Intel® Offload Runtime Library

Generated by Doxygen 1.8.6

Tue Apr 8 2014 11:54:44

FTC Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel.

Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice revision #20110804

Trademarks

Intel, Xeon, and Intel Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries.

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 COI Namespace Reference

Functions

- bool init (void)
- void fini (void)

Variables

- bool is_available
- static void * lib_handle
- COIRESULT(* EngineGetCount)(COI_ISA_TYPE, uint32_t *)
- COIRESULT(* EngineGetHandle)(COI_ISA_TYPE, uint32_t, COIENGINE *)
- COIRESULT(* ProcessCreateFromMemory)(COIENGINE, const char *, const void *, uint64_t, int, const char **, uint8_t, const char **, uint8_t, const char *, uint64_t, const char *, const char *, uint64_t, COIPROCESS *)
- COIRESULT(* ProcessDestroy)(COIPROCESS, int32_t, uint8_t, int8_t *, uint32_t *)
- COIRESULT(* ProcessGetFunctionHandles)(COIPROCESS, uint32_t, const char **, COIFUNCTION *)
- COIRESULT(* ProcessLoadLibraryFromMemory)(COIPROCESS, const void *, uint64_t, const char *, const char *, const char *, uint64_t, uint32_t, COILIBRARY *)
- COIRESULT(* ProcessRegisterLibraries)(uint32_t, const void **, const uint64_t *, const char **, const uint64_t *)
- COIRESULT(* PipelineCreate)(COIPROCESS, COI_CPU_MASK, uint32_t, COIPIPELINE *)
- COIRESULT(* PipelineDestroy)(COIPIPELINE)
- COIRESULT(* PipelineRunFunction)(COIPIPELINE, COIFUNCTION, uint32_t, const COIBUFFER *, const COI_ACCESS_FLAGS *, uint32_t, const COIEVENT *, const void *, uint16_t, void *, uint16_t, COIEVENT *)
- COIRESULT(* BufferCreate)(uint64_t, COI_BUFFER_TYPE, uint32_t, const void *, uint32_t, const COIPROCESS *, COIBUFFER *)
- COIRESULT(* BufferCreateFromMemory)(uint64_t, COI_BUFFER_TYPE, uint32_t, void *, uint32_t, const COIPROCESS *, COIBUFFER *)
- COIRESULT(* BufferDestroy)(COIBUFFER)
- COIRESULT(* BufferMap)(COIBUFFER, uint64_t, uint64_t, COI_MAP_TYPE, uint32_t, const COIEVENT *, COIEVENT *, COIMAPINSTANCE *, void **)
- COIRESULT(* BufferUnmap)(COIMAPINSTANCE, uint32_t, const COIEVENT *, COIEVENT *)
- COIRESULT(* BufferWrite)(COIBUFFER, uint64_t, const void *, uint64_t, COI_COPY_TYPE, uint32_t, const COIEVENT *, COIEVENT *)
- COIRESULT(* BufferRead)(COIBUFFER, uint64_t, void *, uint64_t, COI_COPY_TYPE, uint32_t, const COIEVENT *, COIEVENT *)
- COIRESULT(* BufferCopy)(COIBUFFER, COIBUFFER, uint64_t, uint64_t, uint64_t, COI_COPY_TYPE, uint32_t, const COIEVENT *, COIEVENT *)
- COIRESULT(* BufferGetSinkAddress)(COIBUFFER, uint64_t *)

- COIRESULT(∗ BufferSetState)(COIBUFFER, COIPROCESS, COI_BUFFER_STATE, COI_BUFFER_MOVE_FLAG, uint32_t, const COIEVENT ∗, COIEVENT ∗)
- COIRESULT(∗ EventWait)(uint16_t, const COIEVENT ∗, int32_t, uint8_t, uint32_t ∗, uint32_t ∗)
- uint64_t(∗ PerfGetCycleFrequency)(void)

5.1.1 Function Documentation

void COI::fini (void)

Definition at line 338 of file coi_client.cpp.
Referenced by __offload_fini_library(), and init().

bool COI::init (void)

Definition at line 75 of file coi_client.cpp.
Referenced by __offload_init_library_once().

5.1.2 Variable Documentation

COIRESULT(∗ COI::BufferCopy)(COIBUFFER, COIBUFFER, uint64_t, uint64_t, uint64_t, COI_COPY_TYPE, uint32_t, const COIEVENT ∗, COIEVENT ∗)

Definition at line 63 of file coi_client.cpp.
Referenced by init(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), and OffloadDescriptor::send_pointer_data().

COIRESULT(∗ COI::BufferCreate)(uint64_t, COI_BUFFER_TYPE, uint32_t, const void ∗, uint32_t, const COIPROCESS ∗, COIBUFFER ∗)

Definition at line 50 of file coi_client.cpp.
Referenced by OffloadDescriptor::alloc_ptr_data(), init(), Engine::init_ptr_data(), OffloadDescriptor::offload_stack_memory_manager(), and OffloadDescriptor::setup_misc_data().

COIRESULT(∗ COI::BufferCreateFromMemory)(uint64_t, COI_BUFFER_TYPE, uint32_t, void ∗, uint32_t, const COIPROCESS ∗, COIBUFFER ∗)

Definition at line 52 of file coi_client.cpp.
Referenced by OffloadDescriptor::alloc_ptr_data(), init(), and OffloadDescriptor::init_static_ptr_data().

COIRESULT(∗ COI::BufferDestroy)(COIBUFFER)

Definition at line 55 of file coi_client.cpp.
Referenced by init(), Engine::init_ptr_data(), and OffloadDescriptor::offload_finish().

COIRESULT(∗ COI::BufferGetSinkAddress)(COIBUFFER, uint64_t ∗)

Definition at line 65 of file coi_client.cpp.
Referenced by init(), and OffloadDescriptor::init_mic_address().

COIRESULT(∗ COI::BufferMap)(COIBUFFER, uint64_t, uint64_t, COI_MAP_TYPE, uint32_t, const COIEVENT ∗, COIEVENT ∗, COIMAPINSTANCE ∗, void ∗∗)

Definition at line 56 of file coi_client.cpp.
Referenced by OffloadDescriptor::gather_copyin_data(), init(), Engine::init_ptr_data(), and OffloadDescriptor::scatter_copyout_data().

COIRESULT(∗ COI::BufferRead)(COIBUFFER, uint64_t, void ∗, uint64_t, COI_COPY_TYPE, uint32_t, const COIEVENT ∗, COIEVENT ∗)

Definition at line 61 of file coi_client.cpp.

Referenced by init(), OffloadDescriptor::receive_pointer_data(), and OffloadDescriptor::recieve_noncontiguous_pointer_data().

COIRESULT(∗ COI::BufferSetState)(COIBUFFER, COIPROCESS, COI_BUFFER_STATE, COI_BUFFER_MOVE_FLAG, uint32_t, const COIEVENT ∗, COIEVENT ∗)

Definition at line 66 of file coi_client.cpp.

Referenced by OffloadDescriptor::alloc_ptr_data(), init(), and OffloadDescriptor::offload_stack_memory_manager().

COIRESULT(∗ COI::BufferUnmap)(COIMAPINSTANCE, uint32_t, const COIEVENT ∗, COIEVENT ∗)

Definition at line 58 of file coi_client.cpp.

Referenced by OffloadDescriptor::gather_copyin_data(), init(), Engine::init_ptr_data(), and OffloadDescriptor::scatter_copyout_data().

COIRESULT(∗ COI::BufferWrite)(COIBUFFER, uint64_t, const void ∗, uint64_t, COI_COPY_TYPE, uint32_t, const COIEVENT ∗, COIEVENT ∗)

Definition at line 59 of file coi_client.cpp.

Referenced by init(), OffloadDescriptor::nullify_target_stack(), OffloadDescriptor::send_noncontiguous_pointer_data(), and OffloadDescriptor::send_pointer_data().

COIRESULT(∗ COI::EngineGetCount)(COI_ISA_TYPE, uint32_t ∗)

Definition at line 25 of file coi_client.cpp.

Referenced by __offload_init_library_once(), and init().

COIRESULT(∗ COI::EngineGetHandle)(COI_ISA_TYPE, uint32_t, COIENGINE ∗)

Definition at line 26 of file coi_client.cpp.

Referenced by __offload_init_library_once(), init(), and Engine::init_process().

COIRESULT(∗ COI::EventWait)(uint16_t, const COIEVENT ∗, int32_t, uint8_t, uint32_t ∗, uint32_t ∗)

Definition at line 70 of file coi_client.cpp.

