synchronous = false)
```



```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t Sum(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t Min(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t ArgMin(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t Max(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t ArgMax(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT
d_in, OutputIteratorT d_out, int num_items, hipStream_t stream = 0,
bool debug_synchronous = false)
```

```
template<typename KeysInputIteratorT, typename UniqueOutputIteratorT, typename
ValuesInputIteratorT, typename AggregatesOutputIteratorT, typename
NumRunsOutputIteratorT, typename ReductionOpT>
__host__ static inline hipError_t ReduceByKey(void *d_temp_storage, size_t &temp_storage_bytes,
KeysInputIteratorT d_keys_in, UniqueOutputIteratorT
d_unique_out, ValuesInputIteratorT d_values_in,
AggregatesOutputIteratorT d_aggregates_out,
NumRunsOutputIteratorT d_num_runs_out, ReductionOpT
reduction_op, int num_items, hipStream_t stream = 0, bool
debug_synchronous = false)
```

Class DeviceRunLengthEncode

- Defined in file `hipcub_backend_cub_device_device_run_length_encode.hpp`

Class Documentation

```
class hipcub::DeviceRunLengthEncode
```

Public Static Functions

```
template<typename InputIteratorT, typename UniqueOutputIteratorT, typename  
LengthsOutputIteratorT, typename NumRunsOutputIteratorT>  
__host__ static inline hipError_t Encode(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT  
d_in, UniqueOutputIteratorT d_unique_out, LengthsOutputIteratorT  
d_counts_out, NumRunsOutputIteratorT d_num_runs_out, int  
num_items, hipStream_t stream = 0, bool debug_synchronous =  
false)
```

```
template<typename InputIteratorT, typename OffsetsOutputIteratorT, typename  
LengthsOutputIteratorT, typename NumRunsOutputIteratorT>  
__host__ static inline hipError_t NonTrivialRuns(void *d_temp_storage, size_t &temp_storage_bytes,  
InputIteratorT d_in, OffsetsOutputIteratorT  
d_offsets_out, LengthsOutputIteratorT d_lengths_out,  
NumRunsOutputIteratorT d_num_runs_out, int  
num_items, hipStream_t stream = 0, bool  
debug_synchronous = false)
```

```
template<typename InputIteratorT, typename UniqueOutputIteratorT, typename  
LengthsOutputIteratorT, typename NumRunsOutputIteratorT>  
__host__ static inline hipError_t Encode(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT  
d_in, UniqueOutputIteratorT d_unique_out, LengthsOutputIteratorT  
d_counts_out, NumRunsOutputIteratorT d_num_runs_out, int  
num_items, hipStream_t stream = 0, bool debug_synchronous =  
false)
```

```
template<typename InputIteratorT, typename OffsetsOutputIteratorT, typename  
LengthsOutputIteratorT, typename NumRunsOutputIteratorT>  
__host__ static inline hipError_t NonTrivialRuns(void *d_temp_storage, size_t &temp_storage_bytes,  
InputIteratorT d_in, OffsetsOutputIteratorT  
d_offsets_out, LengthsOutputIteratorT d_lengths_out,  
NumRunsOutputIteratorT d_num_runs_out, int  
num_items, hipStream_t stream = 0, bool  
debug_synchronous = false)
```

Class DeviceScan

- Defined in file `hipcub_backend_cub_device_device_scan.hpp`

Class Documentation

```
class hipcub::DeviceScan
```

Public Static Functions

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t InclusiveSum(void *d_temp_storage, size_t &temp_storage_bytes,
        InputIteratorT d_in, OutputIteratorT d_out, int num_items,
        hipStream_t stream = 0, bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename ScanOpT>
__host__ static inline hipError_t InclusiveScan(void *d_temp_storage, size_t &temp_storage_bytes,
        InputIteratorT d_in, OutputIteratorT d_out, ScanOpT
        scan_op, int num_items, hipStream_t stream = 0, bool
        debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t ExclusiveSum(void *d_temp_storage, size_t &temp_storage_bytes,
        InputIteratorT d_in, OutputIteratorT d_out, int num_items,
        hipStream_t stream = 0, bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename ScanOpT, typename
InitValueT>
__host__ static inline hipError_t ExclusiveScan(void *d_temp_storage, size_t &temp_storage_bytes,
        InputIteratorT d_in, OutputIteratorT d_out, ScanOpT
        scan_op, InitValueT init_value, int num_items,
        hipStream_t stream = 0, bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename ScanOpT, typename
InitValueT, typename InitValueIterT = InitValueT*>
__host__ static inline hipError_t ExclusiveScan(void *d_temp_storage, size_t &temp_storage_bytes,
        InputIteratorT d_in, OutputIteratorT d_out, ScanOpT
        scan_op, FutureValue<InitValueT, InitValueIterT>
        init_value, int num_items, hipStream_t stream = 0, bool
        debug_synchronous = false)
```

```
template<typename KeysInputIteratorT, typename ValuesInputIteratorT, typename
ValuesOutputIteratorT, typename EqualityOpT = ::hipcub::Equality>
__host__ static inline hipError_t ExclusiveSumByKey(void *d_temp_storage, size_t &temp_storage_bytes,
        KeysInputIteratorT d_keys_in, ValuesInputIteratorT
        d_values_in, ValuesOutputIteratorT d_values_out, int
        num_items, EqualityOpT equality_op =
        EqualityOpT(), hipStream_t stream = 0, bool
        debug_synchronous = false)
```

```
template<typename KeysInputIteratorT, typename ValuesInputIteratorT, typename
ValuesOutputIteratorT, typename ScanOpT, typename InitValueT, typename EqualityOpT =
::hipcub::Equality>
__host__ static inline hipError_t ExclusiveScanByKey(void *d_temp_storage, size_t &temp_storage_bytes,
        KeysInputIteratorT d_keys_in, ValuesInputIteratorT
        d_values_in, ValuesOutputIteratorT d_values_out,
        ScanOpT scan_op, InitValueT init_value, int
        num_items, EqualityOpT equality_op =
        EqualityOpT(), hipStream_t stream = 0, bool
        debug_synchronous = false)
```

```
template<typename KeysInputIteratorT, typename ValuesInputIteratorT, typename
ValuesOutputIteratorT, typename EqualityOpT = ::hipcub::Equality>
```

```
__host__ static inline hipError_t InclusiveSumByKey(void *d_temp_storage, size_t &temp_storage_bytes,
                                                    KeysInputIteratorT d_keys_in, ValuesInputIteratorT
                                                    d_values_in, ValuesOutputIteratorT d_values_out, int
                                                    num_items, EqualityOpT equality_op =
                                                    EqualityOpT(), hipStream_t stream = 0, bool
                                                    debug_synchronous = false)
```

```
template<typename KeysInputIteratorT, typename ValuesInputIteratorT, typename
ValuesOutputIteratorT, typename ScanOpT, typename EqualityOpT = ::hipcub::Equality>
__host__ static inline hipError_t InclusiveScanByKey(void *d_temp_storage, size_t &temp_storage_bytes,
                                                    KeysInputIteratorT d_keys_in, ValuesInputIteratorT
                                                    d_values_in, ValuesOutputIteratorT d_values_out,
                                                    ScanOpT scan_op, int num_items, EqualityOpT
                                                    equality_op = EqualityOpT(), hipStream_t stream =
                                                    0, bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t InclusiveSum(void *d_temp_storage, size_t &temp_storage_bytes,
                                                InputIteratorT d_in, OutputIteratorT d_out, size_t
                                                num_items, hipStream_t stream = 0, bool
                                                debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename ScanOpT>
__host__ static inline hipError_t InclusiveScan(void *d_temp_storage, size_t &temp_storage_bytes,
                                                InputIteratorT d_in, OutputIteratorT d_out, ScanOpT
                                                scan_op, size_t num_items, hipStream_t stream = 0, bool
                                                debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT>
__host__ static inline hipError_t ExclusiveSum(void *d_temp_storage, size_t &temp_storage_bytes,
                                                InputIteratorT d_in, OutputIteratorT d_out, size_t
                                                num_items, hipStream_t stream = 0, bool
                                                debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename ScanOpT, typename
InitValueT>
__host__ static inline hipError_t ExclusiveScan(void *d_temp_storage, size_t &temp_storage_bytes,
                                                InputIteratorT d_in, OutputIteratorT d_out, ScanOpT
                                                scan_op, InitValueT init_value, size_t num_items,
                                                hipStream_t stream = 0, bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename ScanOpT, typename
InitValueT, typename InitValueIterT = InitValueT*>
__host__ static inline hipError_t ExclusiveScan(void *d_temp_storage, size_t &temp_storage_bytes,
                                                InputIteratorT d_in, OutputIteratorT d_out, ScanOpT
                                                scan_op, FutureValue<InitValueT, InitValueIterT>
                                                init_value, int num_items, hipStream_t stream = 0, bool
                                                debug_synchronous = false)
```

```
template<typename KeysInputIteratorT, typename ValuesInputIteratorT, typename
ValuesOutputIteratorT, typename EqualityOpT = ::hipcub::Equality>
```

```

__host__ static inline hipError_t ExclusiveSumByKey(void *d_temp_storage, size_t &temp_storage_bytes,
    KeysInputIteratorT d_keys_in, ValuesInputIteratorT
    d_values_in, ValuesOutputIteratorT d_values_out, int
    num_items, EqualityOpT equality_op =
    EqualityOpT(), hipStream_t stream = 0, bool
    debug_synchronous = false)

```

```

template<typename KeysInputIteratorT, typename ValuesInputIteratorT, typename
ValuesOutputIteratorT, typename ScanOpT, typename InitValueT, typename EqualityOpT =
::hipcub::Equality>

```

```

__host__ static inline hipError_t ExclusiveScanByKey(void *d_temp_storage, size_t &temp_storage_bytes,
    KeysInputIteratorT d_keys_in, ValuesInputIteratorT
    d_values_in, ValuesOutputIteratorT d_values_out,
    ScanOpT scan_op, InitValueT init_value, int
    num_items, EqualityOpT equality_op =
    EqualityOpT(), hipStream_t stream = 0, bool
    debug_synchronous = false)

```

```

template<typename KeysInputIteratorT, typename ValuesInputIteratorT, typename
ValuesOutputIteratorT, typename EqualityOpT = ::hipcub::Equality>

```

```

__host__ static inline hipError_t InclusiveSumByKey(void *d_temp_storage, size_t &temp_storage_bytes,
    KeysInputIteratorT d_keys_in, ValuesInputIteratorT
    d_values_in, ValuesOutputIteratorT d_values_out, int
    num_items, EqualityOpT equality_op =
    EqualityOpT(), hipStream_t stream = 0, bool
    debug_synchronous = false)

```

```

template<typename KeysInputIteratorT, typename ValuesInputIteratorT, typename
ValuesOutputIteratorT, typename ScanOpT, typename EqualityOpT = ::hipcub::Equality>

```

```

__host__ static inline hipError_t InclusiveScanByKey(void *d_temp_storage, size_t &temp_storage_bytes,
    KeysInputIteratorT d_keys_in, ValuesInputIteratorT
    d_values_in, ValuesOutputIteratorT d_values_out,
    ScanOpT scan_op, int num_items, EqualityOpT
    equality_op = EqualityOpT(), hipStream_t stream =
    0, bool debug_synchronous = false)

```

Class DeviceSelect

- Defined in file `hipcub_backend_cub_device_device_select.hpp`

Class Documentation

```
class hipcub::DeviceSelect
```

Public Static Functions

```
template<typename InputIteratorT, typename FlagIterator, typename OutputIteratorT, typename  
NumSelectedIteratorT>  
__host__ static inline hipError_t Flagged(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT  
d_in, FlagIterator d_flags, OutputIteratorT d_out,  
NumSelectedIteratorT d_num_selected_out, int num_items,  
hipStream_t stream = 0, bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename NumSelectedIteratorT,  
typename SelectOp>  
__host__ static inline hipError_t If(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT d_in,  
OutputIteratorT d_out, NumSelectedIteratorT d_num_selected_out, int  
num_items, SelectOp select_op, hipStream_t stream = 0, bool  
debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename NumSelectedIteratorT>  
__host__ static inline hipError_t Unique(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT  
d_in, OutputIteratorT d_out, NumSelectedIteratorT  
d_num_selected_out, int num_items, hipStream_t stream = 0, bool  
debug_synchronous = false)
```

```
template<typename InputIteratorT, typename FlagIterator, typename OutputIteratorT, typename  
NumSelectedIteratorT>  
__host__ static inline hipError_t Flagged(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT  
d_in, FlagIterator d_flags, OutputIteratorT d_out,  
NumSelectedIteratorT d_num_selected_out, int num_items,  
hipStream_t stream = 0, bool debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename NumSelectedIteratorT,  
typename SelectOp>  
__host__ static inline hipError_t If(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT d_in,  
OutputIteratorT d_out, NumSelectedIteratorT d_num_selected_out, int  
num_items, SelectOp select_op, hipStream_t stream = 0, bool  
debug_synchronous = false)
```

```
template<typename InputIteratorT, typename OutputIteratorT, typename NumSelectedIteratorT>  
__host__ static inline hipError_t Unique(void *d_temp_storage, size_t &temp_storage_bytes, InputIteratorT  
d_in, OutputIteratorT d_out, NumSelectedIteratorT  
d_num_selected_out, int num_items, hipStream_t stream = 0, bool  
debug_synchronous = false)
```

Class DeviceSpmv

- Defined in file `hipcub_backend_cub_device_device_spmv.hpp`

Nested Relationships

Nested Types

- *Template Struct DeviceSpmv::SpmvParams*

Class Documentation

class hipcub::DeviceSpmv

Public Static Functions

```
template<typename ValueT>
__host__ static inline hipError_t CsrMV(void *d_temp_storage, size_t &temp_storage_bytes, ValueT
*d_values, int *d_row_offsets, int *d_column_indices, ValueT
*d_vector_x, ValueT *d_vector_y, int num_rows, int num_cols, int
num_nonzeros, hipStream_t stream = 0, bool debug_synchronous =
false)
```

Parameters

- **d_temp_storage** – Device-accessible allocation of temporary storage. When NULL, the required allocation size is written to **temp_storage_bytes** and no work is done.
- **temp_storage_bytes** – Reference to size in bytes of **d_temp_storage** allocation
- **d_values** – Pointer to the array of **num_nonzeros** values of the corresponding nonzero elements of matrix **A**.
- **d_row_offsets** – Pointer to the array of $m + 1$ offsets demarcating the start of every row in **d_column_indices** and **d_values** (with the final entry being equal to **num_nonzeros**)
- **d_column_indices** – Pointer to the array of **num_nonzeros** column-indices of the corresponding nonzero elements of matrix **A**. (Indices are zero-valued.)
- **d_vector_x** – Pointer to the array of **num_cols** values corresponding to the dense input vector x
- **d_vector_y** – Pointer to the array of **num_rows** values corresponding to the dense output vector y
- **num_rows** – number of rows of matrix **A**.
- **num_cols** – number of columns of matrix **A**.
- **num_nonzeros** – number of nonzero elements of matrix **A**.
- **stream** – [optional] hip stream to launch kernels within. Default is `stream0`.
- **debug_synchronous** – [optional] Whether or not to synchronize the stream after every kernel launch to check for errors. May cause significant slowdown. Default is `false`.

```
template<typename ValueT>
__global__ static inline void CsrMVKernel(SpmvParams<ValueT>, int> spmv_params)
```

```
template<typename ValueT>
```

```
__host__ static inline hipError_t CsrMV(void *d_temp_storage, size_t &temp_storage_bytes, ValueT
    *d_values, int *d_row_offsets, int *d_column_indices, ValueT
    *d_vector_x, ValueT *d_vector_y, int num_rows, int num_cols, int
    num_nonzeros, hipStream_t stream = 0, bool debug_synchronous =
    false)
```

Parameters

- **d_temp_storage** – Device-accessible allocation of temporary storage. When NULL, the required allocation size is written to `temp_storage_bytes` and no work is done.
- **temp_storage_bytes** – Reference to size in bytes of `d_temp_storage` allocation
- **d_values** – Pointer to the array of `num_nonzeros` values of the corresponding nonzero elements of matrix **A**.
- **d_row_offsets** – Pointer to the array of `m + 1` offsets demarcating the start of every row in `d_column_indices` and `d_values` (with the final entry being equal to `num_nonzeros`)
- **d_column_indices** – Pointer to the array of `num_nonzeros` column-indices of the corresponding nonzero elements of matrix **A**. (Indices are zero-valued.)
- **d_vector_x** – Pointer to the array of `num_cols` values corresponding to the dense input vector *x*
- **d_vector_y** – Pointer to the array of `num_rows` values corresponding to the dense output vector *y*
- **num_rows** – number of rows of matrix **A**.
- **num_cols** – number of columns of matrix **A**.
- **num_nonzeros** – number of nonzero elements of matrix **A**.
- **stream** – [optional] hip stream to launch kernels within. Default is `stream0`.
- **debug_synchronous** – [optional] Whether or not to synchronize the stream after every kernel launch to check for errors. May cause significant slowdown. Default is `false`.