Referenced by __offload_myoFini(), __offload_myoInit_once(), init(), Engine::init_device(), Engine::init_ptr_data(), OffloadDescriptor::is_signaled(), and OffloadDescriptor::offload_finish().

bool COI::is_available

Definition at line 21 of file coi_client.cpp.

Referenced by __offload_fini_library(), __offload_init_library(), fini(), and init().

void∗ COI::lib_handle `[static]`

Definition at line 22 of file coi_client.cpp.

Referenced by fini(), and init().

uint64_t(∗ COI::PerfGetCycleFrequency)(void)

Definition at line 73 of file coi_client.cpp.

Referenced by __offload_init_library_once(), and init().

COIRESULT(∗ COI::PipelineCreate)(COIPROCESS, COI_CPU_MASK, uint32_t, COIPIPELINE ∗)

Definition at line 43 of file coi_client.cpp.
Referenced by Engine::get_pipeline(), and init().

COIRESULT(∗ COI::PipelineDestroy)(COIPIPELINE)

Definition at line 44 of file coi_client.cpp.
Referenced by init(), and Thread::∼Thread().

COIRESULT(∗ COI::PipelineRunFunction)(COIPIPELINE, COIFUNCTION, uint32_t, const COIBUFFER ∗, const COI_ACCESS_FLAGS ∗, uint32_t, const COIEVENT ∗, const void ∗, uint16_t, void ∗, uint16_t, COIEVENT ∗)

Definition at line 45 of file coi_client.cpp.
Referenced by Engine::compute(), init(), Engine::init_device(), and Engine::init_ptr_data().

COIRESULT(∗ COI::ProcessCreateFromMemory)(COIENGINE, const char ∗, const void ∗, uint64_t, int, const char ∗∗, uint8_t, const char ∗∗, uint8_t, const char ∗, uint64_t, const char ∗, const char ∗, uint64_t, COIPROCESS ∗)

Definition at line 28 of file coi_client.cpp.
Referenced by init(), and Engine::init_process().

COIRESULT(∗ COI::ProcessDestroy)(COIPROCESS, int32_t, uint8_t, int8_t ∗, uint32_t ∗)

Definition at line 33 of file coi_client.cpp.
Referenced by Engine::fini_process(), and init().

COIRESULT(∗ COI::ProcessGetFunctionHandles)(COIPROCESS, uint32_t, const char ∗∗, COIFUNCTION ∗)

Definition at line 34 of file coi_client.cpp.
Referenced by init(), and Engine::init_process().

COIRESULT(∗ COI::ProcessLoadLibraryFromMemory)(COIPROCESS, const void ∗, uint64_t, const char ∗, const char ∗, const char ∗, uint64_t, uint32_t, COILIBRARY ∗)

Definition at line 36 of file coi_client.cpp.
Referenced by init(), and Engine::load_libraries().

COIRESULT(∗ COI::ProcessRegisterLibraries)(uint32_t, const void ∗∗, const uint64_t ∗, const char ∗∗, const uint64_t ∗)

Definition at line 40 of file coi_client.cpp.
Referenced by __offload_init_library(), and init().

5.2 ORSL Namespace Reference

Functions

- void init ()
- bool reserve (int device)
- bool try_reserve (int device)
- void release (int device)

Variables

- static bool is_enabled = false
- static const ORSLTag my_tag = ”Offload”

5.2.1 Function Documentation

void ORSL::init ( )

Definition at line 21 of file offload_orsl.cpp.
Referenced by __offload_init_library_once().

void ORSL::release (int *device*)

Definition at line 71 of file offload_orsl.cpp.
Referenced by __intel_cilk_for_32_offload(), __intel_cilk_for_64_offload(), __offload_myoiRemoteIThunkCall(), and OffloadDescriptor::cleanup().

bool ORSL::reserve (int *device*)

Definition at line 43 of file offload_orsl.cpp.
Referenced by __offload_myoIsAvailable(), OFFLOAD_TARGET_ACQUIRE(), and OFFLOAD_TARGET_ACQUIRE1().

bool ORSL::try_reserve (int *device*)

Definition at line 57 of file offload_orsl.cpp.
Referenced by OFFLOAD_TARGET_ACQUIRE().

5.2.2 Variable Documentation

bool ORSL::is_enabled = false `[static]`

Definition at line 18 of file offload_orsl.cpp.
Referenced by init(), release(), reserve(), and try_reserve().

const ORSLTag ORSL::my_tag = "Offload" `[static]`

Definition at line 19 of file offload_orsl.cpp.
Referenced by release(), reserve(), and try_reserve().

Chapter 6

Class Documentation

6.1 _Offload_status Struct Reference

`#include <offload.h>`

Public Attributes

- _Offload_result result
- int device_number
- size_t data_sent
- size_t data_received

6.1.1 Detailed Description

Definition at line 62 of file offload.h.

6.1.2 Member Data Documentation

size_t _Offload_status::data_received

Definition at line 66 of file offload.h.
Referenced by OFFLOAD_TARGET_ACQUIRE(), OffloadDescriptor::receive_pointer_data(), and OffloadDescriptor::scatter_copyout_data().

size_t _Offload_status::data_sent

Definition at line 65 of file offload.h.
Referenced by OffloadDescriptor::gather_copyin_data(), OFFLOAD_TARGET_ACQUIRE(), and OffloadDescriptor::send_pointer_data().

int _Offload_status::device_number

Definition at line 64 of file offload.h.
Referenced by OffloadDescriptor::offload(), and OFFLOAD_TARGET_ACQUIRE().

_Offload_result _Offload_status::result

Definition at line 63 of file offload.h.
Referenced by OffloadDescriptor::alloc_ptr_data(), OffloadDescriptor::compute(), OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::init_mic_address(), OffloadDescriptor::init_static_ptr_data(), OffloadDescriptor::nullify_target_stack(), OffloadDescriptor::offload(), OffloadDescriptor::offload_finish(), OffloadDescriptor::offload_stack_memory_manager(), OFFLOAD_TARGET_ACQUIRE(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::scatter_copyout_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_misc_data().

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

- offload.h

6.2 arr_desc Struct Reference

`#include <cean_util.h>`

Public Attributes

- int64_t base
- int64_t rank
- dim_desc dim [1]

6.2.1 Detailed Description

Definition at line 25 of file cean_util.h.

6.2.2 Member Data Documentation

int64_t arr_desc::base

Definition at line 26 of file cean_util.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), generate_mem_ranges(), init_read_ranges_arr_desc(), make_arr_desc(), offload_get_src_base(), and OffloadDescriptor::setup_descriptors().

dim_desc arr_desc::dim[1]

Definition at line 28 of file cean_util.h.

Referenced by __arr_data_offset_and_length(), OffloadDescriptor::gen_var_descs_for_pointer_array(), generate_mem_ranges(), init_read_ranges_arr_desc(), is_arr_desc_contiguous(), and make_arr_desc().

int64_t arr_desc::rank

Definition at line 27 of file cean_util.h.

Referenced by __arr_data_offset_and_length(), OffloadDescriptor::gen_var_descs_for_pointer_array(), generate_mem_ranges(), init_read_ranges_arr_desc(), is_arr_desc_contiguous(), and make_arr_desc().

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

- cean_util.h

6.3 ArrDesc Struct Reference

`#include <dv_util.h>`

Public Attributes

- dv_size Base
- dv_size Len
- dv_size Offset
- dv_size Flags
- dv_size Rank
- dv_size Reserved
- DimDesc Dim [ArrDescMaxArrayRank]

6.3.1 Detailed Description

Definition at line 34 of file dv_util.h.

6.3.2 Member Data Documentation

dv_size ArrDesc::Base

Definition at line 35 of file dv_util.h.
Referenced by OffloadDescriptor::gather_copyout_data(), init_read_ranges_dv(), offload_get_src_base(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::setup_descriptors().

DimDesc ArrDesc::Dim[ArrDescMaxArrayRank]

Definition at line 42 of file dv_util.h.
Referenced by __dv_data_length(), __dv_is_contiguous(), and init_read_ranges_dv().

dv_size ArrDesc::Flags

Definition at line 39 of file dv_util.h.
Referenced by __dv_is_allocated(), and __dv_is_contiguous().

dv_size ArrDesc::Len

Definition at line 36 of file dv_util.h.
Referenced by __dv_data_length(), __dv_is_contiguous(), and init_read_ranges_dv().

dv_size ArrDesc::Offset

Definition at line 38 of file dv_util.h.

dv_size ArrDesc::Rank

Definition at line 40 of file dv_util.h.
Referenced by __dv_data_length(), __dv_is_contiguous(), and init_read_ranges_dv().

dv_size ArrDesc::Reserved

Definition at line 41 of file dv_util.h.
The documentation for this struct was generated from the following file:

- dv_util.h

6.4 AutoData Class Reference

`#include <offload_engine.h>`

Public Member Functions

- AutoData (const void *addr, uint64_t len)
- bool operator< (const AutoData &o) const
- long add_reference ()
- long remove_reference ()
- long get_reference () const

Public Attributes

- const MemRange cpu_addr

Private Attributes

- long ref_count

6.4.1 Detailed Description

Definition at line 141 of file offload_engine.h.

6.4.2 Constructor & Destructor Documentation

AutoData::AutoData (const void * *addr,* uint64_t *len*) [inline]

Definition at line 143 of file offload_engine.h.

6.4.3 Member Function Documentation

long AutoData::add_reference ( ) [inline]

Definition at line 154 of file offload_engine.h.
Referenced by OffloadDescriptor::setup_descriptors().

long AutoData::get_reference ( ) const [inline]

Definition at line 170 of file offload_engine.h.
Referenced by OffloadDescriptor::setup_descriptors().

bool AutoData::operator< (const AutoData & *o*) const [inline]

Definition at line 147 of file offload_engine.h.

long AutoData::remove_reference ( ) [inline]

Definition at line 162 of file offload_engine.h.
Referenced by OffloadDescriptor::receive_pointer_data().

6.4.4 Member Data Documentation

const MemRange AutoData::cpu_addr

Definition at line 176 of file offload_engine.h.
Referenced by operator<(), and OffloadDescriptor::receive_pointer_data().

long AutoData::ref_count [private]

Definition at line 180 of file offload_engine.h.
Referenced by add_reference(), get_reference(), and remove_reference().
The documentation for this class was generated from the following file:

- offload_engine.h

6.5 VarList::BufEntry Struct Reference

`#include <offload_table.h>`

Public Attributes

- intptr_t name
- intptr_t addr

6.5.1 Detailed Description

Definition at line 231 of file offload_table.h.

6.5.2 Member Data Documentation

intptr_t VarList::BufEntry::addr

Definition at line 233 of file offload_table.h.
Referenced by Engine::init_ptr_data(), and VarList::table_copy().

intptr_t VarList::BufEntry::name

Definition at line 232 of file offload_table.h.
Referenced by Engine::init_ptr_data(), VarList::table_copy(), VarList::table_patch_names(), and target_entry_cmp().
The documentation for this struct was generated from the following file:

- offload_table.h

6.6 MicEnvVar::CardEnvVars Struct Reference

`#include <offload_env.h>`

Public Member Functions

- CardEnvVars ()
- CardEnvVars (int num)
- ~CardEnvVars ()
- void add_new_env_var (int number, char *env_var, int length, char *env_var_value)
- VarValue * find_var (char *env_var_name, int env_var_name_length)

Public Attributes

- int card_number
- std::list< struct VarValue * > env_vars

6.6.1 Detailed Description

Definition at line 66 of file offload_env.h.

6.6.2 Constructor & Destructor Documentation

MicEnvVar::CardEnvVars::CardEnvVars ( ) `[inline]`

Definition at line 72 of file offload_env.h.

MicEnvVar::CardEnvVars::CardEnvVars (int *num*) `[inline]`

Definition at line 73 of file offload_env.h.

MicEnvVar::CardEnvVars::~CardEnvVars ( )

Definition at line 35 of file offload_env.cpp.

6.6.3 Member Function Documentation

void MicEnvVar::CardEnvVars::add_new_env_var (int *number,* char * *env_var,* int *length,* char * *env_var_value*)

MicEnvVar::VarValue * MicEnvVar::CardEnvVars::find_var (char * *env_var_name,* int *env_var_name_length*)

Definition at line 64 of file offload_env.cpp.
Referenced by MicEnvVar::add_env_var(), and MicEnvVar::create_environ_for_card().

6.6.4 Member Data Documentation

int MicEnvVar::CardEnvVars::card_number

Definition at line 69 of file offload_env.h.
Referenced by CardEnvVars(), and MicEnvVar::get_card().

std::list<struct VarValue*> MicEnvVar::CardEnvVars::env_vars

Definition at line 70 of file offload_env.h.
Referenced by MicEnvVar::add_env_var(), and MicEnvVar::create_environ_for_card().
The documentation for this struct was generated from the following files:

- offload_env.h
- offload_env.cpp

6.7 CeanReadDim Struct Reference

`#include <cean_util.h>`

Public Attributes

- int64_t count
- int64_t size

6.7.1 Detailed Description

Definition at line 31 of file cean_util.h.

6.7.2 Member Data Documentation

int64_t CeanReadDim::count

Definition at line 32 of file cean_util.h.
Referenced by get_next_range(), init_read_ranges_arr_desc(), and init_read_ranges_dv().

int64_t CeanReadDim::size

Definition at line 33 of file cean_util.h.
Referenced by get_next_range(), init_read_ranges_arr_desc(), and init_read_ranges_dv().
The documentation for this struct was generated from the following file:

- cean_util.h

6.8 CeanReadRanges Struct Reference

`#include <cean_util.h>`

Public Attributes

- void * ptr
- int64_t current_number
- int64_t range_max_number
- int64_t range_size
- int last_noncont_ind
- int64_t init_offset
- CeanReadDim Dim [1]

6.8.1 Detailed Description

Definition at line 37 of file cean_util.h.

6.8.2 Member Data Documentation

int64_t CeanReadRanges::current_number

Definition at line 39 of file cean_util.h.
Referenced by get_next_range(), and init_read_ranges_arr_desc().

CeanReadDim CeanReadRanges::Dim[1]

Definition at line 44 of file cean_util.h.
Referenced by get_next_range(), init_read_ranges_arr_desc(), and init_read_ranges_dv().

int64_t CeanReadRanges::init_offset

Definition at line 43 of file cean_util.h.
Referenced by get_next_range().

int CeanReadRanges::last_noncont_ind

Definition at line 42 of file cean_util.h.
Referenced by get_next_range(), and init_read_ranges_arr_desc().

void* CeanReadRanges::ptr

Definition at line 38 of file cean_util.h.

int64_t CeanReadRanges::range_max_number

Definition at line 40 of file cean_util.h.
Referenced by cean_get_transf_size(), get_arr_desc_numbers(), get_next_range(), and init_read_ranges_arr_desc().

int64_t CeanReadRanges::range_size

Definition at line 41 of file cean_util.h.
Referenced by cean_get_transf_size(), cean_ranges_match(), get_arr_desc_numbers(), init_read_ranges_arr_desc(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), and OffloadDescriptor::send_noncontiguous_pointer_data().
The documentation for this struct was generated from the following file:

- cean_util.h

6.9 dim_desc Struct Reference

`#include <cean_util.h>`

Public Attributes

- int64_t size
- int64_t lindex
- int64_t lower
- int64_t upper
- int64_t stride

6.9.1 Detailed Description

Definition at line 17 of file cean_util.h.

6.9.2 Member Data Documentation

int64_t dim_desc::lindex

Definition at line 19 of file cean_util.h.

Referenced by __arr_data_offset_and_length(), generate_mem_ranges_one_rank(), init_read_ranges_arr_desc(), and make_arr_desc().

int64_t dim_desc::lower

Definition at line 20 of file cean_util.h.

Referenced by __arr_data_offset_and_length(), generate_mem_ranges_one_rank(), init_read_ranges_arr_desc(), is_arr_desc_contiguous(), and make_arr_desc().

int64_t dim_desc::size

Definition at line 18 of file cean_util.h.

Referenced by __arr_data_offset_and_length(), OffloadDescriptor::gen_var_descs_for_pointer_array(), generate_mem_ranges(), generate_mem_ranges_one_rank(), init_read_ranges_arr_desc(), is_arr_desc_contiguous(), and make_arr_desc().

int64_t dim_desc::stride

Definition at line 22 of file cean_util.h.

Referenced by generate_mem_ranges_one_rank(), init_read_ranges_arr_desc(), is_arr_desc_contiguous(), and make_arr_desc().

int64_t dim_desc::upper

Definition at line 21 of file cean_util.h.

Referenced by __arr_data_offset_and_length(), generate_mem_ranges_one_rank(), init_read_ranges_arr_desc(), is_arr_desc_contiguous(), and make_arr_desc().

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

- cean_util.h

6.10 DimDesc Struct Reference

`#include <dv_util.h>`

Public Attributes

- dv_size Extent
- dv_size Mult
- dv_size LowerBound

6.10.1 Detailed Description

Definition at line 26 of file dv_util.h.