Public Static Attributes

```
static constexpr uint32_t CsrMVKernel_MaxThreads = 256
```

```
template<typename ValueT, typename OffsetT>
```

```
struct SpmvParams
```

```
< Signed integer type for sequence offsets
```

Public Members

```
ValueT *d_values
```

Pointer to the array of `num_nonzeros` values of the corresponding nonzero elements of matrix **A**.

```
OffsetT *d_row_end_offsets
```

Pointer to the array of `m` offsets demarcating the end of every row in `d_column_indices` and `d_values`.

OffsetT ***d_column_indices**

Pointer to the array of `num_nonzeros` column-indices of the corresponding nonzero elements of matrix **A**. (Indices are zero-valued.)

ValueT ***d_vector_x**

Pointer to the array of `num_cols` values corresponding to the dense input vector *x*

ValueT ***d_vector_y**

Pointer to the array of `num_rows` values corresponding to the dense output vector *y*

int **num_rows**

Number of rows of matrix **A**.

int **num_cols**

Number of columns of matrix **A**.

int **num_nonzeros**

Number of nonzero elements of matrix **A**.

ValueT **alpha**

Alpha multiplicand.

ValueT **beta**

Beta addend-multiplicand.

`::cub::TexRefInputIterator<ValueT, 66778899, OffsetT>` **t_vector_x**

`::hipcub::TexRefInputIterator<ValueT, 66778899, OffsetT>` **t_vector_x**

Template Class DiscardOutputIterator

- Defined in file `_hipcub_backend_rocprim_iterator_discard_output_iterator.hpp`

Class Documentation

```
template<typename OffsetT = ptrdiff_t>
```

```
class hipcub::DiscardOutputIterator
```

A discard iterator.

Public Types

typedef *DiscardOutputIterator* **self_type**

My own type.
ostream operator
Not equal to.
Equal to.
Array subscript.
Distance.
Subtraction assignment.
Addition assignment.
Addition.
Indirection.
Postfix increment.

typedef *OffsetT* **difference_type**

Type to express the result of subtracting one iterator from another.

typedef void **value_type**

The type of the element the iterator can point to.

typedef void **pointer**

The type of a pointer to an element the iterator can point to.

typedef void **reference**

The type of a reference to an element the iterator can point to.

typedef std::random_access_iterator_tag **iterator_category**

The iterator category.

Public Functions

`__host__ __device__ __forceinline__ inline DiscardOutputIterator(OffsetT offset = 0)`

Constructor.

Parameters *offset* – Base offset

`__host__ __device__ __forceinline__ inline self_type operator++(int)`

`__host__ __device__ __forceinline__ inline self_type operator++()`

`__host__ __device__ __forceinline__ inline self_type &operator*()`

`template<typename Distance>`

```

__host__ __device__ __forceinline__ inline self_type operator+(Distance n) const

template<typename Distance>
__host__ __device__ __forceinline__ inline self_type &operator+=(Distance n)

template<typename Distance>
__host__ __device__ __forceinline__ inline self_type operator-(Distance n) const

template<typename Distance>
__host__ __device__ __forceinline__ inline self_type &operator-=(Distance n)

__host__ __device__ __forceinline__ inline difference_type operator-(self_type other) const

template<typename Distance>
__host__ __device__ __forceinline__ inline self_type &operator[](Distance)

__host__ __device__ __forceinline__ inline pointer operator->()
    Structure dereference.

template<typename T>
__host__ __device__ __forceinline__ inline void operator=(T const&)
    Assignment to anything else (no-op)

__host__ __device__ __forceinline__ inline operator void*() const
    Cast to void* operator.

__host__ __device__ __forceinline__ inline bool operator==(const self_type &rhs)

__host__ __device__ __forceinline__ inline bool operator!=(const self_type &rhs)

```

Friends

```
inline friend std::ostream &operator<<(std::ostream &os, const self_type &itr)
```

Class GridBarrier

- Defined in file `hipcub_backend_rocprim_grid_grid_barrier.hpp`

Inheritance Relationships

Derived Type

- public `hipcub::GridBarrierLifetime` (Class *GridBarrierLifetime*)

Class Documentation

class hipcub::GridBarrier

GridBarrier implements a software global barrier among thread blocks within a hip grid.

Subclassed by *hipcub::GridBarrierLifetime*

Public Functions

inline GridBarrier()

Constructor

__device__ __forceinline__ inline void Sync() const

Protected Types

typedef unsigned int SyncFlag

Synchronize.

Protected Attributes

SyncFlag *d_sync

Class GridBarrierLifetime

- Defined in file `hipcub_backend_cub_grid_grid_barrier.hpp`

Inheritance Relationships

Base Types

- public hipcub::GridBarrier (*Class GridBarrier*)
- public GridBarrierLifetime

Class Documentation

class hipcub::GridBarrierLifetime : public hipcub::GridBarrier, public GridBarrierLifetime

GridBarrierLifetime extends *GridBarrier* to provide lifetime management of the temporary device storage needed for cooperation.

Uses RAII for lifetime, i.e., device resources are reclaimed when the destructor is called.

Public Functions

```
inline hipError_t HostReset()
```

```
inline hipError_t Setup(int sweep_grid_size)
```

```
inline GridBarrierLifetime()
```

Constructor

```
inline hipError_t HostReset()
```

DeviceFrees and resets the progress counters

```
inline virtual ~GridBarrierLifetime()
```

Destructor

```
inline hipError_t Setup(int sweep_grid_size)
```

Sets up the progress counters for the next kernel launch (lazily allocating and initializing them if necessary)

```
__device__ __forceinline__ inline void Sync() const
```

Protected Types

```
typedef unsigned int SyncFlag
```

Synchronize.

Protected Attributes

```
size_t sync_bytes
```

```
SyncFlag *d_sync
```

Template Class GridQueue

- Defined in file `hipcub_backend_rocprim_grid_grid_queue.hpp`

Class Documentation

```
template<typename OffsetT>
```

```
class hipcub::GridQueue
```

GridQueue is a descriptor utility for dynamic queue management.

Overview *GridQueue* descriptors provides abstractions for “filling” or “draining” globally-shared vectors.

A “filling” *GridQueue* works by atomically-adding to a zero-initialized counter, returning a unique offset for the calling thread to write its items. The *GridQueue* maintains the total “fill-size”. The fill counter must be reset using *GridQueue::ResetFill* by the host or kernel instance prior to the kernel instance that will be filling.

Similarly, a “draining” *GridQueue* works by works by atomically-incrementing a zero-initialized counter, returning a unique offset for the calling thread to read its items. Threads can safely drain until the array’s logical fill-size is exceeded. The drain counter must be reset using *GridQueue::ResetDrain* or *GridQueue::FillAndResetDrain* by the host or kernel instance prior to the kernel instance that will be filling. (For dynamic work distribution of existing data, the corresponding fill-size is simply the number of elements in the array.)

Iterative work management can be implemented simply with a pair of flip-flopping work buffers, each with an associated set of fill and drain *GridQueue* descriptors.

Template Parameters *OffsetT* – Signed integer type for global offsets

Public Functions

`__host__ __device__ __forceinline__ inline GridQueue()`

Constructs an invalid *GridQueue* descriptor.

`__host__ __device__ __forceinline__ inline GridQueue(void *d_storage)`

Constructs a *GridQueue* descriptor around the device storage allocation.

Parameters *d_storage* – Device allocation to back the *GridQueue*. Must be at least as big as *AllocationSize()*.

`__device__ inline hipError_t FillAndResetDrain(OffsetT fill_size, hipStream_t stream = 0)`

This operation sets the fill-size and resets the drain counter, preparing the *GridQueue* for draining in the next kernel instance. To be called by the host or by a kernel prior to that which will be draining.

`__host__ inline hipError_t FillAndResetDrain(OffsetT fill_size, hipStream_t stream = 0)`

`__device__ inline hipError_t ResetDrain(hipStream_t stream = 0)`

This operation resets the drain so that it may advance to meet the existing fill-size. To be called by the host or by a kernel prior to that which will be draining.

`__host__ inline hipError_t ResetDrain(hipStream_t stream = 0)`

`__device__ inline hipError_t ResetFill(hipStream_t stream = 0)`

This operation resets the fill counter. To be called by the host or by a kernel prior to that which will be filling.

`__host__ inline hipError_t ResetFill(hipStream_t stream = 0)`

`__device__ inline hipError_t FillSize(OffsetT &fill_size, hipStream_t stream = 0)`

Returns the fill-size established by the parent or by the previous kernel.

`__host__ inline hipError_t FillSize(OffsetT &fill_size, hipStream_t stream = 0)`

`__device__ inline OffsetT Drain(OffsetT num_items)`

Drain *num_items* from the queue. Returns offset from which to read items. To be called from hip kernel.

`__device__ inline OffsetT Fill(OffsetT num_items)`

Fill *num_items* into the queue. Returns offset from which to write items. To be called from hip kernel.

Public Static Functions

`__host__ __device__ __forceinline__ static inline size_t AllocationSize()`

Returns the device allocation size in bytes needed to construct a *GridQueue* instance.

Template Class TexObjInputIterator

- Defined in file `hipcub_backend_rocprim_iterator_tex_obj_input_iterator.hpp`

Inheritance Relationships

Base Type

- `public rocprim::texture_cache_iterator< T, OffsetT >`

Class Documentation

`template<typename T, typename OffsetT = std::ptrdiff_t>`

`class hipcub::TexObjInputIterator : public rocprim::texture_cache_iterator<T, OffsetT>`

Public Functions

`template<class Qualified>`

`inline hipError_t BindTexture(Qualified *ptr, size_t bytes = size_t(-1), size_t texture_offset = 0)`

`inline hipError_t UnbindTexture()`

`__host__ __device__ inline ~TexObjInputIterator() = default`

`__host__ __device__ inline TexObjInputIterator()`

`__host__ __device__ inline TexObjInputIterator(const ::rocprim::texture_cache_iterator<T, OffsetT> other)`

Template Class TexRefInputIterator

- Defined in file `hipcub_backend_rocprim_iterator_tex_ref_input_iterator.hpp`

Inheritance Relationships

Base Type

- `public rocprim::texture_cache_iterator< T, OffsetT >`

Class Documentation

```
template<typename T, int UNIQUE_ID, typename OffsetT = std::ptrdiff_t>
class hipcub::TexRefInputIterator : public rocpim::texture_cache_iterator<T, OffsetT>
```

Public Functions

```
template<class Qualified>
inline hipError_t BindTexture(Qualified *ptr, size_t bytes = size_t(-1), size_t texture_offset = 0)

inline hipError_t UnbindTexture()

__host__ __device__ inline ~TexRefInputIterator() = default

__host__ __device__ inline TexRefInputIterator()

__host__ __device__ inline TexRefInputIterator(const ::rocpim::texture_cache_iterator<T, OffsetT>
                                                other)
```

Template Class WarpExchange

- Defined in file `hipcub_backend_rocpim_warp_warp_exchange.hpp`

Nested Relationships

Nested Types

- *Struct WarpExchange::TempStorage*
- *Union WarpExchange::_TempStorage*

Class Documentation

```
template<typename InputT, int ITEMS_PER_THREAD, int LOGICAL_WARP_THREADS =
HIPCUB_DEVICE_WARP_THREADS, int ARCH = HIPCUB_ARCH>
class hipcub::WarpExchange
```

Public Functions

```
WarpExchange() = delete

__device__ __forceinline__ inline explicit WarpExchange(TempStorage &temp_storage)

template<typename OutputT>
__device__ __forceinline__ inline void BlockedToStriped(const InputT
                                                         (&input_items)[ITEMS_PER_THREAD],
                                                         OutputT
                                                         (&output_items)[ITEMS_PER_THREAD])

template<typename OutputT>
```



```

__device__ __forceinline__ inline void StripedToBlocked(const InputT
                                                         (&input_items)[ITEMS_PER_THREAD],
                                                         OutputT
                                                         (&output_items)[ITEMS_PER_THREAD])

template<typename OffsetT>
__device__ __forceinline__ inline void ScatterToStriped(InputT (&items)[ITEMS_PER_THREAD],
                                                         OffsetT (&ranks)[ITEMS_PER_THREAD])

template<typename OutputT, typename OffsetT>
__device__ __forceinline__ inline void ScatterToStriped(const InputT
                                                         (&input_items)[ITEMS_PER_THREAD],
                                                         OutputT
                                                         (&output_items)[ITEMS_PER_THREAD],
                                                         OffsetT (&ranks)[ITEMS_PER_THREAD])

struct TempStorage : public hipcub::Uninitialized<_TempStorage>

```

Public Types

```
enum [anonymous]
```

Values:

```
typedef UnitWord<_TempStorage>::DeviceWord DeviceWord
```

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

```
__host__ __device__ __forceinline__ inline _TempStorage &Alias()
```

Alias.

Public Members

```
DeviceWord storage[WORDS]
```

Backing storage.

Template Class WarpLoad

- Defined in file `hipcub_backend_rocprim_warp_warp_load.hpp`

Nested Relationships

Nested Types

- *Template Struct WarpLoad::LoadInternal*
- *Template Struct WarpLoad::LoadInternal< WARP_LOAD_DIRECT >*
- *Template Struct WarpLoad::LoadInternal< WARP_LOAD_STRIPED >*
- *Template Struct WarpLoad::LoadInternal< WARP_LOAD_TRANSPOSE >*
- *Template Struct WarpLoad::LoadInternal< WARP_LOAD_VECTORIZE >*
- *Struct WarpLoad::TempStorage*

Class Documentation

```
template<class InputT, int ITEMS_PER_THREAD, WarpLoadAlgorithm ALGORITHM = WARP_LOAD_DIRECT, int LOGICAL_WARP_THREADS = HIPCUB_DEVICE_WARP_THREADS, int ARCH = HIPCUB_ARCH>
class hipcub::WarpLoad
```

Public Functions

```
__device__ __forceinline__ inline WarpLoad()

__device__ __forceinline__ inline WarpLoad(TempStorage &temp_storage)

template<typename InputIteratorT>
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT
                                             (&items)[ITEMS_PER_THREAD])

template<typename InputIteratorT>
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT
                                             (&items)[ITEMS_PER_THREAD], int valid_items)

template<typename InputIteratorT, typename DefaultT>
__device__ __forceinline__ inline void Load(InputIteratorT block_itr, InputT
                                             (&items)[ITEMS_PER_THREAD], int valid_items, DefaultT
                                             oob_default)

struct TempStorage : public hipcub::Uninitialized<_TempStorage>
```

Public Types

```
enum [anonymous]
```

Values:

```
typedef UnitWord<_TempStorage>::DeviceWord DeviceWord
```

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

`__host__ __device__ __forceinline__ inline _TempStorage &Alias()`
Alias.

Public Members

DeviceWord **storage**[WORDS]
Backing storage.

Template Class WarpMergeSort

- Defined in file `hipcub_backend_rocprim_warp_warp_merge_sort.hpp`

Inheritance Relationships

Base Type

- `public hipcub::BlockMergeSortStrategy< KeyT, ValueT, LOGICAL_WARP_THREADS, ITEMS_PER_THREAD, WarpMergeSort< KeyT, ITEMS_PER_THREAD, LOGICAL_WARP_THREADS, ValueT, PTX_ARCH > >` (*Template Class BlockMergeSortStrategy*)

Class Documentation

```
template<typename KeyT, int ITEMS_PER_THREAD, int LOGICAL_WARP_THREADS = ::rocprim::device_warp_size(),
typename ValueT = NullType, int PTX_ARCH = 1>
class hipcub::WarpMergeSort : public hipcub::BlockMergeSortStrategy<KeyT, ValueT,
LOGICAL_WARP_THREADS, ITEMS_PER_THREAD, WarpMergeSort<KeyT, ITEMS_PER_THREAD,
LOGICAL_WARP_THREADS, ValueT, PTX_ARCH>>
```

The *WarpMergeSort* class provides methods for sorting items partitioned across a CUDA warp using a merge sorting method.

Overview *WarpMergeSort* arranges items into ascending order using a comparison functor with less-than semantics. Merge sort can handle arbitrary types and comparison functors.