6.10.2 Member Data Documentation

dv_size DimDesc::Extent

Definition at line 27 of file dv_util.h.

Referenced by __dv_data_length(), __dv_is_contiguous(), and init_read_ranges_dv().

dv_size DimDesc::LowerBound

Definition at line 31 of file dv_util.h.

dv_size DimDesc::Mult

Definition at line 28 of file dv_util.h.

Referenced by __dv_data_length(), __dv_is_contiguous(), and init_read_ranges_dv().

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

- dv_util.h

6.11 Engine Struct Reference

`#include <offload_engine.h>`

Public Member Functions

- int get_logical_index () const
- int get_physical_index () const
- const COIPROCESS & get_process () const
- void init (void)
- void add_lib (const TargetImage &lib)
- COIRESULT compute (const std::list< COIBUFFER > &buffers, const void *data, uint16_t data_size, void *ret, uint16_t ret_size, uint32_t num_deps, const COIEVENT *deps, COIEVENT *event)
- PtrData * find_ptr_data (const void *ptr)
- PtrData * insert_ptr_data (const void *ptr, uint64_t len, bool &is_new)
- void remove_ptr_data (const void *ptr)
- AutoData * find_auto_data (const void *ptr)
- AutoData * insert_auto_data (const void *ptr, uint64_t len)
- void remove_auto_data (const void *ptr)
- void add_signal (const void *signal, OffloadDescriptor *desc)
- OffloadDescriptor * find_signal (const void *signal, bool remove)
- void fini_process (bool verbose)

Public Attributes

- PersistDataList m_persist_list

Private Types

- enum { c_func_compute = 0, c_func_init, c_func_var_table_size, c_func_var_table_copy, c_funcs_total }
- typedef std::set< PtrData > PtrSet
- typedef std::map< const void *, OffloadDescriptor * > SignalMap

Private Member Functions

- Engine ()
- ~Engine ()
- void set_indexes (int logical_index, int physical_index)
- void init_process ()
- void load_libraries (void)
- void init_ptr_data (void)
- pid_t init_device (void)
- COIPIPELINE get_pipeline (void)
- AutoSet & get_auto_vars (void)

Static Private Member Functions

- static void destroy_thread_data (void *data)

Private Attributes

- int m_index
- int m_physical_index
- long m_proc_number
- COIPROCESS m_process
- bool m_ready
- mutex_t m_lock
- TargetImageList m_images
- PtrSet m_ptr_set
- mutex_t m_ptr_lock
- SignalMap m_signal_map
- mutex_t m_signal_lock
- COIFUNCTION m_funcs [c_funcs_total]

Static Private Attributes

- static const char * m_func_names [c_funcs_total]
- static const int c_signal_max = 32
- static const char * c_signal_names [c_signal_max]

Friends

- void __offload_init_library_once (void)
- void __offload_fini_library (void)

6.11.1 Detailed Description

Definition at line 230 of file offload_engine.h.

6.11.2 Member Typedef Documentation

typedef std::set<PtrData> Engine::PtrSet `[private]`

Definition at line 431 of file offload_engine.h.

typedef std::map<const void*, OffloadDescriptor*> Engine::SignalMap `[private]`

Definition at line 432 of file offload_engine.h.

6.11.3 Member Enumeration Documentation

anonymous enum `[private]`

Enumerator

c_func_compute

c_func_init

c_func_var_table_size

c_func_var_table_copy

c_funcs_total

Definition at line 461 of file offload_engine.h.

6.11.4 Constructor & Destructor Documentation

Engine::Engine ( ) [inline], [private]

Definition at line 395 of file offload_engine.h.

Engine::~Engine ( ) [inline], [private]

Definition at line 399 of file offload_engine.h.

6.11.5 Member Function Documentation

void Engine::add_lib (const TargetImage & *lib*) [inline]

Definition at line 262 of file offload_engine.h.
Referenced by __offload_init_library().

void Engine::add_signal (const void * *signal,* OffloadDescriptor * *desc*) [inline]

Definition at line 364 of file offload_engine.h.
Referenced by OffloadDescriptor::offload().

COIRESULT Engine::compute (const std::list< COIBUFFER > & *buffers,* const void * *data,* uint16_t *data_size,* void * *ret,* uint16_t *ret_size,* uint32_t *num_deps,* const COIEVENT * *deps,* COIEVENT * *event*)

Definition at line 361 of file offload_engine.cpp.
Referenced by OffloadDescriptor::compute().

void Engine::destroy_thread_data (void * *data*) [static], [private]

Definition at line 528 of file offload_engine.cpp.
Referenced by __offload_init_library_once().

AutoData* Engine::find_auto_data (const void * *ptr*) [inline]

Definition at line 341 of file offload_engine.h.
Referenced by OffloadDescriptor::setup_descriptors().

PtrData* Engine::find_ptr_data (const void * *ptr*) [inline]

Definition at line 305 of file offload_engine.h.
Referenced by OffloadDescriptor::find_ptr_data().

OffloadDescriptor* Engine::find_signal (const void * *signal,* bool *remove*) [inline]

Definition at line 370 of file offload_engine.h.
Referenced by _Offload_signaled(), and OffloadDescriptor::wait_dependencies().

void Engine::fini_process (bool *verbose*)

Definition at line 164 of file offload_engine.cpp.
Referenced by OffloadDescriptor::report_coi_error(), and ~Engine().

AutoSet & Engine::get_auto_vars (void) [private]

Definition at line 517 of file offload_engine.cpp.
Referenced by find_auto_data(), insert_auto_data(), and remove_auto_data().

int Engine::get_logical_index ( ) const [inline]

Definition at line 246 of file offload_engine.h.

Referenced by OffloadDescriptor::cleanup(), OffloadDescriptor::offload(), OffloadDescriptor::report_coi_error(), and OffloadDescriptor::wait_dependencies().

int Engine::get_physical_index ( ) const [inline]

Definition at line 250 of file offload_engine.h.

Referenced by __offload_myoInit_once(), ORSL::release(), ORSL::reserve(), and ORSL::try_reserve().

COIPIPELINE Engine::get_pipeline (void) [private]

Definition at line 485 of file offload_engine.cpp.

Referenced by compute(), init_device(), and init_ptr_data().

const COIPROCESS& Engine::get_process ( ) const [inline]

Definition at line 254 of file offload_engine.h.

Referenced by OffloadDescriptor::alloc_ptr_data(), OffloadDescriptor::init_static_ptr_data(), OffloadDescriptor::offload_stack_memory_manager(), and OffloadDescriptor::setup_misc_data().

void Engine::init (void)

Definition at line 70 of file offload_engine.cpp.

Referenced by __offload_myoInit_once(), __offload_myoIsAvailable(), __offload_register_image(), OFFLOAD_TARGET_ACQUIRE(), and OFFLOAD_TARGET_ACQUIRE1().

pid_t Engine::init_device (void) [private]

Definition at line 408 of file offload_engine.cpp.

Referenced by init_process().

void Engine::init_process (void) [private]

Definition at line 93 of file offload_engine.cpp.

Referenced by init().

void Engine::init_ptr_data (void) [private]

Definition at line 250 of file offload_engine.cpp.

Referenced by init().

AutoData* Engine::insert_auto_data (const void * *ptr,* uint64_t *len*) [inline]

Definition at line 350 of file offload_engine.h.

Referenced by OffloadDescriptor::setup_descriptors().

PtrData* Engine::insert_ptr_data (const void * *ptr,* uint64_t *len,* bool & *is_new*) [inline]

Definition at line 315 of file offload_engine.h.

Referenced by OffloadDescriptor::alloc_ptr_data().

void Engine::load_libraries (void) [private]

Definition at line 204 of file offload_engine.cpp.

Referenced by init().

void Engine::remove_auto_data (const void * *ptr*) [inline]

Definition at line 357 of file offload_engine.h.
Referenced by OffloadDescriptor::receive_pointer_data().

void Engine::remove_ptr_data (const void * *ptr*) [inline]

Definition at line 332 of file offload_engine.h.
Referenced by OffloadDescriptor::receive_pointer_data().

void Engine::set_indexes (int *logical_index,* int *physical_index*) [inline], [private]

Definition at line 406 of file offload_engine.h.
Referenced by __offload_init_library_once().

6.11.6 Friends And Related Function Documentation

void __offload_fini_library (void) [friend]

Definition at line 3873 of file offload_host.cpp.

void __offload_init_library_once (void) [friend]

Definition at line 3901 of file offload_host.cpp.

6.11.7 Member Data Documentation

const int Engine::c_signal_max = 32 [static], [private]

Definition at line 478 of file offload_engine.h.
Referenced by fini_process().

const char * Engine::c_signal_names [static], [private]

Definition at line 479 of file offload_engine.h.
Referenced by fini_process().

const char * Engine::m_func_names [static], [private]

Initial value:

```
=
{
    "server_compute",



    "server_init",
    "server_var_table_size",
    "server_var_table_copy"
}
```

Definition at line 472 of file offload_engine.h.
Referenced by init_process().

COIFUNCTION Engine::m_funcs[c_funcs_total] [private]

Definition at line 475 of file offload_engine.h.
Referenced by compute(), init_device(), init_process(), and init_ptr_data().

TargetImageList Engine::m_images [private]

Definition at line 450 of file offload_engine.h.
Referenced by add_lib(), and load_libraries().

int Engine::m_index [private]

Definition at line 435 of file offload_engine.h.
Referenced by fini_process(), get_logical_index(), get_pipeline(), init_device(), init_process(), init_ptr_data(), load_libraries(), and set_indexes().

mutex_t Engine::m_lock [private]

Definition at line 447 of file offload_engine.h.
Referenced by add_lib(), and init().

PersistDataList Engine::m_persist_list

Definition at line 392 of file offload_engine.h.
Referenced by OffloadDescriptor::offload_stack_memory_manager(), and OffloadDescriptor::setup_descriptors().

int Engine::m_physical_index [private]

Definition at line 436 of file offload_engine.h.
Referenced by get_physical_index(), init_device(), init_process(), and set_indexes().

long Engine::m_proc_number [private]

Definition at line 439 of file offload_engine.h.
Referenced by get_auto_vars(), and get_pipeline().

COIPROCESS Engine::m_process [private]

Definition at line 442 of file offload_engine.h.
Referenced by fini_process(), get_pipeline(), get_process(), init(), init_process(), init_ptr_data(), load_libraries(), and ~Engine().

mutex_t Engine::m_ptr_lock [private]

Definition at line 454 of file offload_engine.h.
Referenced by find_ptr_data(), insert_ptr_data(), and remove_ptr_data().

PtrSet Engine::m_ptr_set [private]

Definition at line 453 of file offload_engine.h.
Referenced by find_ptr_data(), init_ptr_data(), insert_ptr_data(), and remove_ptr_data().

bool Engine::m_ready [private]

Definition at line 446 of file offload_engine.h.
Referenced by add_lib(), and init().

mutex_t Engine::m_signal_lock [private]

Definition at line 458 of file offload_engine.h.
Referenced by add_signal(), and find_signal().

SignalMap Engine::m_signal_map [private]

Definition at line 457 of file offload_engine.h.
Referenced by add_signal(), and find_signal().
The documentation for this struct was generated from the following files:

- offload_engine.h
- offload_engine.cpp

6.12 FuncTable::Entry Struct Reference

Function table entry.

`#include <offload_table.h>`

Public Attributes

- const char * name

  Name of the function.

- void * func

  Address of the function.

6.12.1 Detailed Description

Function table entry.

This table contains functions created from offload regions.

Each entry consists of a pointer to the function's "key" and the function address.

Each shared library or executable may contain one such table.

The end of the table is marked with an entry whose name field has value -1.

Definition at line 79 of file offload_table.h.

6.12.2 Member Data Documentation

void* FuncTable::Entry::func

Address of the function.

Definition at line 81 of file offload_table.h.

const char* FuncTable::Entry::name

Name of the function.

Definition at line 80 of file offload_table.h.

Referenced by FuncList::dump(), FuncList::find_addr(), FuncList::find_name(), and FuncList::max_name_length().

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

- offload_table.h

6.13 VarTable::Entry Struct Reference

Variable table entry.

`#include <offload_table.h>`

Public Attributes

- const char * name

  Name of the variable.

- void * addr

  Address of the variable.

6.13.1 Detailed Description

Variable table entry.

This table contains statically allocated variables marked with __declspec(target(mic) or #pragma omp declare target.

Each entry consists of a pointer to the variable's "key", the variable address and its size in bytes.

Because memory allocation is done from the host, the MIC table does not need the size of the variable.

Padding to make the table entry size a power of 2 is necessary to avoid "holes" between table contributions from different object files on Windows when debug information is specified with /Zi.

Definition at line 136 of file offload_table.h.

6.13.2 Member Data Documentation

void* VarTable::Entry::addr

Address of the variable.

Definition at line 138 of file offload_table.h.

const char* VarTable::Entry::name

Name of the variable.

Definition at line 137 of file offload_table.h.

Referenced by VarList::dump(), host_entry_cmp(), VarList::Iterator::new_node(), VarList::Iterator::operator++(), VarList::table_copy(), and VarList::table_size().

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

- offload_table.h

6.14 FptrTableEntry Struct Reference

`#include <offload_myo_host.h>`

Public Attributes

- const char * funcName

  Function Name.
- void * funcAddr

  Function Address.
- void * localThunkAddr

  Local Thunk Address.

6.14.1 Detailed Description

Definition at line 21 of file offload_myo_host.h.

6.14.2 Member Data Documentation

void* FptrTableEntry::funcAddr

Function Address.

Definition at line 25 of file offload_myo_host.h.

Referenced by __offload_myo_fptr_table_register().

const char* FptrTableEntry::funcName

Function Name.

Definition at line 23 of file offload_myo_host.h.

Referenced by __offload_myo_fptr_table_register(), __offload_myoRegisterTables(), and fptr_table_entries().

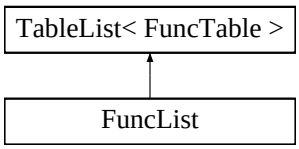
void* FptrTableEntry::localThunkAddr

Local Thunk Address.
Definition at line 27 of file offload_myo_host.h.
Referenced by __offload_myo_fptr_table_register().
The documentation for this struct was generated from the following file:

- offload_myo_host.h

6.15 FuncList Class Reference

`#include <offload_table.h>`
Inheritance diagram for FuncList:



Public Member Functions

- FuncList (Node *node=0)
- void add_table (Node *node)
- const void * find_addr (const char *name)
- const char * find_name (const void *addr)
- int64_t max_name_length (void)
- void dump (void)

Private Attributes

- int64_t m_max_name_len

Additional Inherited Members

6.15.1 Detailed Description

Definition at line 92 of file offload_table.h.

6.15.2 Constructor & Destructor Documentation

FuncList::FuncList (Node * *node = 0*) [inline], [explicit]

Definition at line 94 of file offload_table.h.

6.15.3 Member Function Documentation

void FuncList::add_table (Node * *node*) [inline]

Definition at line 99 of file offload_table.h.
Referenced by __offload_register_tables().

void FuncList::dump (void)

Definition at line 245 of file offload_table.cpp.
Referenced by OffloadDescriptor::offload(), Marshaller::receive_func_ptr(), and Marshaller::send_func_ptr().

const void * FuncList::find_addr (const char * *name*)

Definition at line 167 of file offload_table.cpp.
Referenced by OffloadDescriptor::offload(), and Marshaller::receive_func_ptr().

const char * FuncList::find_name (const void * *addr*)

Definition at line 189 of file offload_table.cpp.
Referenced by Marshaller::send_func_ptr().

int64_t FuncList::max_name_length (void)

Definition at line 211 of file offload_table.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

6.15.4 Member Data Documentation

int64_t FuncList::m_max_name_len [private]

Definition at line 121 of file offload_table.h.
Referenced by add_table(), and max_name_length().
The documentation for this class was generated from the following files:

- offload_table.h
- offload_table.cpp

6.16 FuncTable Struct Reference

`#include <offload_table.h>`

Classes

- struct Entry

  Function table entry.

Public Attributes

- const Entry * entries
- int64_t max_name_len

6.16.1 Detailed Description

Definition at line 71 of file offload_table.h.

6.16.2 Member Data Documentation

const Entry* FuncTable::entries

Definition at line 85 of file offload_table.h.

int64_t FuncTable::max_name_len

Definition at line 88 of file offload_table.h.
The documentation for this struct was generated from the following file:

- offload_table.h

6.17 FunctionDescriptor Struct Reference

`#include <offload_common.h>`

Public Attributes

- long long in_datalen
- long long out_datalen
- uint8_t console_enabled
- uint8_t timer_enabled
- int offload_report_level
- int offload_number
- int vars_num
- int data_offset
- char data []

6.17.1 Detailed Description

Definition at line 409 of file offload_common.h.