A Simple Example

The code snippet below illustrates a sort of 64 integer keys that are partitioned across 16 threads where each thread owns 4 consecutive items.

```
#include <cub/cub.cuh> // or equivalently <cub/warp/warp_merge_sort.cuh>

struct CustomLess
{
    template <typename DataType>
    __device__ bool operator()(const DataType &lhs, const DataType &rhs)
    {
```

(continues on next page)

(continued from previous page)

```

    return lhs < rhs;
}
};

__global__ void ExampleKernel(...)
{
    constexpr int warp_threads = 16;
    constexpr int block_threads = 256;
    constexpr int items_per_thread = 4;
    constexpr int warps_per_block = block_threads / warp_threads;
    const int warp_id = static_cast<int>(threadIdx.x) / warp_threads;

    // Specialize WarpMergeSort for a virtual warp of 16 threads
    // owning 4 integer items each
    using WarpMergeSortT =
        cub::WarpMergeSort<int, items_per_thread, warp_threads>;

    // Allocate shared memory for WarpMergeSort
    __shared__ typename WarpMergeSortT::TempStorage temp_storage[warp_per_
    ↪block];

    // Obtain a segment of consecutive items that are blocked across threads
    int thread_keys[items_per_thread];
    // ...

    WarpMergeSortT(temp_storage[warp_id]).Sort(thread_keys, CustomLess());
    // ...
}

```

Suppose the set of input `thread_keys` across the block of threads is { [0, 511, 1, 510], [2, 509, 3, 508], [4, 507, 5, 506], ..., [254, 257, 255, 256] }. The corresponding output `thread_keys` in those threads will be { [0, 1, 2, 3], [4, 5, 6, 7], [8, 9, 10, 11], ..., [508, 509, 510, 511] }.

Template Parameters

- **KeyT** – Key type
- **ITEMS_PER_THREAD** – The number of items per thread
- **LOGICAL_WARP_THREADS** – [optional] The number of threads per “logical” warp (may be less than the number of hardware warp threads). Default is the warp size of the targeted CUDA compute-capability (e.g., 32 threads for SM86). Must be a power of two.
- **ValueT** – [optional] Value type (default: `cub::NullType`, which indicates a keys-only sort)
- **PTX_ARCH** – [optional]

Public Functions

WarpMergeSort() = delete

```
__device__ __forceinline__ inline WarpMergeSort(typename BlockMergeSortStrategyT::TempStorage
                                                &temp_storage)
```

```
__device__ __forceinline__ inline uint64_t get_member_mask() const
```

```
__device__ __forceinline__ inline unsigned int get_linear_tid() const
```

```
__device__ __forceinline__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp
                                             compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

Sort is not guaranteed to be stable. That is, suppose that *i* and *j* are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.

Template Parameters CompareOp – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
__device__ __forceinline__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp
                                             compare_op, int valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- Sort is not guaranteed to be stable. That is, suppose that *i* and *j* are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.
- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. If there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters CompareOp – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

```
__device__ __forceinline__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], ValueT
(&items)[ITEMS_PER_THREAD], CompareOp compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

Sort is not guaranteed to be stable. That is, suppose that *i* and *j* are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.

Template Parameters **CompareOp** – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
__device__ __forceinline__ inline void Sort(KeyT (&keys)[ITEMS_PER_THREAD], ValueT
(&items)[ITEMS_PER_THREAD], CompareOp compare_op, int
valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- Sort is not guaranteed to be stable. That is, suppose that *i* and *j* are equivalent: neither one is less than the other. It is not guaranteed that the relative order of these two elements will be preserved by sort.
- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. If there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters

- **CompareOp** – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).
- **IS_LAST_TILE** – True if `valid_items` isn't equal to the `ITEMS_PER_TILE`

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

```
__device__ __forceinline__ inline void StableSort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp
compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

StableSort is stable: it preserves the relative ordering of equivalent elements. That is, if x and y are elements such that x precedes y , and if the two elements are equivalent (neither $x < y$ nor $y < x$) then a postcondition of **StableSort** is that x still precedes y .

Template Parameters **CompareOp** – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. **CompareOp** is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
__device__ __forceinline__ inline void StableSort(KeyT (&keys)[ITEMS_PER_THREAD], ValueT
(&items)[ITEMS_PER_THREAD], CompareOp
compare_op)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

StableSort is stable: it preserves the relative ordering of equivalent elements. That is, if x and y are elements such that x precedes y , and if the two elements are equivalent (neither $x < y$ nor $y < x$) then a postcondition of **StableSort** is that x still precedes y .

Template Parameters **CompareOp** – functor type having member `bool operator()(KeyT lhs, KeyT rhs)`. **CompareOp** is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second

```
__device__ __forceinline__ inline void StableSort(KeyT (&keys)[ITEMS_PER_THREAD], CompareOp
compare_op, int valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- **StableSort** is stable: it preserves the relative ordering of equivalent elements. That is, if x and y are elements such that x precedes y , and if the two elements are equivalent (neither $x < y$ nor $y < x$) then a postcondition of **StableSort** is that x still precedes y .
- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. If there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters **CompareOp** – functor type having member `bool operator() (KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).

Parameters

- **keys** – [inout] Keys to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

```
__device__ __forceinline__ inline void StableSort (KeyT (&keys)[ITEMS_PER_THREAD], ValueT (&items)[ITEMS_PER_THREAD], CompareOp compare_op, int valid_items, KeyT oob_default)
```

Sorts items partitioned across a CUDA thread block using a merge sorting method.

- `StableSort` is stable: it preserves the relative ordering of equivalent elements. That is, if `x` and `y` are elements such that `x` precedes `y`, and if the two elements are equivalent (neither `x < y` nor `y < x`) then a postcondition of `StableSort` is that `x` still precedes `y`.
- The value of `oob_default` is assigned to all elements that are out of `valid_items` boundaries. It's expected that `oob_default` is ordered after any value in the `valid_items` boundaries. The algorithm always sorts a fixed amount of elements, which is equal to `ITEMS_PER_THREAD * BLOCK_THREADS`. If there is a value that is ordered after `oob_default`, it won't be placed within `valid_items` boundaries.

Template Parameters

- **CompareOp** – functor type having member `bool operator() (KeyT lhs, KeyT rhs)`. `CompareOp` is a model of [Strict Weak Ordering](#).
- **IS_LAST_TILE** – True if `valid_items` isn't equal to the `ITEMS_PER_TILE`

Parameters

- **keys** – [inout] Keys to sort
- **items** – [inout] Values to sort
- **compare_op** – [in] Comparison function object which returns true if the first argument is ordered before the second
- **valid_items** – [in] Number of valid items to sort
- **oob_default** – [in] Default value to assign out-of-bound items

Template Class WarpReduce

- Defined in file `hipcub_backend_rocprim_warp_warp_reduce.hpp`

Inheritance Relationships

Base Type

- `private rocprim::warp_reduce< T, LOGICAL_WARP_THREADS >`

Class Documentation

```
template<typename T, int LOGICAL_WARP_THREADS = ::rocprim::device_warp_size(), int ARCH = 1>
```

```
class hipcub::WarpReduce : private rocprim::warp_reduce<T, LOGICAL_WARP_THREADS>
```

Public Types

```
using TempStorage = typename base_type::storage_type
```

Public Functions

```
__device__ inline WarpReduce(TempStorage &temp_storage)
```

```
__device__ inline T Sum(T input)
```

```
__device__ inline T Sum(T input, int valid_items)
```

```
template<typename FlagT>
```

```
__device__ inline T HeadSegmentedSum(T input, FlagT head_flag)
```

```
template<typename FlagT>
```

```
__device__ inline T TailSegmentedSum(T input, FlagT tail_flag)
```

```
template<typename ReduceOp>
```

```
__device__ inline T Reduce(T input, ReduceOp reduce_op)
```

```
template<typename ReduceOp>
```

```
__device__ inline T Reduce(T input, ReduceOp reduce_op, int valid_items)
```

```
template<typename ReduceOp, typename FlagT>
```

```
__device__ inline T HeadSegmentedReduce(T input, FlagT head_flag, ReduceOp reduce_op)
```

```
template<typename ReduceOp, typename FlagT>
```

```
__device__ inline T TailSegmentedReduce(T input, FlagT tail_flag, ReduceOp reduce_op)
```

Template Class WarpScan

- Defined in file `hipcub_backend_rocprim_warp_warp_scan.hpp`

Inheritance Relationships

Base Type

- `private rocprim::warp_scan< T, LOGICAL_WARP_THREADS >`

Class Documentation

```
template<typename T, int LOGICAL_WARP_THREADS = ::rocprim::device_warp_size(), int ARCH = 1>
```

```
class hipcub::WarpScan : private rocprim::warp_scan<T, LOGICAL_WARP_THREADS>
```

Public Types

```
using TempStorage = typename base_type::storage_type
```

Public Functions

```
__device__ inline WarpScan(TempStorage &temp_storage)
```

```
__device__ inline void InclusiveSum(T input, T &inclusive_output)
```

```
__device__ inline void InclusiveSum(T input, T &inclusive_output, T &warp_aggregate)
```

```
__device__ inline void ExclusiveSum(T input, T &exclusive_output)
```

```
__device__ inline void ExclusiveSum(T input, T &exclusive_output, T &warp_aggregate)
```

```
template<typename ScanOp>
```

```
__device__ inline void InclusiveScan(T input, T &inclusive_output, ScanOp scan_op)
```

```
template<typename ScanOp>
```

```
__device__ inline void InclusiveScan(T input, T &inclusive_output, ScanOp scan_op, T  
                                     &warp_aggregate)
```

```
template<typename ScanOp>
```

```
__device__ inline void ExclusiveScan(T input, T &exclusive_output, ScanOp scan_op)
```

```
template<typename ScanOp>
```

```
__device__ inline void ExclusiveScan(T input, T &exclusive_output, T initial_value, ScanOp scan_op)
```

```
template<typename ScanOp>
```

```
__device__ inline void ExclusiveScan(T input, T &exclusive_output, ScanOp scan_op, T  
                                     &warp_aggregate)
```

```
template<typename ScanOp>
```

```

__device__ inline void ExclusiveScan(T input, T &exclusive_output, T initial_value, ScanOp scan_op, T
&warp_aggregate)

template<typename ScanOp>
__device__ inline void Scan(T input, T &inclusive_output, T &exclusive_output, ScanOp scan_op)

template<typename ScanOp>
__device__ inline void Scan(T input, T &inclusive_output, T &exclusive_output, T initial_value, ScanOp
scan_op)

__device__ inline T Broadcast(T input, unsigned int src_lane)

```

Template Class WarpStore

- Defined in file_hipcub_backend_rocprim_warp_warp_store.hpp

Nested Relationships

Nested Types

- *Template Struct WarpStore::StoreInternal*
- *Template Struct WarpStore::StoreInternal< WARP_STORE_DIRECT >*
- *Template Struct WarpStore::StoreInternal< WARP_STORE_STRIPED >*
- *Template Struct WarpStore::StoreInternal< WARP_STORE_TRANSPOSE >*
- *Template Struct WarpStore::StoreInternal< WARP_STORE_VECTORIZE >*
- *Struct WarpStore::TempStorage*

Class Documentation

```

template<class T, int ITEMS_PER_THREAD, WarpStoreAlgorithm ALGORITHM = WARP_STORE_DIRECT, int
LOGICAL_WARP_THREADS = HIPCUB_DEVICE_WARP_THREADS, int ARCH = HIPCUB_ARCH>
class hipcub: WarpStore

```

Public Functions

```

__device__ __forceinline__ inline WarpStore()

__device__ __forceinline__ inline WarpStore(TempStorage &temp_storage)

template<typename OutputIteratorT>
__device__ __forceinline__ inline void Store(OutputIteratorT block_itr, T
(&items)[ITEMS_PER_THREAD])

template<typename OutputIteratorT>
__device__ __forceinline__ inline void Store(OutputIteratorT block_itr, T
(&items)[ITEMS_PER_THREAD], int valid_items)

template<typename OutputIteratorT, typename DefaultT>

```

```
__device__ __forceinline__ inline void Store(OutputIteratorT block_itr, T  
(&items)[ITEMS_PER_THREAD], int valid_items, DefaultT  
oob_default)
```

```
struct TempStorage : public hipcub::Uninitialized<_TempStorage>
```

Public Types

```
enum [anonymous]
```

Values:

```
typedef UnitWord<_TempStorage>::DeviceWord DeviceWord
```

Biggest memory-access word that T is a whole multiple of and is not larger than the alignment of T.

Public Functions

```
__host__ __device__ __forceinline__ inline _TempStorage &Alias()
```

Alias.

Public Members

```
DeviceWord storage[WORDS]
```

Backing storage.

2.1.3 Enums

Enum BlockHistogramAlgorithm

- Defined in file `hipcub_backend_rocprim_block_block_histogram.hpp`

Enum Documentation

```
enum BlockHistogramAlgorithm
```

Values:

```
enumerator BLOCK_HISTO_ATOMIC
```

```
enumerator BLOCK_HISTO_SORT
```

Enum BlockReduceAlgorithm

- Defined in file_hipcub_backend_rocprim_block_block_reduce.hpp

Enum Documentation

enum **BlockReduceAlgorithm**

Values:

enumerator **BLOCK_REDUCE_RAKING_COMMUTATIVE_ONLY**

enumerator **BLOCK_REDUCE_RAKING**

enumerator **BLOCK_REDUCE_WARP_REDUCTIONS**

Enum CacheLoadModifier

- Defined in file_hipcub_backend_rocprim_thread_thread_load.hpp

Enum Documentation

enum **CacheLoadModifier**

Values:

enumerator **LOAD_DEFAULT**

Default (no modifier)

enumerator **LOAD_CA**

Cache at all levels.

enumerator **LOAD_CG**

Cache at global level.

enumerator **LOAD_CS**

Cache streaming (likely to be accessed once)

enumerator **LOAD_CV**

Cache as volatile (including cached system lines)

enumerator **LOAD_LDG**

Cache as texture.

enumerator **LOAD_VOLATILE**

Volatile (any memory space)

Enum CacheStoreModifier

- Defined in file_hipcub_backend_rocprim_thread_thread_store.hpp

Enum Documentation

enum **CacheStoreModifier**

Values:

enumerator **STORE_DEFAULT**

Default (no modifier)

enumerator **STORE_WB**

Cache write-back all coherent levels.

enumerator **STORE_CG**

Cache at global level.

enumerator **STORE_CS**

Cache streaming (likely to be accessed once)

enumerator **STORE_WT**

Cache write-through (to system memory)

enumerator **STORE_VOLATILE**

Volatile shared (any memory space)

Enum BlockLoadAlgorithm

- Defined in file_hipcub_backend_rocprim_block_block_load.hpp

Enum Documentation

enum hipcub::BlockLoadAlgorithm

Values:

enumerator **BLOCK_LOAD_DIRECT**

enumerator **BLOCK_LOAD_STRIPED**

enumerator **BLOCK_LOAD_VECTORIZE**

enumerator **BLOCK_LOAD_TRANSPOSE**

enumerator **BLOCK_LOAD_WARP_TRANSPOSE**

enumerator **BLOCK_LOAD_WARP_TRANSPOSE_TIMESLICED**

Enum BlockScanAlgorithm

- Defined in file `hipcub_backend_rocprim_block_block_scan.hpp`

Enum Documentation

enum `hipcub::BlockScanAlgorithm`

Values:

enumerator **BLOCK_SCAN_RAKING**

enumerator **BLOCK_SCAN_RAKING_MEMOIZE**

enumerator **BLOCK_SCAN_WARP_SCANS**

Enum BlockStoreAlgorithm

- Defined in file `hipcub_backend_rocprim_block_block_store.hpp`

Enum Documentation

enum `hipcub::BlockStoreAlgorithm`

Values:

enumerator **BLOCK_STORE_DIRECT**

enumerator **BLOCK_STORE_STRIPED**

enumerator **BLOCK_STORE_VECTORIZE**

enumerator **BLOCK_STORE_TRANSPOSE**

enumerator **BLOCK_STORE_WARP_TRANSPOSE**

enumerator **BLOCK_STORE_WARP_TRANSPOSE_TIMESLICED**

Enum GridMappingStrategy

- Defined in file `hipcub_backend_rocprim_grid_grid_mapping.hpp`

Enum Documentation

enum `hipcub::GridMappingStrategy`

`cub::GridMappingStrategy` enumerates alternative strategies for mapping constant-sized tiles of device-wide data onto a grid of CUDA thread blocks.

Values:

enumerator **GRID_MAPPING_RAKE**

An a “raking” access pattern in which each thread block is assigned a consecutive sequence of input tiles.

Overview The input is evenly partitioned into p segments, where p is constant and corresponds loosely to the number of thread blocks that may actively reside on the target device. Each segment is comprised of consecutive tiles, where a tile is a small, constant-sized unit of input to be processed to completion before the thread block terminates or obtains more work. The kernel invokes p thread blocks, each of which iteratively consumes a segment of n/p elements in tile-size increments.

enumerator **GRID_MAPPING_STRIP_MINE**

An a “strip mining” access pattern in which the input tiles assigned to each thread block are separated by a stride equal to the the extent of the grid.