6.17.2 Member Data Documentation

uint8_t FunctionDescriptor::console_enabled

Definition at line 420 of file offload_common.h.
Referenced by OffloadDescriptor::offload(), and OffloadDescriptor::setup_misc_data().

char FunctionDescriptor::data[]

Definition at line 437 of file offload_common.h.
Referenced by OffloadDescriptor::offload(), and OffloadDescriptor::setup_misc_data().

int FunctionDescriptor::data_offset

Definition at line 434 of file offload_common.h.
Referenced by OffloadDescriptor::compute(), OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::offload(), OffloadDescriptor::scatter_copyout_data(), and OffloadDescriptor::setup_misc_data().

long long FunctionDescriptor::in_datalen

Definition at line 412 of file offload_common.h.
Referenced by OffloadDescriptor::offload(), and OffloadDescriptor::setup_misc_data().

int FunctionDescriptor::offload_number

Definition at line 427 of file offload_common.h.
Referenced by OffloadDescriptor::offload(), and OffloadDescriptor::setup_misc_data().

int FunctionDescriptor::offload_report_level

Definition at line 426 of file offload_common.h.
Referenced by OffloadDescriptor::offload(), and OffloadDescriptor::setup_misc_data().

long long FunctionDescriptor::out_datalen

Definition at line 415 of file offload_common.h.
Referenced by OffloadDescriptor::offload(), and OffloadDescriptor::setup_misc_data().

uint8_t FunctionDescriptor::timer_enabled

Definition at line 424 of file offload_common.h.
Referenced by OffloadDescriptor::offload(), and OffloadDescriptor::setup_misc_data().

int FunctionDescriptor::vars_num

Definition at line 430 of file offload_common.h.
Referenced by OffloadDescriptor::offload(), and OffloadDescriptor::setup_misc_data().
The documentation for this struct was generated from the following file:

- offload_common.h

6.18 Image Struct Reference

The target image is packed as follows.
`#include <offload_host.h>`

Public Attributes

- int64_t size

  Size in bytes of the target binary name and contents.
- char data []

  The name and contents of the target image.

6.18.1 Detailed Description

The target image is packed as follows.

1. 8 bytes containing the size of the target binary
2. a null-terminated string which is the binary name
3. <size> number of bytes that are the contents of the image

The address of symbol __offload_target_image is the address of this structure.
Definition at line 38 of file offload_host.h.

6.18.2 Member Data Documentation

char Image::data[]

The name and contents of the target image.
Definition at line 40 of file offload_host.h.
Referenced by __offload_register_image(), and __offload_unregister_image().

int64_t Image::size

Size in bytes of the target binary name and contents.
Definition at line 39 of file offload_host.h.
Referenced by __offload_register_image(), and __offload_unregister_image().
The documentation for this struct was generated from the following file:

- offload_host.h

6.19 InitTableEntry Struct Reference

`#include <offload_myo_host.h>`

Public Attributes

- void(∗ func)(void)

6.19.1 Detailed Description

Definition at line 34 of file offload_myo_host.h.

6.19.2 Member Data Documentation

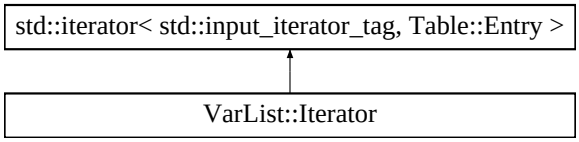
void(∗ InitTableEntry::func)(void)

Definition at line 40 of file offload_myo_host.h.
Referenced by __offload_myo_shared_init_table_register().
The documentation for this struct was generated from the following file:

- offload_myo_host.h

6.20 VarList::Iterator Class Reference

`#include <offload_table.h>`
Inheritance diagram for VarList::Iterator:



Public Member Functions

- Iterator ()
- Iterator (Node ∗node)
- Iterator & operator++ ()
- bool operator== (const Iterator &other) const
- bool operator!= (const Iterator &other) const
- const Table::Entry ∗ operator∗ () const

Private Member Functions

- void new_node (Node ∗node)

Private Attributes

- Node ∗ m_node
- const Table::Entry ∗ m_entry

6.20.1 Detailed Description

Definition at line 165 of file offload_table.h.

6.20.2 Constructor & Destructor Documentation

VarList::Iterator::Iterator ( ) [inline]

Definition at line 168 of file offload_table.h.

VarList::Iterator::Iterator (Node * *node*) [inline], [explicit]

Definition at line 170 of file offload_table.h.

6.20.3 Member Function Documentation

void VarList::Iterator::new_node (Node * *node*) [inline], [private]

Definition at line 200 of file offload_table.h.
Referenced by Iterator(), and operator++().

bool VarList::Iterator::operator!= (const Iterator & *other*) const [inline]

Definition at line 191 of file offload_table.h.

const Table::Entry* VarList::Iterator::operator* ( ) const [inline]

Definition at line 195 of file offload_table.h.

Iterator& VarList::Iterator::operator++ ( ) [inline]

Definition at line 174 of file offload_table.h.

bool VarList::Iterator::operator== (const Iterator & *other*) const [inline]

Definition at line 187 of file offload_table.h.

6.20.4 Member Data Documentation

const Table::Entry* VarList::Iterator::m_entry [private]

Definition at line 218 of file offload_table.h.
Referenced by new_node(), operator!=(), operator*(), operator++(), and operator==().

Node* VarList::Iterator::m_node [private]

Definition at line 217 of file offload_table.h.
Referenced by new_node(), and operator++().
The documentation for this class was generated from the following file:

- offload_table.h

6.21 kmp_affinity_mask_target_t Struct Reference

`#include <offload.h>`

Public Attributes

- kmp_affinity_mask_t mask

6.21.1 Detailed Description

Definition at line 336 of file offload.h.

6.21.2 Member Data Documentation

kmp_affinity_mask_t kmp_affinity_mask_target_t::mask

Definition at line 337 of file offload.h.

Referenced by kmp_create_affinity_mask_lrb(), kmp_destroy_affinity_mask_lrb(), kmp_get_affinity_lrb(), kmp_get_affinity_mask_proc_lrb(), kmp_set_affinity_lrb(), kmp_set_affinity_mask_proc_lrb(), and kmp_unset_affinity_mask_proc_lrb().

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

- offload.h

6.22 mic_lib::kmp_create_affinity_mask_target Interface Reference

Public Member Functions

- subroutine kmp_create_affinity_mask_target (target_type, target_number, mask)

6.22.1 Detailed Description

Definition at line 361 of file mic_lib.f90.

6.22.2 Constructor & Destructor Documentation

subroutine mic_lib::kmp_create_affinity_mask_target::kmp_create_affinity_mask_target (integer (kind=c_int) ***target_type,*** **integer (kind=c_int)** ***target_number,*** **integer (kind=c_intptr_t)** ***mask*** **)**

Definition at line 361 of file mic_lib.f90.

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

- mic_lib.f90

6.23 mic_lib::kmp_destroy_affinity_mask_target Interface Reference

Public Member Functions

- subroutine kmp_destroy_affinity_mask_target (target_type, target_number, mask)

6.23.1 Detailed Description

Definition at line 370 of file mic_lib.f90.

6.23.2 Constructor & Destructor Documentation

subroutine mic_lib::kmp_destroy_affinity_mask_target::kmp_destroy_affinity_mask_target (integer (kind=c_int) ***target_type,*** **integer (kind=c_int)** ***target_number,*** **integer (kind=c_intptr_t)** ***mask*** **)**

Definition at line 370 of file mic_lib.f90.

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

- mic_lib.f90

6.24 mic_lib::kmp_get_affinity_mask_proc_target Interface Reference

Public Member Functions

- integer(kind=c_int) function kmp_get_affinity_mask_proc_target (target_type, target_number, proc, mask)

6.24.1 Detailed Description

Definition at line 429 of file mic_lib.f90.

6.24.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::kmp_get_affinity_mask_proc_target::kmp_get_affinity_mask_proc_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_int) *proc,* integer (kind=c_intptr_t) *mask*)

Definition at line 429 of file mic_lib.f90.

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

- mic_lib.f90

6.25 mic_lib::kmp_get_affinity_max_proc_target Interface Reference

Public Member Functions

- integer(kind=c_int) function kmp_get_affinity_max_proc_target (target_type, target_number)

6.25.1 Detailed Description

Definition at line 399 of file mic_lib.f90.

6.25.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::kmp_get_affinity_max_proc_target::kmp_get_affinity_max_proc_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 399 of file mic_lib.f90.

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

- mic_lib.f90

6.26 mic_lib::kmp_get_affinity_target Interface Reference

Public Member Functions

- integer(kind=c_int) function kmp_get_affinity_target (target_type, target_number, mask)

6.26.1 Detailed Description

Definition at line 389 of file mic_lib.f90.

6.26.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::kmp_get_affinity_target::kmp_get_affinity_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_intptr_t) *mask*)

Definition at line 389 of file mic_lib.f90.

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

- mic_lib.f90

6.27 mic_lib::kmp_get_blocktime_target Interface Reference

Public Member Functions

- integer(kind=c_int) function kmp_get_blocktime_target (target_type, target_number)

6.27.1 Detailed Description

Definition at line 305 of file mic_lib.f90.

6.27.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::kmp_get_blocktime_target::kmp_get_blocktime_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 305 of file mic_lib.f90.

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

- mic_lib.f90

6.28 mic_lib::kmp_get_library_target Interface Reference

Public Member Functions

- integer(kind=c_int) function kmp_get_library_target (target_type, target_number)

6.28.1 Detailed Description

Definition at line 342 of file mic_lib.f90.

6.28.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::kmp_get_library_target::kmp_get_library_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 342 of file mic_lib.f90.

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

- mic_lib.f90

6.29 mic_lib::kmp_get_stacksize_s_target Interface Reference

Public Member Functions

- integer(kind=c_int) function kmp_get_stacksize_s_target (target_type, target_number)

6.29.1 Detailed Description

Definition at line 289 of file mic_lib.f90.

6.29.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::kmp_get_stacksize_s_target::kmp_get_stacksize_s_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 289 of file mic_lib.f90.

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

- mic_lib.f90

6.30 mic_lib::kmp_get_stacksize_target Interface Reference

Public Member Functions

- integer(kind=c_int) function kmp_get_stacksize_target (target_type, target_number)

6.30.1 Detailed Description

Definition at line 273 of file mic_lib.f90.

6.30.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::kmp_get_stacksize_target::kmp_get_stacksize_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 273 of file mic_lib.f90.

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

- mic_lib.f90

6.31 mic_lib::kmp_set_affinity_mask_proc_target Interface Reference

Public Member Functions

- integer(kind=c_int) function kmp_set_affinity_mask_proc_target (target_type, target_number, proc, mask)

6.31.1 Detailed Description

Definition at line 407 of file mic_lib.f90.

6.31.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::kmp_set_affinity_mask_proc_target::kmp_set_affinity_mask_proc_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_int) *proc,* integer (kind=c_intptr_t) *mask*)

Definition at line 407 of file mic_lib.f90.

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

- mic_lib.f90

6.32 mic_lib::kmp_set_affinity_target Interface Reference

Public Member Functions

- integer(kind=c_int) function kmp_set_affinity_target (target_type, target_number, mask)

6.32.1 Detailed Description

Definition at line 379 of file mic_lib.f90.

6.32.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::kmp_set_affinity_target::kmp_set_affinity_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_intptr_t) *mask*)

Definition at line 379 of file mic_lib.f90.

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

- mic_lib.f90

6.33 mic_lib::kmp_set_blocktime_target Interface Reference

Public Member Functions

- subroutine kmp_set_blocktime_target (target_type, target_number, time)

6.33.1 Detailed Description

Definition at line 297 of file mic_lib.f90.

6.33.2 Constructor & Destructor Documentation

subroutine mic_lib::kmp_set_blocktime_target::kmp_set_blocktime_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_int) *time*)

Definition at line 297 of file mic_lib.f90.

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

- mic_lib.f90

6.34 mic_lib::kmp_set_defaults_target Interface Reference

Public Member Functions

- subroutine kmp_set_defaults_target (target_type, target_number, defaults)

6.34.1 Detailed Description

Definition at line 350 of file mic_lib.f90.

6.34.2 Constructor & Destructor Documentation

subroutine mic_lib::kmp_set_defaults_target::kmp_set_defaults_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* character (kind=c_char), dimension(*) *defaults*)

Definition at line 350 of file mic_lib.f90.

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

- mic_lib.f90

6.35 mic_lib::kmp_set_library_serial_target Interface Reference

Public Member Functions

- subroutine kmp_set_library_serial_target (target_type, target_number)

6.35.1 Detailed Description

Definition at line 313 of file mic_lib.f90.

6.35.2 Constructor & Destructor Documentation

subroutine mic_lib::kmp_set_library_serial_target::kmp_set_library_serial_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 313 of file mic_lib.f90.

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

- mic_lib.f90

6.36 mic_lib::kmp_set_library_target Interface Reference

Public Member Functions

- subroutine kmp_set_library_target (target_type, target_number, mode)

6.36.1 Detailed Description

Definition at line 334 of file mic_lib.f90.

6.36.2 Constructor & Destructor Documentation

subroutine mic_lib::kmp_set_library_target::kmp_set_library_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_int) *mode*)

Definition at line 334 of file mic_lib.f90.

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

- mic_lib.f90

6.37 mic_lib::kmp_set_library_throughput_target Interface Reference

Public Member Functions

- subroutine kmp_set_library_throughput_target (target_type, target_number)

6.37.1 Detailed Description

Definition at line 327 of file mic_lib.f90.

6.37.2 Constructor & Destructor Documentation

subroutine mic_lib::kmp_set_library_throughput_target::kmp_set_library_throughput_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 327 of file mic_lib.f90.

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

- mic_lib.f90

6.38 mic_lib::kmp_set_library_turnaround_target Interface Reference

Public Member Functions

- subroutine kmp_set_library_turnaround_target (target_type, target_number)

6.38.1 Detailed Description

Definition at line 320 of file mic_lib.f90.

6.38.2 Constructor & Destructor Documentation

subroutine mic_lib::kmp_set_library_turnaround_target::kmp_set_library_turnaround_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 320 of file mic_lib.f90.

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

- mic_lib.f90

6.39 mic_lib::kmp_set_stacksize_s_target Interface Reference

Public Member Functions

- subroutine kmp_set_stacksize_s_target (target_type, target_number, size)

6.39.1 Detailed Description

Definition at line 281 of file mic_lib.f90.

6.39.2 Constructor & Destructor Documentation

subroutine mic_lib::kmp_set_stacksize_s_target::kmp_set_stacksize_s_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_int) *size*)

Definition at line 281 of file mic_lib.f90.

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

- mic_lib.f90

6.40 mic_lib::kmp_set_stacksize_target Interface Reference

Public Member Functions

- subroutine kmp_set_stacksize_target (target_type, target_number, size)

6.40.1 Detailed Description

Definition at line 265 of file mic_lib.f90.

6.40.2 Constructor & Destructor Documentation

subroutine mic_lib::kmp_set_stacksize_target::kmp_set_stacksize_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_int) *size*)

Definition at line 265 of file mic_lib.f90.

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

- mic_lib.f90

6.41 mic_lib::kmp_unset_affinity_mask_proc_target Interface Reference

Public Member Functions

- integer(kind=c_int) function kmp_unset_affinity_mask_proc_target (target_type, target_number, proc, mask)

6.41.1 Detailed Description

Definition at line 418 of file mic_lib.f90.

6.41.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::kmp_unset_affinity_mask_proc_target::kmp_unset_affinity_mask_proc_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_int) *proc,* integer (kind=c_intptr_t) *mask*)

Definition at line 418 of file mic_lib.f90.

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

- mic_lib.f90

6.42 Marshaller Class Reference

```
#include <offload_common.h>
```

Public Member Functions

- Marshaller ()
- long long get_tfr_size () const
- char * get_buffer_start () const
- long long get_buffer_size () const
- void init_buffer (char *d, long long s)
- void send_data (const void *data, int64_t length)
- void receive_data (void *data, int64_t length)
- void send_func_ptr (const void *data)
- void receive_func_ptr (const void **data)

Private Attributes

- char * buffer_start
- char * buffer_ptr
- long long buffer_size
- long long tfr_size

6.42.1 Detailed Description

Definition at line 330 of file offload_common.h.

6.42.2 Constructor & Destructor Documentation

Marshaller::Marshaller ( ) [inline]

Definition at line 347 of file offload_common.h.