Overview The input is evenly partitioned into p sets, where p is constant and corresponds loosely to the number of thread blocks that may actively reside on the target device. Each set is comprised of data tiles separated by stride `tiles`, where a tile is a small, constant-sized unit of input to be processed to completion before the thread block terminates or obtains more work. The kernel invokes p thread blocks, each of which iteratively consumes a segment of n/p elements in tile-size increments.

enumerator **GRID_MAPPING_DYNAMIC**

A dynamic “queue-based” strategy for assigning input tiles to thread blocks.

Overview The input is treated as a queue to be dynamically consumed by a grid of thread blocks. Work is atomically dequeued in tiles, where a tile is a unit of input to be processed to completion before the thread block terminates or obtains more work. The grid size p is constant, loosely corresponding to the number of thread blocks that may actively reside on the target device.

Enum WarpLoadAlgorithm

- Defined in file_hipcub_backend_rocprim_warp_warp_load.hpp

Enum Documentation

enum hipcub::WarpLoadAlgorithm

Values:

enumerator WARP_LOAD_DIRECT

enumerator WARP_LOAD_STRIPED

enumerator WARP_LOAD_VECTORIZE

enumerator WARP_LOAD_TRANSPOSE

Enum WarpStoreAlgorithm

- Defined in file_hipcub_backend_rocprim_warp_warp_store.hpp

Enum Documentation

enum hipcub::WarpStoreAlgorithm

Values:

enumerator WARP_STORE_DIRECT

enumerator WARP_STORE_STRIPED

enumerator WARP_STORE_VECTORIZE

enumerator WARP_STORE_TRANSPOSE

2.1.4 Unions

Union BlockMergeSortStrategy::_TempStorage

- Defined in file_hipcub_backend_rocprim_block_block_merge_sort.hpp

Nested Relationships

This union is a nested type of *Template Class BlockMergeSortStrategy*.

Union Documentation

union hipcub::*BlockMergeSortStrategy*::_TempStorage

Shared memory type required by this thread block.

Public Members

KeyT **keys_shared**[ITEMS_PER_TILE + 1]

ValueT **items_shared**[ITEMS_PER_TILE + 1]

Union BlockRunLengthDecode::_TempStorage

- Defined in file `_hipcub_backend_rocprim_block_block_run_length_decode.hpp`

Nested Relationships

This union is a nested type of *Template Class BlockRunLengthDecode*.

Union Documentation

union hipcub::*BlockRunLengthDecode*::_TempStorage

Shared memory type required by this thread block.

Public Members

RunOffsetScanT::TempStorage **offset_scan**

ItemT **run_values**[BLOCK_RUNS]

DecodedOffsetT **run_offsets**[BLOCK_RUNS]

struct hipcub::*BlockRunLengthDecode*::_TempStorage::[anonymous] **runs**

Union WarpExchange::_TempStorage

- Defined in file_hipub_backend_rocprim_warp_warp_exchange.hpp

Nested Relationships

This union is a nested type of *Template Class WarpExchange*.

Union Documentation

```
union hipcub::WarpExchange::_TempStorage
```

Public Members

```
InputT items_shared[ITEMS_PER_TILE]
```

2.1.5 Functions

Template Function AsmThreadLoad

- Defined in file_hipub_backend_rocprim_thread_thread_load.hpp

Function Documentation

```
template<CacheLoadModifier MODIFIER = LOAD_DEFAULT, typename T>
__device__ __forceinline__ T AsmThreadLoad(void *ptr)
```

Template Function AsmThreadStore

- Defined in file_hipub_backend_rocprim_thread_thread_store.hpp

Function Documentation

```
template<CacheStoreModifier MODIFIER = STORE_DEFAULT, typename T>
__device__ __forceinline__ void AsmThreadStore(void *ptr, T val)
```

Function detail::to_BlockHistogramAlgorithm_enum

- Defined in file_hipcub_backend_rocprim_block_block_histogram.hpp

Function Documentation

inline constexpr std::underlying_type<::rocprim::block_histogram_algorithm>::type detail::to_BlockHistogramAlgorithm_enum

Function detail::to_BlockReduceAlgorithm_enum

- Defined in file_hipcub_backend_rocprim_block_block_reduce.hpp

Function Documentation

inline constexpr std::underlying_type<::rocprim::block_reduce_algorithm>::type detail::to_BlockReduceAlgorithm_enum(rocprim::block_reduce_algorithm v)

Function hipcub::BAR

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

__device__ inline void hipcub::BAR(int count)

Template Function hipcub::BFE

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

template<typename **UnsignedBits**>
__device__ inline unsigned int hipcub::BFE(*UnsignedBits* source, unsigned int bit_start, unsigned int num_bits)

Function hipcub::BFI

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

```
__device__ inline void hipcub::BFI(unsigned int &ret, unsigned int x, unsigned int y, unsigned int bit_start,
                                   unsigned int num_bits)
```

Function hipcub::CTA_SYNC

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

```
__device__ inline void hipcub::CTA_SYNC()
```

Function hipcub::Debug

- Defined in file_hipcub_config.hpp

Function Documentation

```
inline hipError_t hipcub::Debug(hipError_t error, const char *filename, int line)
    hipCUB error reporting macro (prints error messages to stderr)
```

Template Function hipcub::detail::convert_result_type

- Defined in file_hipcub_backend_rocprim_thread_thread_operators.hpp

Function Documentation

```
template<class InputIteratorT, class OutputIteratorT, class BinaryFunction>
inline convert_result_type_wrapper<InputIteratorT, OutputIteratorT, BinaryFunction> hipcub::detail::convert_result_type(
```

Template Function hipcub::detail::get_lowest_value

- Defined in file_hipcub_backend_rocprim_device_device_reduce.hpp

Function Documentation

```
template<class T>
inline T hipcub::detail::get_lowest_value()
```

Specialized Template Function `hipcub::detail::get_lowest_value< __half >`

- Defined in file `hipcub_backend_rocprim_device_device_reduce.hpp`

Function Documentation

```
template<>
inline __half hipcub::detail::get_lowest_value<__half>()
```

Specialized Template Function `hipcub::detail::get_lowest_value< hip_bfloat16 >`

- Defined in file `hipcub_backend_rocprim_device_device_reduce.hpp`

Function Documentation

```
template<>
inline hip_bfloat16 hipcub::detail::get_lowest_value<hip_bfloat16>()
```

Template Function `hipcub::detail::get_max_value`

- Defined in file `hipcub_backend_rocprim_device_device_reduce.hpp`

Function Documentation

```
template<class T>
inline T hipcub::detail::get_max_value()
```

Specialized Template Function `hipcub::detail::get_max_value< __half >`

- Defined in file `hipcub_backend_rocprim_device_device_reduce.hpp`

Function Documentation

```
template<>
inline __half hipcub::detail::get_max_value<__half>()
```

Specialized Template Function `hipcub::detail::get_max_value< hip_bfloat16 >`

- Defined in file `hipcub_backend_rocprim_device_device_reduce.hpp`

Function Documentation

```
template<>
inline hip_bfloat16 hipcub::detail::get_max_value<hip_bfloat16>(C)
```

Function hipcub::detail::to_BlockLoadAlgorithm_enum

- Defined in file_hipcub_backend_rocprim_block_block_load.hpp

Function Documentation

```
inline constexpr std::underlying_type<::rocprim::block_load_method>::type hipcub::detail::to_BlockLoadAlgorithm_enum(
v)
```

Function hipcub::detail::to_BlockScanAlgorithm_enum

- Defined in file_hipcub_backend_rocprim_block_block_scan.hpp

Function Documentation

```
inline constexpr std::underlying_type<::rocprim::block_scan_algorithm>::type hipcub::detail::to_BlockScanAlgorithm_enum(
```

Function hipcub::detail::to_BlockStoreAlgorithm_enum

- Defined in file_hipcub_backend_rocprim_block_block_store.hpp

Function Documentation

```
inline constexpr std::underlying_type<::rocprim::block_store_method>::type hipcub::detail::to_BlockStoreAlgorithm_enum(
```

Template Function hipcub::detail::to_double_buffer

- Defined in file_hipcub_backend_rocprim_util_type.hpp

Function Documentation

```
template<typename T>
inline ::rocprim::double_buffer<T> hipcub::detail::to_double_buffer(DoubleBuffer<T> &source)
```

Template Function `hipcub::detail::unsigned_bit_extract`

- Defined in file `hipcub_backend_rocprim_util_ptx.hpp`

Function Documentation

```
template<typename UnsignedBits>
__device__ inline auto hipcub::detail::unsigned_bit_extract(UnsignedBits source, unsigned int bit_start,
                                                         unsigned int num_bits) -> typename
                                                         std::enable_if<sizeof(UnsignedBits) == 8,
                                                         unsigned int>::type
```

Template Function `hipcub::detail::update_double_buffer`

- Defined in file `hipcub_backend_rocprim_util_type.hpp`

Function Documentation

```
template<typename T>
inline void hipcub::detail::update_double_buffer(DoubleBuffer<T> &target, ::rocprim::double_buffer<T>
                                                         &source)
```

Template Function `hipcub::DivideAndRoundUp`

- Defined in file `hipcub_backend_rocprim_util_type.hpp`

Function Documentation

```
template<typename NumeratorT, typename DenominatorT>
__host__ __device__ __forceinline__ constexpr NumeratorT hipcub::DivideAndRoundUp(NumeratorT n,
                                                         DenominatorT d)
```

Function `hipcub::IADD3`

- Defined in file `hipcub_backend_rocprim_util_ptx.hpp`

Function Documentation

```
__device__ inline unsigned int hipcub::IADD3(unsigned int x, unsigned int y, unsigned int z)
```


Function `hipcub::internal::ThreadScanExclusive`

- Defined in file `hipcub_backend_rocprim_thread_thread_scan.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “`hipcub::internal::ThreadScanExclusive`” with arguments () in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<int LENGTH, typename T, typename ScanOp> __device__ __forceinline__ T
↳ ThreadScanExclusive(T inclusive, T exclusive, T *input, T *output, ScanOp scan_op,
↳ Int2Type<LENGTH>)
```

Function `hipcub::internal::ThreadScanInclusive`

- Defined in file `hipcub_backend_rocprim_thread_thread_scan.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “`hipcub::internal::ThreadScanInclusive`” with arguments () in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<int LENGTH, typename T, typename ScanOp> __device__ __forceinline__ T
↳ ThreadScanInclusive(T inclusive, T *input, T *output, ScanOp scan_op, Int2Type
↳ <LENGTH>)
```

Function `hipcub::LaneId`

- Defined in file `hipcub_backend_rocprim_util_ptx.hpp`

Function Documentation

```
__device__ inline unsigned int hipcub::LaneId()
```

Function `hipcub::LaneMaskGe`

- Defined in file `hipcub_backend_rocprim_util_ptx.hpp`

Function Documentation

`__device__ inline uint64_t hipcub::LaneMaskGe()`

Function `hipcub::LaneMaskGt`

- Defined in file `hipcub_backend_rocprim_util_ptx.hpp`

Function Documentation

`__device__ inline uint64_t hipcub::LaneMaskGt()`

Function `hipcub::LaneMaskLe`

- Defined in file `hipcub_backend_rocprim_util_ptx.hpp`

Function Documentation

`__device__ inline uint64_t hipcub::LaneMaskLe()`

Function `hipcub::LaneMaskLt`

- Defined in file `hipcub_backend_rocprim_util_ptx.hpp`

Function Documentation

`__device__ inline uint64_t hipcub::LaneMaskLt()`

Template Function `hipcub::LoadDirectBlocked(int, InputIteratorT, T(&))`

- Defined in file `hipcub_backend_rocprim_block_block_load_func.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::LoadDirectBlocked” with arguments (int, InputIteratorT, T (&)) in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

- `template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void LoadDirectBlocked(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_PER_THREAD])`
- `template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void LoadDirectBlocked(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items)`
- `template<typename T, typename Default, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void LoadDirectBlocked(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items, Default oob_default)`

Template Function `hipcub::LoadDirectBlocked(int, InputIteratorT, T(&), int)`

- Defined in file `hipcub_backend_rocprim_block_block_load_func.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::LoadDirectBlocked” with arguments (int, InputIteratorT, T (&), int) in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void
↳LoadDirectBlocked(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_PER_
↳THREAD])
- template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void
↳LoadDirectBlocked(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_PER_
↳THREAD], int valid_items)
- template<typename T, typename Default, int ITEMS_PER_THREAD, typename
↳InputIteratorT> __device__ void LoadDirectBlocked(int linear_id, InputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items, Default oob_default)
```

Template Function `hipcub::LoadDirectBlocked(int, InputIteratorT, T(&), int, Default)`

- Defined in file `hipcub_backend_rocprim_block_block_load_func.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::LoadDirectBlocked” with arguments (int, InputIteratorT, T (&), int, Default) in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void
↳LoadDirectBlocked(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_PER_
↳THREAD])
- template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void
↳LoadDirectBlocked(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_PER_
↳THREAD], int valid_items)
- template<typename T, typename Default, int ITEMS_PER_THREAD, typename
↳InputIteratorT> __device__ void LoadDirectBlocked(int linear_id, InputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items, Default oob_default)
```

Template Function `hipcub::LoadDirectBlockedVectorized`

- Defined in file `hipcub_backend_rocprim_block_block_load_func.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::LoadDirectBlockedVectorized” with arguments (int, T*, T (&)) in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<typename T, int ITEMS_PER_THREAD> __device__ void
↳LoadDirectBlockedVectorized(int linear_id, T *block_iter, T (&items)[ITEMS_PER_
↳THREAD])
```

Template Function `hipcub::LoadDirectStriped(int, InputIteratorT, T(&))`

- Defined in file `hipcub_backend_rocprim_block_block_load_func.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::LoadDirectStriped” with arguments (int, InputIteratorT, T (&)) in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<int BLOCK_THREADS, typename T, int ITEMS_PER_THREAD, typename
↳InputIteratorT> __device__ void LoadDirectStriped(int linear_id, InputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD])
- template<int BLOCK_THREADS, typename T, int ITEMS_PER_THREAD, typename
↳InputIteratorT> __device__ void LoadDirectStriped(int linear_id, InputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items)
- template<int BLOCK_THREADS, typename T, typename Default, int ITEMS_PER_THREAD,
↳typename InputIteratorT> __device__ void LoadDirectStriped(int linear_id,
↳InputIteratorT block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items, Default
↳oob_default)
```

Template Function `hipcub::LoadDirectStriped(int, InputIteratorT, T(&), int)`

- Defined in file `hipcub_backend_rocprim_block_block_load_func.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::LoadDirectStriped” with arguments (int, InputIteratorT, T (&), int) in doxygen xml output for project “hipCUB” from directory: ./docBin/xml. Potential matches:

```
- template<int BLOCK_THREADS, typename T, int ITEMS_PER_THREAD, typename
↳InputIteratorT> __device__ void LoadDirectStriped(int linear_id, InputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD])
- template<int BLOCK_THREADS, typename T, int ITEMS_PER_THREAD, typename
↳InputIteratorT> __device__ void LoadDirectStriped(int linear_id, InputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items)
- template<int BLOCK_THREADS, typename T, typename Default, int ITEMS_PER_THREAD,
↳typename InputIteratorT> __device__ void LoadDirectStriped(int linear_id,
↳InputIteratorT block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items, Default
↳oob_default)
```

Template Function hipcub::LoadDirectStriped(int, InputIteratorT, T(&), int, Default)

- Defined in file_hipcub_backend_rocprim_block_block_load_func.hpp

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::LoadDirectStriped” with arguments (int, InputIteratorT, T (&), int, Default) in doxygen xml output for project “hipCUB” from directory: ./docBin/xml. Potential matches:

```
- template<int BLOCK_THREADS, typename T, int ITEMS_PER_THREAD, typename
↳InputIteratorT> __device__ void LoadDirectStriped(int linear_id, InputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD])
- template<int BLOCK_THREADS, typename T, int ITEMS_PER_THREAD, typename
↳InputIteratorT> __device__ void LoadDirectStriped(int linear_id, InputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items)
- template<int BLOCK_THREADS, typename T, typename Default, int ITEMS_PER_THREAD,
↳typename InputIteratorT> __device__ void LoadDirectStriped(int linear_id,
↳InputIteratorT block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items, Default
↳oob_default)
```

Template Function hipcub::LoadDirectWarpStriped(int, InputIteratorT, T(&))

- Defined in file_hipcub_backend_rocprim_block_block_load_func.hpp

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::LoadDirectWarpStriped” with arguments (int, InputIteratorT, T (&)) in doxygen xml output for project “hipCUB” from directory: ./docBin/xml. Potential matches:

```
- template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void_
↳LoadDirectWarpStriped(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_
↳PER_THREAD])
- template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void_
↳LoadDirectWarpStriped(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_
↳PER_THREAD], int valid_items)
- template<typename T, typename Default, int ITEMS_PER_THREAD, typename_
↳InputIteratorT> __device__ void LoadDirectWarpStriped(int linear_id, InputIteratorT_
↳block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items, Default oob_default)
```

Template Function hipcub::LoadDirectWarpStriped(int, InputIteratorT, T(&), int)

- Defined in file_hipcub_backend_rocprim_block_block_load_func.hpp

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::LoadDirectWarpStriped” with arguments (int, InputIteratorT, T (&), int) in doxygen xml output for project “hipCUB” from directory: ./docBin/xml. Potential matches:

```
- template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void_
↳LoadDirectWarpStriped(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_
↳PER_THREAD])
- template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void_
↳LoadDirectWarpStriped(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_
↳PER_THREAD], int valid_items)
- template<typename T, typename Default, int ITEMS_PER_THREAD, typename_
↳InputIteratorT> __device__ void LoadDirectWarpStriped(int linear_id, InputIteratorT_
↳block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items, Default oob_default)
```

Template Function hipcub::LoadDirectWarpStriped(int, InputIteratorT, T(&), int, Default)

- Defined in file_hipcub_backend_rocprim_block_block_load_func.hpp

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::LoadDirectWarpStriped” with arguments (int, InputIteratorT, T (&), int, Default) in doxygen xml output for project “hipCUB” from directory: ./docBin/xml. Potential matches:

```
- template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void_
↳LoadDirectWarpStriped(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_
↳PER_THREAD])
- template<typename T, int ITEMS_PER_THREAD, typename InputIteratorT> __device__ void_
↳LoadDirectWarpStriped(int linear_id, InputIteratorT block_iter, T (&items)[ITEMS_
↳PER_THREAD], int valid_items)
- template<typename T, typename Default, int ITEMS_PER_THREAD, typename_
↳InputIteratorT> __device__ void LoadDirectWarpStriped(int linear_id, InputIteratorT_
↳block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items, Default oob_default)
```

Template Function hipcub::MergePath

- Defined in file_hipcub_backend_rocprim_block_block_merge_sort.hpp

Function Documentation

```
template<typename KeyT, typename KeyIteratorT, typename OffsetT, typename BinaryPred>
__device__ __forceinline__ OffsetT hipcub::MergePath(KeyIteratorT keys1, KeyIteratorT keys2, OffsetT
keys1_count, OffsetT keys2_count, OffsetT diag,
BinaryPred binary_pred)
```

Function hipcub::PRMT

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

```
__device__ inline int hipcub::PRMT(unsigned int a, unsigned int b, unsigned int index)
```

Function hipcub::RowMajorTid

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

`__device__ inline int hipcub::RowMajorTid(int block_dim_x, int block_dim_y, int block_dim_z)`

Template Function `hipcub::SerialMerge`

- Defined in file `hipcub_backend_rocprim_block_block_merge_sort.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::SerialMerge” with arguments (KeyT*, int, int, int, int, KeyT (&), int (&), CompareOp) in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<typename KeyT, typename CompareOp, int ITEMS_PER_THREAD> __device__ __
↳forceinline__ void SerialMerge(KeyT *keys_shared, int keys1_beg, int keys2_beg, int_
↳keys1_count, int keys2_count, KeyT (&output)[ITEMS_PER_THREAD], int (&
↳indices)[ITEMS_PER_THREAD], CompareOp compare_op)
```

Function `hipcub::SHL_ADD`

- Defined in file `hipcub_backend_rocprim_util_ptx.hpp`

Function Documentation

`__device__ inline unsigned int hipcub::SHL_ADD(unsigned int x, unsigned int shift, unsigned int addend)`

Function `hipcub::SHR_ADD`

- Defined in file `hipcub_backend_rocprim_util_ptx.hpp`

Function Documentation

`__device__ inline unsigned int hipcub::SHR_ADD(unsigned int x, unsigned int shift, unsigned int addend)`

Template Function `hipcub::ShuffleDown`

- Defined in file `hipcub_backend_rocprim_util_ptx.hpp`

Function Documentation

```
template<int LOGICAL_WARP_THREADS, typename T>
__device__ inline T hipcub::ShuffleDown(T input, int src_offset, int last_thread, unsigned int member_mask)
```

Template Function hipcub::ShuffleIndex

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

```
template<int LOGICAL_WARP_THREADS, typename T>
__device__ inline T hipcub::ShuffleIndex(T input, int src_lane, unsigned int member_mask)
```

Template Function hipcub::ShuffleUp

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

```
template<int LOGICAL_WARP_THREADS, typename T>
__device__ inline T hipcub::ShuffleUp(T input, int src_offset, int first_thread, unsigned int member_mask)
```

Template Function hipcub::StableOddEvenSort

- Defined in file_hipcub_backend_rocprim_thread_thread_sort.hpp

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::StableOddEvenSort” with arguments (KeyT (&), ValueT (&), CompareOp) in doxygen xml output for project “hipCUB” from directory: ./docBin/xml. Potential matches:

```
- template<typename KeyT, typename ValueT, typename CompareOp, int ITEMS_PER_THREAD> _
↳__device__ __forceinline__ void StableOddEvenSort(KeyT (&keys)[ITEMS_PER_THREAD], _
↳ValueT (&items)[ITEMS_PER_THREAD], CompareOp compare_op)
```

Template Function `hipcub::StoreDirectBlocked(int, OutputIteratorT, T(&))`

- Defined in file `hipcub_backend_rocprim_block_block_store_func.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::StoreDirectBlocked” with arguments (int, OutputIteratorT, T (&)) in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<typename T, int ITEMS_PER_THREAD, typename OutputIteratorT> __device__  
  ↪void StoreDirectBlocked(int linear_id, OutputIteratorT block_iter, T (&items)[ITEMS_  
  ↪PER_THREAD])  
- template<typename T, int ITEMS_PER_THREAD, typename OutputIteratorT> __device__  
  ↪void StoreDirectBlocked(int linear_id, OutputIteratorT block_iter, T (&items)[ITEMS_  
  ↪PER_THREAD], int valid_items)
```

Template Function `hipcub::StoreDirectBlocked(int, OutputIteratorT, T(&), int)`

- Defined in file `hipcub_backend_rocprim_block_block_store_func.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::StoreDirectBlocked” with arguments (int, OutputIteratorT, T (&), int) in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<typename T, int ITEMS_PER_THREAD, typename OutputIteratorT> __device__  
  ↪void StoreDirectBlocked(int linear_id, OutputIteratorT block_iter, T (&items)[ITEMS_  
  ↪PER_THREAD])  
- template<typename T, int ITEMS_PER_THREAD, typename OutputIteratorT> __device__  
  ↪void StoreDirectBlocked(int linear_id, OutputIteratorT block_iter, T (&items)[ITEMS_  
  ↪PER_THREAD], int valid_items)
```

Template Function `hipcub::StoreDirectBlockedVectorized`

- Defined in file `hipcub_backend_rocprim_block_block_store_func.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::StoreDirectBlockedVectorized” with arguments (int, T*, T (&)) in doxygen xml output for project “hipCUB” from directory: ./docBin/xml. Potential matches:

```
- template<typename T, int ITEMS_PER_THREAD> __device__ void
↳StoreDirectBlockedVectorized(int linear_id, T *block_iter, T (&items)[ITEMS_PER_
↳THREAD])
```

Template Function hipcub::StoreDirectStriped(int, OutputIteratorT, T(&))

- Defined in file_hipcub_backend_rocprim_block_block_store_func.hpp

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::StoreDirectStriped” with arguments (int, OutputIteratorT, T (&)) in doxygen xml output for project “hipCUB” from directory: ./docBin/xml. Potential matches:

```
- template<int BLOCK_THREADS, typename T, int ITEMS_PER_THREAD, typename
↳OutputIteratorT> __device__ void StoreDirectStriped(int linear_id, OutputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD])
- template<int BLOCK_THREADS, typename T, int ITEMS_PER_THREAD, typename
↳OutputIteratorT> __device__ void StoreDirectStriped(int linear_id, OutputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items)
```

Template Function hipcub::StoreDirectStriped(int, OutputIteratorT, T(&), int)

- Defined in file_hipcub_backend_rocprim_block_block_store_func.hpp

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::StoreDirectStriped” with arguments (int, OutputIteratorT, T (&), int) in doxygen xml output for project “hipCUB” from directory: ./docBin/xml. Potential matches:

```
- template<int BLOCK_THREADS, typename T, int ITEMS_PER_THREAD, typename
↳OutputIteratorT> __device__ void StoreDirectStriped(int linear_id, OutputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD])
- template<int BLOCK_THREADS, typename T, int ITEMS_PER_THREAD, typename
↳OutputIteratorT> __device__ void StoreDirectStriped(int linear_id, OutputIteratorT
↳block_iter, T (&items)[ITEMS_PER_THREAD], int valid_items)
```

Template Function `hipcub::StoreDirectWarpStriped(int, OutputIteratorT, T(&))`

- Defined in file `hipcub_backend_rocprim_block_block_store_func.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::StoreDirectWarpStriped” with arguments (int, OutputIteratorT, T (&)) in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<typename T, int ITEMS_PER_THREAD, typename OutputIteratorT> __device__  
↪void StoreDirectWarpStriped(int linear_id, OutputIteratorT block_iter, T (&  
↪items)[ITEMS_PER_THREAD])  
- template<typename T, int ITEMS_PER_THREAD, typename OutputIteratorT> __device__  
↪void StoreDirectWarpStriped(int linear_id, OutputIteratorT block_iter, T (&  
↪items)[ITEMS_PER_THREAD], int valid_items)
```

Template Function `hipcub::StoreDirectWarpStriped(int, OutputIteratorT, T(&), int)`

- Defined in file `hipcub_backend_rocprim_block_block_store_func.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “hipcub::StoreDirectWarpStriped” with arguments (int, OutputIteratorT, T (&), int) in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<typename T, int ITEMS_PER_THREAD, typename OutputIteratorT> __device__  
↪void StoreDirectWarpStriped(int linear_id, OutputIteratorT block_iter, T (&  
↪items)[ITEMS_PER_THREAD])  
- template<typename T, int ITEMS_PER_THREAD, typename OutputIteratorT> __device__  
↪void StoreDirectWarpStriped(int linear_id, OutputIteratorT block_iter, T (&  
↪items)[ITEMS_PER_THREAD], int valid_items)
```

Template Function `hipcub::Swap`

- Defined in file `hipcub_backend_rocprim_thread_thread_sort.hpp`

Function Documentation

```
template<typename T>  
__device__ __forceinline__ void hipcub::Swap(T &lhs, T &rhs)
```

Function hipcub::WARP_ALL

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

```
__device__ inline int hipcub::WARP_ALL(int predicate, uint64_t member_mask)
```

Function hipcub::WARP_ANY

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

```
__device__ inline int hipcub::WARP_ANY(int predicate, uint64_t member_mask)
```

Function hipcub::WARP_BALLOT

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

```
__device__ inline int64_t hipcub::WARP_BALLOT(int predicate, uint64_t member_mask)
```

Function hipcub::WARP_SYNC

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

```
__device__ inline void hipcub::WARP_SYNC(unsigned int member_mask)
```

Function hipcub::WarpId

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

```
__device__ inline unsigned int hipcub::WarpId()
```

Template Function hipcub::WarpMask

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Function Documentation

```
template<int LOGICAL_WARP_THREADS, int = 0>  
__device__ inline uint64_t hipcub::WarpMask(unsigned int warp_id)
```

Template Function internal::ThreadReduce(T *, ReductionOp, T)

- Defined in file_hipcub_backend_rocprim_thread_thread_reduce.hpp

Function Documentation

```
template<int LENGTH, typename T, typename ReductionOp, bool NoPrefix = false>  
__device__ __forceinline__ T internal::ThreadReduce(T *input, ReductionOp reduction_op, T prefix = T(0))
```

Template Function internal::ThreadReduce(T(&), ReductionOp, T)

- Defined in file_hipcub_backend_rocprim_thread_thread_reduce.hpp

Function Documentation

Warning: doxygenfunction: Unable to resolve function “internal::ThreadReduce” with arguments (T (&), ReductionOp, T) in doxygen xml output for project “hipCUB” from directory: ./docBin/xml. Potential matches:

```
- template<int LENGTH, typename T, typename ReductionOp, bool NoPrefix = false> __  
  ↪device__ __forceinline__ T ThreadReduce(T *input, ReductionOp reduction_op, T_  
  ↪prefix = T(0))  
- template<int LENGTH, typename T, typename ReductionOp> __device__ __forceinline__ T_  
  ↪ThreadReduce(T (&input)[LENGTH], ReductionOp reduction_op)  
- template<int LENGTH, typename T, typename ReductionOp> __device__ __forceinline__ T_  
  ↪ThreadReduce(T (&input)[LENGTH], ReductionOp reduction_op, T prefix)
```

Template Function `internal::ThreadReduce(T(&), ReductionOp)`

- Defined in file `hipcub_backend_rocprim_thread_thread_reduce.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “`internal::ThreadReduce`” with arguments `(T (&), ReductionOp)` in doxygen xml output for project “hipCUB” from directory: `./docBin/xml`. Potential matches:

```
- template<int LENGTH, typename T, typename ReductionOp, bool NoPrefix = false> __
  ↳device__ __forceinline__ T ThreadReduce(T *input, ReductionOp reduction_op, T
  ↳prefix = T(0))
- template<int LENGTH, typename T, typename ReductionOp> __device__ __forceinline__ T
  ↳ThreadReduce(T (&input)[LENGTH], ReductionOp reduction_op)
- template<int LENGTH, typename T, typename ReductionOp> __device__ __forceinline__ T
  ↳ThreadReduce(T (&input)[LENGTH], ReductionOp reduction_op, T prefix)
```

Template Function `MidPoint`

- Defined in file `hipcub_backend_rocprim_util_math.hpp`

Function Documentation

Warning: doxygenfunction: Unable to resolve function “`MidPoint`” with arguments “`(T, T)`”. Candidate function could not be parsed. Parsing error is Error when parsing function declaration. If the function has no return type: Error in declarator or parameters-and-qualifiers Invalid C++ declaration: Expecting “(” in parameters-and-qualifiers. [error at 44] `template<typename T> BEGIN_HIPCUB_NAMESPACE constexpr __device__ __host__ T MidPoint (T begin, T end)` ^ If the function has a return type: Error in declarator or parameters-and-qualifiers If pointer to member declarator: Invalid C++ declaration: Expected ‘:’ in pointer to member (function). [error at 76] `template<typename T> BEGIN_HIPCUB_NAMESPACE constexpr __device__ __host__ T MidPoint (T begin, T end)` ^ If declarator-id: Invalid C++ declaration: Expecting “(” in parameters-and-qualifiers. [error at 76] `template<typename T> BEGIN_HIPCUB_NAMESPACE constexpr __device__ __host__ T MidPoint (T begin, T end)` ^

Template Function `ThreadLoad(InputIteratorT)`

- Defined in file `hipcub_backend_rocprim_thread_thread_load.hpp`

Function Documentation

```
template<CacheLoadModifier MODIFIER = LOAD_DEFAULT, typename InputIteratorT>
__device__ __forceinline__ std::iterator_traits<InputIteratorT>::value_type ThreadLoad(InputIteratorT itr)
```

Template Function ThreadLoad(T *)

- Defined in file_hipcub_backend_rocprim_thread_thread_load.hpp

Function Documentation

```
template<CacheLoadModifier MODIFIER = LOAD_DEFAULT, typename T>
__device__ __forceinline__ T ThreadLoad(T *ptr)
```

Template Function ThreadStore(OutputIteratorT, T)

- Defined in file_hipcub_backend_rocprim_thread_thread_store.hpp

Function Documentation

```
template<CacheStoreModifier MODIFIER = STORE_DEFAULT, typename OutputIteratorT, typename T>
__device__ __forceinline__ void ThreadStore(OutputIteratorT itr, T val)
```

Template Function ThreadStore(T *, T)

- Defined in file_hipcub_backend_rocprim_thread_thread_store.hpp

Function Documentation

```
template<CacheStoreModifier MODIFIER = STORE_DEFAULT, typename T>
__device__ __forceinline__ void ThreadStore(T *ptr, T val)
```

2.1.6 Defines

Define _HipcubLog

- Defined in file_hipcub_backend_rocprim_util_allocator.hpp

Define Documentation

`_HipcubLog(format, ...)`

Define BEGIN_HIPCUB_NAMESPACE

- Defined in file_hipcub_config.hpp

Define Documentation

`BEGIN_HIPCUB_NAMESPACE`

Define END_HIPCUB_NAMESPACE

- Defined in file_hipcub_config.hpp

Define Documentation

`END_HIPCUB_NAMESPACE`

Define HIPCUB_ARCH

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Define Documentation

`HIPCUB_ARCH`

Define HIPCUB_DEVICE

- Defined in file_hipcub_config.hpp

Define Documentation

`HIPCUB_DEVICE`

Define HIPCUB_DEVICE_WARP_THREADS

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Define Documentation

HIPCUB_DEVICE_WARP_THREADS

Define HIPCUB_HOST

- Defined in file_hipcub_config.hpp

Define Documentation

HIPCUB_HOST

Define HIPCUB_HOST_DEVICE

- Defined in file_hipcub_config.hpp

Define Documentation

HIPCUB_HOST_DEVICE

Define HIPCUB_HOST_WARP_THREADS

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Define Documentation

HIPCUB_HOST_WARP_THREADS

Define HIPCUB_MAIN_HEADER_INCLUDED

- Defined in file_hipcub_hipcub.hpp

Define Documentation

HIPCUB_MAIN_HEADER_INCLUDED

Define HIPCUB_MAX_WARP_SIZE

- Defined in file_hipcup_config.hpp

Define Documentation

HIPCUB_MAX_WARP_SIZE

Define HIPCUB_NAMESPACE

- Defined in file_hipcup_config.hpp

Define Documentation

HIPCUB_NAMESPACE

Define HIPCUB_ROCPRIM_API

- Defined in file_hipcup_config.hpp

Define Documentation

HIPCUB_ROCPRIM_API

Define HIPCUB_RUNTIME_FUNCTION

- Defined in file_hipcup_config.hpp

Define Documentation

HIPCUB_RUNTIME_FUNCTION

Define HIPCUB_SHARED_MEMORY

- Defined in file_hipcub_config.hpp

Define Documentation

HIPCUB_SHARED_MEMORY

Define HIPCUB_THREAD_LOAD_USE_CACHE_MODIFIERS

- Defined in file_hipcub_thread_thread_load.hpp

Define Documentation

HIPCUB_THREAD_LOAD_USE_CACHE_MODIFIERS

Define HIPCUB_THREAD_STORE_USE_CACHE_MODIFIERS

- Defined in file_hipcub_thread_thread_store.hpp

Define Documentation

HIPCUB_THREAD_STORE_USE_CACHE_MODIFIERS

Define HIPCUB_WARP_SIZE_32

- Defined in file_hipcub_config.hpp

Define Documentation

HIPCUB_WARP_SIZE_32

Supported warp sizes.

Define HIPCUB_WARP_SIZE_64

- Defined in file_hipcub_config.hpp

Define Documentation

HIPCUB_WARP_SIZE_64

Define HIPCUB_WARP_THREADS

- Defined in file_hipcub_backend_rocprim_util_ptx.hpp

Define Documentation

HIPCUB_WARP_THREADS

Define HipcubDebug

- Defined in file_hipcub_config.hpp

Define Documentation

HipcubDebug(e)

2.1.7 Typedefs

Typedef hipcub::FutureValue

- Defined in file_hipcub_backend_rocprim_util_type.hpp

Typedef Documentation

```
using hipcub::FutureValue = ::rocprim::future_value<T, Iter>
```


INDICES AND TABLES

- genindex
- search

Symbols

`_HipcubLog` (*C macro*), 165

A

`AsmThreadLoad` (*C++ function*), 143

`AsmThreadStore` (*C++ function*), 143

B

`BEGIN_HIPCUB_NAMESPACE` (*C macro*), 165

`BlockHistogram` (*C++ class*), 70

`BlockHistogram::BlockHistogram` (*C++ function*), 70

`BlockHistogram::Composite` (*C++ function*), 70

`BlockHistogram::Histogram` (*C++ function*), 70

`BlockHistogram::InitHistogram` (*C++ function*), 70

`BlockHistogram::TempStorage` (*C++ type*), 70

`BlockHistogramAlgorithm` (*C++ enum*), 136

`BlockHistogramAlgorithm::BLOCK_HISTO_ATOMIC` (*C++ enumerator*), 136

`BlockHistogramAlgorithm::BLOCK_HISTO_SORT` (*C++ enumerator*), 136

`BlockReduce` (*C++ class*), 70

`BlockReduce::BlockReduce` (*C++ function*), 71

`BlockReduce::Reduce` (*C++ function*), 71

`BlockReduce::Sum` (*C++ function*), 71

`BlockReduce::TempStorage` (*C++ type*), 71

`BlockReduceAlgorithm` (*C++ enum*), 137

`BlockReduceAlgorithm::BLOCK_REDUCE_RAKING` (*C++ enumerator*), 137

`BlockReduceAlgorithm::BLOCK_REDUCE_RAKING_COMMUTATIVE_ONLY` (*C++ enumerator*), 137

`BlockReduceAlgorithm::BLOCK_REDUCE_WARP_REDUCTIONS` (*C++ enumerator*), 137

C

`CacheLoadModifier` (*C++ enum*), 137

`CacheLoadModifier::LOAD_CA` (*C++ enumerator*), 137

`CacheLoadModifier::LOAD_CG` (*C++ enumerator*), 137

`CacheLoadModifier::LOAD_CS` (*C++ enumerator*), 137

`CacheLoadModifier::LOAD_CV` (*C++ enumerator*), 137

`CacheLoadModifier::LOAD_DEFAULT` (*C++ enumerator*), 137

`CacheLoadModifier::LOAD_LDG` (*C++ enumerator*), 137

`CacheLoadModifier::LOAD_VOLATILE` (*C++ enumerator*), 137

`CacheStoreModifier` (*C++ enum*), 138

`CacheStoreModifier::STORE_CG` (*C++ enumerator*), 138

`CacheStoreModifier::STORE_CS` (*C++ enumerator*), 138

`CacheStoreModifier::STORE_DEFAULT` (*C++ enumerator*), 138

`CacheStoreModifier::STORE_VOLATILE` (*C++ enumerator*), 138

`CacheStoreModifier::STORE_WB` (*C++ enumerator*), 138

`CacheStoreModifier::STORE_WT` (*C++ enumerator*), 138

D

`detail::to_BlockHistogramAlgorithm_enum` (*C++ function*), 144

`detail::to_BlockReduceAlgorithm_enum` (*C++ function*), 144

E

`END_HIPCUB_NAMESPACE` (*C macro*), 165

H

`hipcub::ArgMax` (*C++ struct*), 9

`hipcub::ArgMax::operator()` (*C++ function*), 10

`hipcub::ArgMin` (*C++ struct*), 10

`hipcub::ArgMin::operator()` (*C++ function*), 10

`hipcub::BAR` (*C++ function*), 144

`hipcub::BaseDigitExtractor` (*C++ struct*), 10

`hipcub::BaseDigitExtractor::ProcessFloatMinusZero` (*C++ function*), 11

`hipcub::BaseDigitExtractor::TraitsT` (*C++ type*), 11

hipcub::BaseDigitExtractor::UnsignedBits (C++ type), 11
 hipcub::BaseDigitExtractor::[anonymous] (C++ enum), 11
 hipcub::BaseDigitExtractor::[anonymous]::FLOAT_KEY (C++ enumerator), 11
 hipcub::BFE (C++ function), 144
 hipcub::BFEDigitExtractor (C++ struct), 11
 hipcub::BFEDigitExtractor::BFEDigitExtractor (C++ function), 12
 hipcub::BFEDigitExtractor::bit_start (C++ member), 12
 hipcub::BFEDigitExtractor::Digit (C++ function), 12
 hipcub::BFEDigitExtractor::num_bits (C++ member), 12
 hipcub::BFEDigitExtractor::ProcessFloatMinusZero (C++ function), 12
 hipcub::BFEDigitExtractor::TraitsT (C++ type), 12
 hipcub::BFEDigitExtractor::UnsignedBits (C++ type), 12
 hipcub::BFEDigitExtractor::[anonymous] (C++ enum), 12
 hipcub::BFEDigitExtractor::[anonymous]::FLOAT_KEY (C++ enumerator), 12
 hipcub::BFI (C++ function), 145
 hipcub::block_raking_layout (C++ struct), 13
 hipcub::block_raking_layout::PlacementPtr (C++ function), 14
 hipcub::block_raking_layout::RakingPtr (C++ function), 14
 hipcub::block_raking_layout::TempStorage (C++ struct), 14, 15
 hipcub::block_raking_layout::TempStorage::Alias (C++ function), 14, 15
 hipcub::block_raking_layout::TempStorage::DeviceWord (C++ function), 14, 15
 hipcub::block_raking_layout::TempStorage::storage (C++ member), 14, 15
 hipcub::block_raking_layout::TempStorage::[anonymous] (C++ function), 14, 15
 hipcub::block_raking_layout::[anonymous] (C++ enum), 13
 hipcub::block_raking_layout::[anonymous]::GRID_ELEMENTS (C++ enumerator), 13
 hipcub::block_raking_layout::[anonymous]::MAX_RAKING_THREADS (C++ enumerator), 13
 hipcub::block_raking_layout::[anonymous]::RAKING_THREADS (C++ enumerator), 13
 hipcub::block_raking_layout::[anonymous]::SEGMENT_LENGTH (C++ enumerator), 13
 hipcub::block_raking_layout::[anonymous]::SHARED_ELEMENTS (C++ enumerator), 13
 hipcub::block_raking_layout::[anonymous]::UNGUARDED (C++ enumerator), 13
 hipcub::block_raking_layout::[anonymous]::USE_SEGMENT_PADDING (C++ enumerator), 13
 hipcub::BlockAdjacentDifference (C++ class), 71
 hipcub::BlockAdjacentDifference::BlockAdjacentDifference (C++ function), 72
 hipcub::BlockAdjacentDifference::FlagHeads (C++ function), 72
 hipcub::BlockAdjacentDifference::FlagHeadsAndTails (C++ function), 72
 hipcub::BlockAdjacentDifference::FlagTails (C++ function), 72
 hipcub::BlockAdjacentDifference::TempStorage (C++ type), 72
 hipcub::BlockDiscontinuity (C++ class), 73
 hipcub::BlockDiscontinuity::BlockDiscontinuity (C++ function), 73
 hipcub::BlockDiscontinuity::FlagHeads (C++ function), 73
 hipcub::BlockDiscontinuity::FlagHeadsAndTails (C++ function), 73, 74
 hipcub::BlockDiscontinuity::FlagTails (C++ function), 73
 hipcub::BlockDiscontinuity::TempStorage (C++ type), 73
 hipcub::BlockExchange (C++ class), 74
 hipcub::BlockExchange::BlockedToStriped (C++ function), 74
 hipcub::BlockExchange::BlockedToWarpStriped (C++ function), 75
 hipcub::BlockExchange::BlockExchange (C++ function), 74
 hipcub::BlockExchange::ScatterToBlocked (C++ function), 75
 hipcub::BlockExchange::ScatterToStriped (C++ function), 75
 hipcub::BlockExchange::ScatterToStripedFlagged (C++ function), 75
 hipcub::BlockExchange::ScatterToStripedGuarded (C++ function), 75
 hipcub::BlockExchange::StripedToBlocked (C++ function), 74
 hipcub::BlockExchange::TempStorage (C++ type), 74
 hipcub::BlockExchange::WarpStripedToBlocked (C++ function), 75
 hipcub::BlockLoad (C++ class), 76
 hipcub::BlockLoad::BlockLoad (C++ function), 76
 hipcub::BlockLoad::Load (C++ function), 76
 hipcub::BlockLoad::TempStorage (C++ type), 76
 hipcub::BlockLoadAlgorithm (C++ enum), 138
 hipcub::BlockLoadAlgorithm::BLOCK_LOAD_DIRECT (C++ enumerator), 138

hipcub::BlockLoadAlgorithm::BLOCK_LOAD_STRIPED (*struct*), 17, 89
 (C++ *enumerator*), 138
 hipcub::BlockLoadAlgorithm::BLOCK_LOAD_TRANSPOSE (*C++ function*), 18, 90
 (C++ *enumerator*), 138
 hipcub::BlockLoadAlgorithm::BLOCK_LOAD_VECTORIZE (*C++ type*), 17, 90
 (C++ *enumerator*), 138
 hipcub::BlockLoadAlgorithm::BLOCK_LOAD_WARP_TRANSPOSE (*C++ member*), 18, 90
 (C++ *enumerator*), 138
 hipcub::BlockLoadAlgorithm::BLOCK_LOAD_WARP_TRANSPOSE_TIMESLICE (*C++ member*), 17, 90
 (C++ *enumerator*), 139
 hipcub::BlockMergeSort (*C++ class*), 76
 hipcub::BlockMergeSort::BlockMergeSort (*C++ function*), 78
 hipcub::BlockMergeSort::get_linear_tid (*C++ function*), 78
 hipcub::BlockMergeSort::Sort (*C++ function*), 78, 79
 hipcub::BlockMergeSort::StableSort (*C++ function*), 80, 81
 hipcub::BlockMergeSortStrategy (*C++ class*), 82
 hipcub::BlockMergeSortStrategy::_TempStorage (*C++ union*), 142
 hipcub::BlockMergeSortStrategy::_TempStorage::ItemsShared (*C++ member*), 142
 hipcub::BlockMergeSortStrategy::_TempStorage::KeysShared (*C++ member*), 142
 hipcub::BlockMergeSortStrategy::BlockMergeSortStrategy (*C++ function*), 83
 hipcub::BlockMergeSortStrategy::get_linear_tid (*C++ function*), 83
 hipcub::BlockMergeSortStrategy::Sort (*C++ function*), 83, 84
 hipcub::BlockMergeSortStrategy::StableSort (*C++ function*), 85, 86
 hipcub::BlockMergeSortStrategy::TempStorage (*C++ struct*), 16, 87
 hipcub::BlockMergeSortStrategy::TempStorage::Alias (*C++ function*), 16, 87
 hipcub::BlockMergeSortStrategy::TempStorage::DeviceWord (*C++ type*), 16, 87
 hipcub::BlockMergeSortStrategy::TempStorage::Storage (*C++ member*), 16, 87
 hipcub::BlockMergeSortStrategy::TempStorage::[anonymous] (*C++ enum*), 16, 87
 hipcub::BlockRadixRank (*C++ class*), 88
 hipcub::BlockRadixRank::BlockRadixRank (*C++ function*), 89
 hipcub::BlockRadixRank::PrefixCallBack (*C++ struct*), 17
 hipcub::BlockRadixRank::PrefixCallBack::operator() (*C++ function*), 17
 hipcub::BlockRadixRank::RankKeys (*C++ function*), 89
 hipcub::BlockRadixRank::TempStorage (*C++ struct*), 17, 89
 hipcub::BlockRadixRank::TempStorage::Alias (*C++ function*), 18, 90
 hipcub::BlockRadixRank::TempStorage::DeviceWord (*C++ type*), 18, 90
 hipcub::BlockRadixRank::TempStorage::storage (*C++ member*), 18, 90
 hipcub::BlockRadixRank::TempStorage::[anonymous] (*C++ enum*), 89
 hipcub::BlockRadixRank::[anonymous]::BINS_TRACKED_PER_THREAD (*C++ enumerator*), 89
 hipcub::BlockRadixRankMatch (*C++ class*), 90
 hipcub::BlockRadixRankMatch::BlockRadixRankMatch (*C++ function*), 90
 hipcub::BlockRadixRankMatch::RankKeys (*C++ function*), 91
 hipcub::BlockRadixRankMatch::TempStorage (*C++ struct*), 18, 91
 hipcub::BlockRadixRankMatch::TempStorage::Alias (*C++ function*), 19, 92
 hipcub::BlockRadixRankMatch::TempStorage::DeviceWord (*C++ type*), 18, 92
 hipcub::BlockRadixRankMatch::TempStorage::storage (*C++ member*), 19, 92
 hipcub::BlockRadixRankMatch::TempStorage::[anonymous] (*C++ enum*), 18, 92
 hipcub::BlockRadixRankMatch::[anonymous] (*C++ enum*), 91
 hipcub::BlockRadixRankMatch::[anonymous]::BINS_TRACKED_PER_THREAD (*C++ enumerator*), 91
 hipcub::BlockRadixSort (*C++ class*), 92
 hipcub::BlockRadixSort::BlockRadixSort (*C++ function*), 93
 hipcub::BlockRadixSort::Sort (*C++ function*), 93
 hipcub::BlockRadixSort::SortBlockedToStriped (*C++ function*), 93
 hipcub::BlockRadixSort::SortDescending (*C++ function*), 93
 hipcub::BlockRadixSort::SortDescendingBlockedToStriped (*C++ function*), 93
 hipcub::BlockRadixSort::TempStorage (*C++ type*), 93
 hipcub::BlockRunLengthDecode (*C++ class*), 94
 hipcub::BlockRunLengthDecode::_TempStorage (*C++ union*), 142
 hipcub::BlockRunLengthDecode::_TempStorage::offset_scan (*C++ member*), 142
 hipcub::BlockRunLengthDecode::_TempStorage::run_offsets (*C++ member*), 142
 hipcub::BlockRunLengthDecode::_TempStorage::run_values (*C++ member*), 142
 hipcub::BlockRunLengthDecode::_TempStorage::runs (*C++ member*), 142

(C++ member), 142

hipcub::BlockRunLengthDecode::BlockRunLengthDecode (C++ enumerator), 139

(C++ function), 96

hipcub::BlockRunLengthDecode::RunLengthDecode (C++ enumerator), 139

(C++ function), 96, 97

hipcub::BlockRunLengthDecode::TempStorage (C++ enumerator), 139

(C++ struct), 19, 97

hipcub::BlockRunLengthDecode::TempStorage::Alias (C++ enumerator), 139

(C++ function), 20, 97

hipcub::BlockRunLengthDecode::TempStorage::DeviceWord (C++ enumerator), 139

(C++ function), 104

(C++ type), 19, 97

hipcub::BlockRunLengthDecode::TempStorage::storage (C++ enumerator), 139

(C++ type), 104

hipcub::BlockRunLengthDecode::TempStorage::[anonymous] (C++ enumerator), 139

(C++ type), 104

(C++ enum), 19, 97

hipcub::BlockScan (C++ class), 98

hipcub::BlockScan::BlockScan (C++ function), 98

hipcub::BlockScan::ExclusiveScan (C++ function), 99, 100

hipcub::BlockScan::ExclusiveSum (C++ function), 99

hipcub::BlockScan::InclusiveScan (C++ function), 98, 99

hipcub::BlockScan::InclusiveSum (C++ function), 98

hipcub::BlockScan::TempStorage (C++ type), 98

hipcub::BlockScanAlgorithm (C++ enum), 139

hipcub::BlockScanAlgorithm::BLOCK_SCAN_RAKING (C++ enumerator), 139

hipcub::BlockScanAlgorithm::BLOCK_SCAN_RAKING_MEMOIZE (C++ enumerator), 139

(C++ function), 105

hipcub::BlockScanAlgorithm::BLOCK_SCAN_WARP_SCANS (C++ enumerator), 139

(C++ function), 105

hipcub::BlockShuffle (C++ class), 100

hipcub::BlockShuffle::BlockShuffle (C++ function), 100

hipcub::BlockShuffle::Down (C++ function), 102

hipcub::BlockShuffle::Offset (C++ function), 100

hipcub::BlockShuffle::Rotate (C++ function), 101

hipcub::BlockShuffle::TempStorage (C++ type), 100

hipcub::BlockShuffle::Up (C++ function), 101

hipcub::BlockStore (C++ class), 103

hipcub::BlockStore::BlockStore (C++ function), 103

hipcub::BlockStore::Store (C++ function), 103

hipcub::BlockStore::TempStorage (C++ type), 103

hipcub::BlockStoreAlgorithm (C++ enum), 139

hipcub::BlockStoreAlgorithm::BLOCK_STORE_DIRECT (C++ enumerator), 139

hipcub::BlockStoreAlgorithm::BLOCK_STORE_STRIPED (C++ enumerator), 139

hipcub::BlockStoreAlgorithm::BLOCK_STORE_TRANSPOSE (C++ enumerator), 139

hipcub::BlockStoreAlgorithm::BLOCK_STORE_VECTORIZE (C++ enumerator), 139

hipcub::BlockStoreAlgorithm::BLOCK_STORE_WARP_TRANSPOSE (C++ enumerator), 139

hipcub::BlockStoreAlgorithm::BLOCK_STORE_WARP_TRANSPOSE_TRANSPOSE (C++ enumerator), 139

hipcub::CacheModifiedInputIterator (C++ class), 104

hipcub::CacheModifiedInputIterator::CacheModifiedInputIterator (C++ function), 104

hipcub::CacheModifiedInputIterator::difference_type (C++ type), 104

hipcub::CacheModifiedInputIterator::iterator_category (C++ type), 104

hipcub::CacheModifiedInputIterator::operator!= (C++ function), 105

hipcub::CacheModifiedInputIterator::operator* (C++ function), 104

hipcub::CacheModifiedInputIterator::operator+ (C++ function), 104

hipcub::CacheModifiedInputIterator::operator++ (C++ function), 104

hipcub::CacheModifiedInputIterator::operator+= (C++ function), 104

hipcub::CacheModifiedInputIterator::operator== (C++ function), 105

hipcub::CacheModifiedInputIterator::operator- (C++ function), 104, 105

hipcub::CacheModifiedInputIterator::operator-- (C++ function), 105

hipcub::CacheModifiedInputIterator::operator-> (C++ function), 105

hipcub::CacheModifiedInputIterator::operator[] (C++ function), 105

hipcub::CacheModifiedInputIterator::pointer (C++ type), 104

hipcub::CacheModifiedInputIterator::ptr (C++ member), 105

hipcub::CacheModifiedInputIterator::reference (C++ type), 104

hipcub::CacheModifiedInputIterator::self_type (C++ type), 104

hipcub::CacheModifiedInputIterator::value_type (C++ type), 104

hipcub::CacheModifiedOutputIterator (C++ class), 105

hipcub::CacheModifiedOutputIterator::CacheModifiedOutputIterator (C++ function), 106

hipcub::CacheModifiedOutputIterator::difference_type (C++ type), 106

hipcub::CacheModifiedOutputIterator::iterator_category (C++ type), 106

hipcub::CacheModifiedOutputIterator::operator!= (C++ function), 107

(C++ class), 25, 107
 hipcub::CachingDeviceAllocator::TotalBytes::free (C++ member), 25, 107
 hipcub::CachingDeviceAllocator::TotalBytes::live (C++ member), 25, 107
 hipcub::CachingDeviceAllocator::TotalBytes::TotalBytes (C++ function), 25, 107
 hipcub::CTA_SYNC (C++ function), 145
 hipcub::Debug (C++ function), 145
 hipcub::detail::convert_result_type (C++ function), 145
 hipcub::detail::get_lowest_value (C++ function), 145
 hipcub::detail::get_lowest_value<__half> (C++ function), 146
 hipcub::detail::get_lowest_value<hip_bfloat16> (C++ function), 146
 hipcub::detail::get_max_value (C++ function), 146
 hipcub::detail::get_max_value<__half> (C++ function), 146
 hipcub::detail::get_max_value<hip_bfloat16> (C++ function), 147
 hipcub::detail::to_BlockLoadAlgorithm_enum (C++ function), 147
 hipcub::detail::to_BlockScanAlgorithm_enum (C++ function), 147
 hipcub::detail::to_BlockStoreAlgorithm_enum (C++ function), 147
 hipcub::detail::to_double_buffer (C++ function), 147
 hipcub::detail::unsigned_bit_extract (C++ function), 148
 hipcub::detail::update_double_buffer (C++ function), 148
 hipcub::DeviceHistogram (C++ struct), 27
 hipcub::DeviceHistogram::HistogramEven (C++ function), 27, 28
 hipcub::DeviceHistogram::HistogramRange (C++ function), 27–29
 hipcub::DeviceHistogram::MultiHistogramEven (C++ function), 27–29
 hipcub::DeviceHistogram::MultiHistogramRange (C++ function), 28, 29
 hipcub::DeviceMergeSort (C++ struct), 30
 hipcub::DeviceMergeSort::SortKeys (C++ function), 30, 31
 hipcub::DeviceMergeSort::SortKeysCopy (C++ function), 30, 31
 hipcub::DeviceMergeSort::SortPairs (C++ function), 30, 31
 hipcub::DeviceMergeSort::SortPairsCopy (C++ function), 30, 31
 hipcub::DeviceMergeSort::StableSortKeys (C++ function), 30, 31
 hipcub::DeviceMergeSort::StableSortPairs (C++ function), 30, 31
 hipcub::DevicePartition (C++ struct), 32
 hipcub::DevicePartition::Flagged (C++ function), 32, 33
 hipcub::DevicePartition::If (C++ function), 32–34
 hipcub::DeviceRadixSort (C++ struct), 34
 hipcub::DeviceRadixSort::SortKeys (C++ function), 35, 36
 hipcub::DeviceRadixSort::SortKeysDescending (C++ function), 35, 36
 hipcub::DeviceRadixSort::SortPairs (C++ function), 35, 36
 hipcub::DeviceRadixSort::SortPairsDescending (C++ function), 35, 36
 hipcub::DeviceReduce (C++ class), 108
 hipcub::DeviceReduce::ArgMax (C++ function), 108, 109
 hipcub::DeviceReduce::ArgMin (C++ function), 108, 109
 hipcub::DeviceReduce::Max (C++ function), 108, 109
 hipcub::DeviceReduce::Min (C++ function), 108, 109
 hipcub::DeviceReduce::Reduce (C++ function), 108
 hipcub::DeviceReduce::ReduceByKey (C++ function), 108, 109
 hipcub::DeviceReduce::Sum (C++ function), 108, 109
 hipcub::DeviceRunLengthEncode (C++ class), 109
 hipcub::DeviceRunLengthEncode::Encode (C++ function), 110
 hipcub::DeviceRunLengthEncode::NonTrivialRuns (C++ function), 110
 hipcub::DeviceScan (C++ class), 110
 hipcub::DeviceScan::ExclusiveScan (C++ function), 111, 112
 hipcub::DeviceScan::ExclusiveScanByKey (C++ function), 111, 113
 hipcub::DeviceScan::ExclusiveSum (C++ function), 111, 112
 hipcub::DeviceScan::ExclusiveSumByKey (C++ function), 111, 112
 hipcub::DeviceScan::InclusiveScan (C++ function), 111, 112
 hipcub::DeviceScan::InclusiveScanByKey (C++ function), 112, 113
 hipcub::DeviceScan::InclusiveSum (C++ function), 111, 112
 hipcub::DeviceScan::InclusiveSumByKey (C++ function), 111, 113
 hipcub::DeviceSegmentedRadixSort (C++ struct),

37

hipcub::DeviceSegmentedRadixSort::SortKeys (C++ function), 38, 39

hipcub::DeviceSegmentedRadixSort::SortKeysDescending (C++ function), 38, 39

hipcub::DeviceSegmentedRadixSort::SortPairs (C++ function), 37, 38

hipcub::DeviceSegmentedRadixSort::SortPairsDescending (C++ function), 37–39

hipcub::DeviceSegmentedReduce (C++ struct), 40

hipcub::DeviceSegmentedReduce::ArgMax (C++ function), 40, 41

hipcub::DeviceSegmentedReduce::ArgMin (C++ function), 40, 41

hipcub::DeviceSegmentedReduce::Max (C++ function), 40, 41

hipcub::DeviceSegmentedReduce::Min (C++ function), 40, 41

hipcub::DeviceSegmentedReduce::Reduce (C++ function), 40

hipcub::DeviceSegmentedReduce::Sum (C++ function), 40, 41

hipcub::DeviceSegmentedSort (C++ struct), 41

hipcub::DeviceSegmentedSort::SortKeys (C++ function), 42, 45

hipcub::DeviceSegmentedSort::SortKeysDescending (C++ function), 42, 45

hipcub::DeviceSegmentedSort::SortPairs (C++ function), 43, 44

hipcub::DeviceSegmentedSort::SortPairsDescending (C++ function), 43–45

hipcub::DeviceSegmentedSort::StableSortKeys (C++ function), 42, 46

hipcub::DeviceSegmentedSort::StableSortKeysDescending (C++ function), 42, 43, 46

hipcub::DeviceSegmentedSort::StableSortPairs (C++ function), 43–46

hipcub::DeviceSegmentedSort::StableSortPairsDescending (C++ function), 44, 46

hipcub::DeviceSelect (C++ class), 113

hipcub::DeviceSelect::Flagged (C++ function), 114

hipcub::DeviceSelect::If (C++ function), 114

hipcub::DeviceSelect::Unique (C++ function), 114

hipcub::DeviceSpmv (C++ class), 115

hipcub::DeviceSpmv::CsrMV (C++ function), 115

hipcub::DeviceSpmv::CsrMVKernel (C++ function), 115

hipcub::DeviceSpmv::CsrMVKernel_MaxThreads (C++ member), 116

hipcub::DeviceSpmv::SpmvParams (C++ struct), 47, 116

hipcub::DeviceSpmv::SpmvParams::alpha (C++ member), 48, 117

hipcub::DeviceSpmv::SpmvParams::beta (C++ member), 48, 117

hipcub::DeviceSpmv::SpmvParams::d_column_indices (C++ member), 47, 116

hipcub::DeviceSpmv::SpmvParams::d_row_end_offsets (C++ member), 47, 116

hipcub::DeviceSpmv::SpmvParams::d_values (C++ member), 47, 117

hipcub::DeviceSpmv::SpmvParams::d_vector_x (C++ member), 47, 117

hipcub::DeviceSpmv::SpmvParams::d_vector_y (C++ member), 47, 117

hipcub::DeviceSpmv::SpmvParams::num_cols (C++ member), 47, 117

hipcub::DeviceSpmv::SpmvParams::num_nonzeros (C++ member), 47, 117

hipcub::DeviceSpmv::SpmvParams::num_rows (C++ member), 47, 117

hipcub::DeviceSpmv::SpmvParams::t_vector_x (C++ member), 48, 117

hipcub::DiscardOutputIterator (C++ class), 117

hipcub::DiscardOutputIterator::difference_type (C++ type), 118

hipcub::DiscardOutputIterator::DiscardOutputIterator (C++ function), 118

hipcub::DiscardOutputIterator::iterator_category (C++ type), 118

hipcub::DiscardOutputIterator::operator void* (C++ function), 119

hipcub::DiscardOutputIterator::operator!= (C++ function), 119

hipcub::DiscardOutputIterator::operator* (C++ function), 118

hipcub::DiscardOutputIterator::operator+ (C++ function), 118

hipcub::DiscardOutputIterator::operator++ (C++ function), 118

hipcub::DiscardOutputIterator::operator+= (C++ function), 119

hipcub::DiscardOutputIterator::operator= (C++ function), 119

hipcub::DiscardOutputIterator::operator== (C++ function), 119

hipcub::DiscardOutputIterator::operator- (C++ function), 119

hipcub::DiscardOutputIterator::operator-= (C++ function), 119

hipcub::DiscardOutputIterator::operator-> (C++ function), 119

hipcub::DiscardOutputIterator::operator<< (C++ function), 119

hipcub::DiscardOutputIterator::operator[] (C++ function), 119

hipcub::DiscardOutputIterator::pointer (C++

- type*), 118
 hipcub::DiscardOutputIterator::reference (C++ *type*), 118
 hipcub::DiscardOutputIterator::self_type (C++ *type*), 118
 hipcub::DiscardOutputIterator::value_type (C++ *type*), 118
 hipcub::DivideAndRoundUp (C++ *function*), 148
 hipcub::DoubleBuffer (C++ *struct*), 48
 hipcub::DoubleBuffer::Alternate (C++ *function*), 48
 hipcub::DoubleBuffer::Current (C++ *function*), 48
 hipcub::DoubleBuffer::d_buffers (C++ *member*), 48
 hipcub::DoubleBuffer::DoubleBuffer (C++ *function*), 48
 hipcub::DoubleBuffer::selector (C++ *member*), 48
 hipcub::Equality (C++ *struct*), 49
 hipcub::Equality::operator() (C++ *function*), 49
 hipcub::FutureValue (C++ *type*), 169
 hipcub::GridBarrier (C++ *class*), 120
 hipcub::GridBarrier::d_sync (C++ *member*), 120
 hipcub::GridBarrier::GridBarrier (C++ *function*), 120
 hipcub::GridBarrier::Sync (C++ *function*), 120
 hipcub::GridBarrier::SyncFlag (C++ *type*), 120
 hipcub::GridBarrierLifetime (C++ *class*), 120
 hipcub::GridBarrierLifetime::~~GridBarrierLifetime (C++ *function*), 121
 hipcub::GridBarrierLifetime::d_sync (C++ *member*), 121
 hipcub::GridBarrierLifetime::GridBarrierLifetime (C++ *function*), 121
 hipcub::GridBarrierLifetime::HostReset (C++ *function*), 121
 hipcub::GridBarrierLifetime::Setup (C++ *function*), 121
 hipcub::GridBarrierLifetime::Sync (C++ *function*), 121
 hipcub::GridBarrierLifetime::sync_bytes (C++ *member*), 121
 hipcub::GridBarrierLifetime::SyncFlag (C++ *type*), 121
 hipcub::GridEvenShare (C++ *struct*), 49
 hipcub::GridEvenShare::block_end (C++ *member*), 50
 hipcub::GridEvenShare::block_offset (C++ *member*), 50
 hipcub::GridEvenShare::block_stride (C++ *member*), 50
 hipcub::GridEvenShare::BlockInit (C++ *function*), 50
 hipcub::GridEvenShare::DispatchInit (C++ *function*), 49
 hipcub::GridEvenShare::grid_size (C++ *member*), 50
 hipcub::GridEvenShare::GridEvenShare (C++ *function*), 49
 hipcub::GridEvenShare::num_items (C++ *member*), 50
 hipcub::GridMappingStrategy (C++ *enum*), 140
 hipcub::GridMappingStrategy::GRID_MAPPING_DYNAMIC (C++ *enumerator*), 140
 hipcub::GridMappingStrategy::GRID_MAPPING_RAKE (C++ *enumerator*), 140
 hipcub::GridMappingStrategy::GRID_MAPPING_STRIP_MINE (C++ *enumerator*), 140
 hipcub::GridQueue (C++ *class*), 121
 hipcub::GridQueue::AllocationSize (C++ *function*), 123
 hipcub::GridQueue::Drain (C++ *function*), 122
 hipcub::GridQueue::Fill (C++ *function*), 122
 hipcub::GridQueue::FillAndResetDrain (C++ *function*), 122
 hipcub::GridQueue::FillSize (C++ *function*), 122
 hipcub::GridQueue::GridQueue (C++ *function*), 122
 hipcub::GridQueue::ResetDrain (C++ *function*), 122
 hipcub::GridQueue::ResetFill (C++ *function*), 122
 hipcub::IADD3 (C++ *function*), 148
 hipcub::If (C++ *struct*), 51
 hipcub::If::Type (C++ *type*), 51
 hipcub::Inequality (C++ *struct*), 51
 hipcub::Inequality::operator() (C++ *function*), 51
 hipcub::InequalityWrapper (C++ *struct*), 51
 hipcub::InequalityWrapper::InequalityWrapper (C++ *function*), 52
 hipcub::InequalityWrapper::op (C++ *member*), 52
 hipcub::InequalityWrapper::operator() (C++ *function*), 52
 hipcub::Int2Type (C++ *struct*), 52
 hipcub::Int2Type::[anonymous] (C++ *enum*), 52
 hipcub::Int2Type::[anonymous]::VALUE (C++ *enumerator*), 52
 hipcub::IsPointer (C++ *struct*), 52
 hipcub::IsPointer::VALUE (C++ *member*), 53
 hipcub::IsVolatile (C++ *struct*), 53
 hipcub::IsVolatile::VALUE (C++ *member*), 53
 hipcub::LaneId (C++ *function*), 149
 hipcub::LaneMaskGe (C++ *function*), 150
 hipcub::LaneMaskGt (C++ *function*), 150
 hipcub::LaneMaskLe (C++ *function*), 150
 hipcub::LaneMaskLt (C++ *function*), 150
 hipcub::Log2 (C++ *struct*), 53
 hipcub::Log2::VALUE (C++ *member*), 53
 hipcub::Max (C++ *struct*), 54

hipcub::Max::operator() (C++ function), 54
 hipcub::MergePath (C++ function), 155
 hipcub::Min (C++ struct), 54
 hipcub::Min::operator() (C++ function), 54
 hipcub::PowerOfTwo (C++ struct), 54
 hipcub::PowerOfTwo::VALUE (C++ member), 54
 hipcub::PRMT (C++ function), 155
 hipcub::RadixSortTwiddle (C++ struct), 55
 hipcub::RadixSortTwiddle::DefaultKey (C++ function), 55
 hipcub::RadixSortTwiddle::In (C++ function), 55
 hipcub::RadixSortTwiddle::Out (C++ function), 55
 hipcub::RadixSortTwiddle::TraitsT (C++ type), 55
 hipcub::RadixSortTwiddle::UnsignedBits (C++ type), 55
 hipcub::RemoveQualifiers (C++ struct), 55
 hipcub::RemoveQualifiers::Type (C++ type), 55
 hipcub::RowMajorTid (C++ function), 156
 hipcub::ShiftDigitExtractor (C++ struct), 56
 hipcub::ShiftDigitExtractor::bit_start (C++ member), 56
 hipcub::ShiftDigitExtractor::Digit (C++ function), 56
 hipcub::ShiftDigitExtractor::mask (C++ member), 56
 hipcub::ShiftDigitExtractor::ProcessFloatMinusZero (C++ function), 57
 hipcub::ShiftDigitExtractor::ShiftDigitExtractor (C++ function), 56
 hipcub::ShiftDigitExtractor::TraitsT (C++ type), 56
 hipcub::ShiftDigitExtractor::UnsignedBits (C++ type), 56
 hipcub::ShiftDigitExtractor::[anonymous] (C++ enum), 56
 hipcub::ShiftDigitExtractor::[anonymous]::FLOAT_KEY (C++ enumerator), 56
 hipcub::SHL_ADD (C++ function), 156
 hipcub::SHR_ADD (C++ function), 156
 hipcub::ShuffleDown (C++ function), 157
 hipcub::ShuffleIndex (C++ function), 157
 hipcub::ShuffleUp (C++ function), 157
 hipcub::Sum (C++ struct), 57
 hipcub::Sum::operator() (C++ function), 57
 hipcub::Swap (C++ function), 161
 hipcub::TexObjInputIterator (C++ class), 123
 hipcub::TexObjInputIterator::~TexObjInputIterator (C++ function), 123
 hipcub::TexObjInputIterator::BindTexture (C++ function), 123
 hipcub::TexObjInputIterator::TexObjInputIterator (C++ function), 123
 hipcub::TexObjInputIterator::UnbindTexture (C++ function), 123
 hipcub::TexRefInputIterator (C++ class), 124
 hipcub::TexRefInputIterator::~TexRefInputIterator (C++ function), 124
 hipcub::TexRefInputIterator::BindTexture (C++ function), 124
 hipcub::TexRefInputIterator::TexRefInputIterator (C++ function), 124
 hipcub::TexRefInputIterator::UnbindTexture (C++ function), 124
 hipcub::Uninitialized (C++ struct), 57
 hipcub::Uninitialized::Alias (C++ function), 58
 hipcub::Uninitialized::DeviceWord (C++ type), 57
 hipcub::Uninitialized::storage (C++ member), 58
 hipcub::Uninitialized::[anonymous] (C++ enum), 57
 hipcub::Uninitialized::[anonymous]::WORDS (C++ enumerator), 57
 hipcub::WARP_ALL (C++ function), 161
 hipcub::WARP_ANY (C++ function), 161
 hipcub::WARP_BALLOT (C++ function), 161
 hipcub::WARP_SYNC (C++ function), 161
 hipcub::WarpExchange (C++ class), 124
 hipcub::WarpExchange::_TempStorage (C++ union), 143
 hipcub::WarpExchange::_TempStorage::items_shared (C++ member), 143
 hipcub::WarpExchange::BlockedToStriped (C++ function), 124
 hipcub::WarpExchange::ScatterToStriped (C++ function), 125
 hipcub::WarpExchange::StripedToBlocked (C++ function), 124
 hipcub::WarpExchange::TempStorage (C++ struct), 58, 125
 hipcub::WarpExchange::TempStorage::Alias (C++ function), 59, 125
 hipcub::WarpExchange::TempStorage::DeviceWord (C++ type), 58, 125
 hipcub::WarpExchange::TempStorage::storage (C++ member), 59, 125
 hipcub::WarpExchange::TempStorage::[anonymous] (C++ enum), 58, 125
 hipcub::WarpExchange::WarpExchange (C++ function), 124
 hipcub::WarpId (C++ function), 162
 hipcub::WarpLoad (C++ class), 126
 hipcub::WarpLoad::Load (C++ function), 126
 hipcub::WarpLoad::LoadInternal (C++ struct), 59
 hipcub::WarpLoad::LoadInternal<WARP_LOAD_DIRECT> (C++ struct), 59
 hipcub::WarpLoad::LoadInternal<WARP_LOAD_DIRECT>::linear_t

(C++ member), 60

hipcub::WarpLoad::LoadInternal<WARP_LOAD_DIRECT> (C++ function), 60

hipcub::WarpLoad::LoadInternal<WARP_LOAD_DIRECT> (C++ function), 60

hipcub::WarpLoad::LoadInternal<WARP_LOAD_DIRECT> (C++ type), 60

hipcub::WarpLoad::LoadInternal<WARP_LOAD_STRIPED> (C++ struct), 60

hipcub::WarpLoad::LoadInternal<WARP_LOAD_STRIPED> (C++ member), 61

hipcub::WarpLoad::LoadInternal<WARP_LOAD_STRIPED> (C++ function), 61

hipcub::WarpLoad::LoadInternal<WARP_LOAD_STRIPED> (C++ function), 61

hipcub::WarpLoad::LoadInternal<WARP_LOAD_STRIPED> (C++ type), 61

hipcub::WarpLoad::LoadInternal<WARP_LOAD_TRANSPOSE> (C++ struct), 61

hipcub::WarpLoad::LoadInternal<WARP_LOAD_TRANSPOSE> (C++ member), 62

hipcub::WarpLoad::LoadInternal<WARP_LOAD_TRANSPOSE> (C++ function), 62

hipcub::WarpLoad::LoadInternal<WARP_LOAD_TRANSPOSE> (C++ function), 62

hipcub::WarpLoad::LoadInternal<WARP_LOAD_TRANSPOSE> (C++ member), 62

hipcub::WarpLoad::LoadInternal<WARP_LOAD_TRANSPOSE> (C++ type), 62

hipcub::WarpLoad::LoadInternal<WARP_LOAD_TRANSPOSE> (C++ type), 62

hipcub::WarpLoad::LoadInternal<WARP_LOAD_TRANSPOSE> (C++ struct), 62

hipcub::WarpLoad::LoadInternal<WARP_LOAD_VECTORIZE> (C++ member), 63

hipcub::WarpLoad::LoadInternal<WARP_LOAD_VECTORIZE> (C++ function), 63

hipcub::WarpLoad::LoadInternal<WARP_LOAD_VECTORIZE> (C++ function), 63

hipcub::WarpLoad::LoadInternal<WARP_LOAD_VECTORIZE> (C++ type), 63

hipcub::WarpLoad::TempStorage (C++ struct), 64, 126

hipcub::WarpLoad::TempStorage::Alias (C++ function), 64, 127

hipcub::WarpLoad::TempStorage::DeviceWord (C++ type), 64, 126

hipcub::WarpLoad::TempStorage::storage (C++ member), 64, 127

hipcub::WarpLoad::TempStorage:::[anonymous] (C++ enum), 64, 126

hipcub::WarpLoad::WarpLoad (C++ function), 126

hipcub::WarpLoadAlgorithm (C++ enum), 141

hipcub::WarpLoadAlgorithm::WARP_LOAD_DIRECT (C++ enumerator), 141

hipcub::WarpLoadAlgorithm::WARP_LOAD_STRIPED (C++ enumerator), 141

hipcub::WarpLoadAlgorithm::WARP_LOAD_TRANSPOSE (C++ enumerator), 141

hipcub::WarpLoadAlgorithm::WARP_LOAD_VECTORIZE (C++ enumerator), 141

hipcub::WarpMask (C++ function), 162

hipcub::WarpMergeSort (C++ class), 127

hipcub::WarpMergeSort::get_linear_tid (C++ function), 129

hipcub::WarpMergeSort::get_member_mask (C++ function), 129

hipcub::WarpMergeSort::Sort (C++ function), 129, 130

hipcub::WarpMergeSort::StableSort (C++ function), 130–132

hipcub::WarpMergeSort::WarpMergeSort (C++ function), 129

hipcub::WarpReduce (C++ class), 133

hipcub::WarpReduce::HeadSegmentedReduce (C++ function), 133

hipcub::WarpReduce::HeadSegmentedSum (C++ function), 133

hipcub::WarpReduce::Internal (C++ function), 133

hipcub::WarpReduce::Reduce (C++ function), 133

hipcub::WarpReduce::TempStorage::Sum (C++ function), 133

hipcub::WarpReduce::TailSegmentedReduce (C++ function), 133

hipcub::WarpReduce::TailSegmentedSum (C++ function), 133

hipcub::WarpReduce::TempStorage (C++ type), 133

hipcub::WarpReduce::WarpReduce (C++ function), 133

hipcub::WarpReduce::WarpReduce (C++ class), 134

hipcub::WarpReduce::Broadcast (C++ function), 135

hipcub::WarpReduce::ExclusiveScan (C++ function), 134

hipcub::WarpReduce::ExclusiveSum (C++ function), 134

hipcub::WarpReduce::InclusiveScan (C++ function), 134

hipcub::WarpReduce::InclusiveSum (C++ function), 134

hipcub::WarpReduce::Scan (C++ function), 135

hipcub::WarpReduce::TempStorage (C++ type), 134

hipcub::WarpReduce::WarpScan (C++ function), 134

hipcub::WarpStore (C++ class), 135

hipcub::WarpStore::Store (C++ function), 135

hipcub::WarpStore::StoreInternal (C++ struct), 65

hipcub::WarpStore::StoreInternal<WARP_STORE_DIRECT> (C++ struct), 65

hipcub::WarpStore::StoreInternal<WARP_STORE_DIRECT>::linear (C++ member), 65

hipcub::WarpStore::StoreInternal<WARP_STORE_DIRECT>::StoreStoreAlgorithm::WARP_STORE_STRIPED
 (C++ function), 65 (C++ enumerator), 141
 hipcub::WarpStore::StoreInternal<WARP_STORE_DIRECT>::StoreStoreAlgorithm::WARP_STORE_TRANSPOSE
 (C++ function), 65 (C++ enumerator), 141
 hipcub::WarpStore::StoreInternal<WARP_STORE_DIRECT>::TempStorageAlgorithm::WARP_STORE_VECTORIZE
 (C++ type), 65 (C++ enumerator), 141
 hipcub::WarpStore::StoreInternal<WARP_STORE_STRIPED>::ARCH (C macro), 165
 (C++ struct), 66 HIPCUB_DEVICE (C macro), 165
 hipcub::WarpStore::StoreInternal<WARP_STORE_STRIPED>::DEVICE_WARP_THREADS (C macro), 166
 (C++ member), 66 HIPCUB_HOST (C macro), 166
 hipcub::WarpStore::StoreInternal<WARP_STORE_STRIPED>::HOST_DEVICE (C macro), 166
 (C++ function), 66 HIPCUB_HOST_WARP_THREADS (C macro), 166
 hipcub::WarpStore::StoreInternal<WARP_STORE_STRIPED>::MATH_HEADER_INCLUDED (C macro), 167
 (C++ function), 66 HIPCUB_MAX_WARP_SIZE (C macro), 167
 hipcub::WarpStore::StoreInternal<WARP_STORE_STRIPED>::MESSAGE_SIZE (C macro), 167
 (C++ type), 66 HIPCUB_ROCPRI_API (C macro), 167
 hipcub::WarpStore::StoreInternal<WARP_STORE_TRANSPOSE>::RUNTIME_FUNCTION (C macro), 167
 (C++ struct), 67 HIPCUB_SHARED_MEMORY (C macro), 168
 hipcub::WarpStore::StoreInternal<WARP_STORE_TRANSPOSE>::THREAD_LOAD_USE_CACHE_MODIFIERS (C
 (C++ member), 67 macro), 168
 hipcub::WarpStore::StoreInternal<WARP_STORE_TRANSPOSE>::STORE_USE_CACHE_MODIFIERS (C
 (C++ function), 67 macro), 168
 hipcub::WarpStore::StoreInternal<WARP_STORE_TRANSPOSE>::WARP_SIZE_12 (C macro), 168
 (C++ function), 67 HIPCUB_WARP_SIZE_64 (C macro), 169
 hipcub::WarpStore::StoreInternal<WARP_STORE_TRANSPOSE>::WARP_THREADS_64 (C macro), 169
 (C++ member), 67 HipcubDebug (C macro), 169
 hipcub::WarpStore::StoreInternal<WARP_STORE_TRANSPOSE>::TempStorage
 (C++ type), 67
 hipcub::WarpStore::StoreInternal<WARP_STORE_TRANSPOSE>::WarpExchangeT
 (C++ type), 67 internal::ThreadLoad (C++ function), 162
 hipcub::WarpStore::StoreInternal<WARP_STORE_VECTORIZE>
 (C++ struct), 68 ThreadLoad (C++ function), 164
 hipcub::WarpStore::StoreInternal<WARP_STORE_VECTORIZE>::linear_tid
 (C++ member), 68 ThreadStore (C++ function), 164
 hipcub::WarpStore::StoreInternal<WARP_STORE_VECTORIZE>::Store
 (C++ function), 68
 hipcub::WarpStore::StoreInternal<WARP_STORE_VECTORIZE>::StoreInternal
 (C++ function), 68
 hipcub::WarpStore::StoreInternal<WARP_STORE_VECTORIZE>::TempStorage
 (C++ type), 68
 hipcub::WarpStore::TempStorage (C++ struct), 69,
 136
 hipcub::WarpStore::TempStorage::Alias (C++
 function), 69, 136
 hipcub::WarpStore::TempStorage::DeviceWord
 (C++ type), 69, 136
 hipcub::WarpStore::TempStorage::storage
 (C++ member), 69, 136
 hipcub::WarpStore::TempStorage::[anonymous]
 (C++ enum), 69, 136
 hipcub::WarpStore::WarpStore (C++ function), 135
 hipcub::WarpStoreAlgorithm (C++ enum), 141
 hipcub::WarpStoreAlgorithm::WARP_STORE_DIRECT
 (C++ enumerator), 141