6.42.3 Member Function Documentation

long long Marshaller::get_buffer_size ( ) const [inline]

Definition at line 366 of file offload_common.h.
Referenced by OffloadDescriptor::gather_copyout_data(), and OffloadDescriptor::scatter_copyin_data().

char* Marshaller::get_buffer_start ( ) const [inline]

Definition at line 360 of file offload_common.h.
Referenced by OffloadDescriptor::gather_copyout_data(), and OffloadDescriptor::scatter_copyin_data().

long long Marshaller::get_tfr_size ( ) const [inline]

Definition at line 354 of file offload_common.h.
Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::scatter_copyout_data().

void Marshaller::init_buffer (char * *d,* long long *s*) [inline]

Definition at line 372 of file offload_common.h.
Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::offload(), and OffloadDescriptor::scatter_copyout_data().

void Marshaller::receive_data (void * *data,* int64_t *length*)

Definition at line 69 of file offload_common.cpp.
Referenced by OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::scatter_copyout_data().

void Marshaller::receive_func_ptr (const void ** *data*)

Definition at line 114 of file offload_common.cpp.
Referenced by OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::scatter_copyout_data().

void Marshaller::send_data (const void * *data,* int64_t *length*)

Definition at line 57 of file offload_common.cpp.
Referenced by OffloadDescriptor::gather_copyin_data(), and OffloadDescriptor::gather_copyout_data().

void Marshaller::send_func_ptr (const void * *data*)

Definition at line 82 of file offload_common.cpp.
Referenced by OffloadDescriptor::gather_copyin_data(), and OffloadDescriptor::gather_copyout_data().

6.42.4 Member Data Documentation

char* Marshaller::buffer_ptr [private]

Definition at line 337 of file offload_common.h.
Referenced by init_buffer(), receive_data(), receive_func_ptr(), send_data(), and send_func_ptr().

long long Marshaller::buffer_size [private]

Definition at line 340 of file offload_common.h.
Referenced by get_buffer_size(), and init_buffer().

char* Marshaller::buffer_start [private]

Definition at line 334 of file offload_common.h.
Referenced by get_buffer_start(), and init_buffer().

long long Marshaller::tfr_size [private]

Definition at line 343 of file offload_common.h.
Referenced by get_tfr_size(), receive_data(), receive_func_ptr(), send_data(), and send_func_ptr().
The documentation for this class was generated from the following files:

- offload_common.h
- offload_common.cpp

6.43 MemRange Class Reference

`#include <offload_engine.h>`

Public Member Functions

- MemRange ()
- MemRange (const void *addr, uint64_t len)
- const void * start () const
- const void * end () const
- uint64_t length () const
- bool overlaps (const MemRange &o) const
- bool contains (const MemRange &o) const

Private Attributes

- const void * m_start
- uint64_t m_length

6.43.1 Detailed Description

Definition at line 23 of file offload_engine.h.

6.43.2 Constructor & Destructor Documentation

MemRange::MemRange ( ) [inline]

Definition at line 25 of file offload_engine.h.

MemRange::MemRange (const void * *addr,* uint64_t *len*) [inline]

Definition at line 26 of file offload_engine.h.

6.43.3 Member Function Documentation

bool MemRange::contains (const MemRange & *o*) const [inline]

Definition at line 48 of file offload_engine.h.
Referenced by OffloadDescriptor::alloc_ptr_data(), and OffloadDescriptor::find_ptr_data().

const void* MemRange::end ( ) const [inline]

Definition at line 32 of file offload_engine.h.
Referenced by contains(), and overlaps().

uint64_t MemRange::length ( ) const [inline]

Definition at line 36 of file offload_engine.h.
Referenced by OffloadDescriptor::alloc_ptr_data(), OffloadDescriptor::find_ptr_data(), and OffloadDescriptor::init_static_ptr_data().

bool MemRange::overlaps (const MemRange & *o*) const [inline]

Definition at line 41 of file offload_engine.h.
Referenced by PtrData::operator<(), and AutoData::operator<().

const void* MemRange::start ( ) const [inline]

Definition at line 28 of file offload_engine.h.
Referenced by OffloadDescriptor::alloc_ptr_data(), contains(), OffloadDescriptor::find_ptr_data(), OffloadDescriptor::init_static_ptr_data(), PtrData::operator<(), AutoData::operator<(), overlaps(), OffloadDescriptor::receive_pointer_data(), and OffloadDescriptor::setup_descriptors().

6.43.4 Member Data Documentation

uint64_t MemRange::m_length [private]

Definition at line 54 of file offload_engine.h.
Referenced by end(), and length().

const void* MemRange::m_start [private]

Definition at line 53 of file offload_engine.h.
Referenced by end(), and start().
The documentation for this class was generated from the following file:

- offload_engine.h

6.44 mic_lib Module Reference

Data Types

- interface kmp_create_affinity_mask_target
- interface kmp_destroy_affinity_mask_target
- interface kmp_get_affinity_mask_proc_target
- interface kmp_get_affinity_max_proc_target
- interface kmp_get_affinity_target
- interface kmp_get_blocktime_target
- interface kmp_get_library_target
- interface kmp_get_stacksize_s_target
- interface kmp_get_stacksize_target
- interface kmp_set_affinity_mask_proc_target
- interface kmp_set_affinity_target
- interface kmp_set_blocktime_target
- interface kmp_set_defaults_target
- interface kmp_set_library_serial_target
- interface kmp_set_library_target
- interface kmp_set_library_throughput_target
- interface kmp_set_library_turnaround_target
- interface kmp_set_stacksize_s_target
- interface kmp_set_stacksize_target
- interface kmp_unset_affinity_mask_proc_target
- interface offload_get_device_number
- interface offload_get_physical_device_number
- interface offload_number_of_devices
- interface offload_report
- interface offload_signaled
- type offload_status
- interface omp_destroy_lock_target
- interface omp_destroy_nest_lock_target
- interface omp_get_dynamic_target
- interface omp_get_max_threads_target
- interface omp_get_nested_target
- interface omp_get_num_procs_target
- interface omp_get_schedule_target
- interface omp_init_lock_target
- interface omp_init_nest_lock_target
- interface omp_set_dynamic_target
- interface omp_set_lock_target
- interface omp_set_nest_lock_target
- interface omp_set_nested_target
- interface omp_set_num_threads_target
- interface omp_set_schedule_target
- interface omp_test_lock_target
- interface omp_test_nest_lock_target
- interface omp_unset_lock_target
- interface omp_unset_nest_lock_target

Public Attributes

- integer, parameter target_mic =2
- integer, parameter default_target_type =target_mic
- integer, parameter default_target_number =0

6.44.1 Detailed Description

Definition at line 19 of file mic_lib.f90.

6.44.2 Member Data Documentation

integer, parameter mic_lib::default_target_number =0

Definition at line 24 of file mic_lib.f90.

integer, parameter mic_lib::default_target_type =target_mic

Definition at line 23 of file mic_lib.f90.

integer, parameter mic_lib::target_mic =2

Definition at line 22 of file mic_lib.f90.

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

- mic_lib.f90

6.45 MicEnvVar Struct Reference

`#include <offload_env.h>`

Classes

- struct CardEnvVars
- struct VarValue

Public Member Functions

- MicEnvVar ()
- ~MicEnvVar ()
- void analyze_env_var (char *env_var_string)
- char ** create_environ_for_card (int card_num)
- MicEnvVarKind get_env_var_kind (char *env_var_string, int *card_number, char **env_var_name, int *env_var_name_length, char **env_var_def)
- void add_env_var (int card_number, char *env_var_name, int env_var_name_length, char *env_var_def)
- void set_prefix (const char *pref)

Static Public Attributes

- static const int any_card = -1

Private Member Functions

- void mic_parse_env_var_list (int card_number, char *env_var_def)
- CardEnvVars * get_card (int number)

Private Attributes

- const char * prefix
- std::list< struct CardEnvVars * > card_spec_list
- CardEnvVars common_vars

6.45.1 Detailed Description

Definition at line 26 of file offload_env.h.

6.45.2 Constructor & Destructor Documentation

MicEnvVar::MicEnvVar ( ) [inline]

Definition at line 28 of file offload_env.h.

MicEnvVar::~MicEnvVar ( )

Definition at line 20 of file offload_env.cpp.

6.45.3 Member Function Documentation

void MicEnvVar::add_env_var (int *card_number,* char * *env_var_name,* int *env_var_name_length,* char * *env_var_def*)

Definition at line 111 of file offload_env.cpp.
Referenced by analyze_env_var(), and mic_parse_env_var_list().

void MicEnvVar::analyze_env_var (char * *env_var_string*)

Definition at line 81 of file offload_env.cpp.
Referenced by __offload_init_library_once().

char ** MicEnvVar::create_environ_for_card (int *card_num*)

Definition at line 310 of file offload_env.cpp.
Referenced by Engine::init_process().

MicEnvVar::CardEnvVars * MicEnvVar::get_card (int *number*) [private]

Definition at line 46 of file offload_env.cpp.
Referenced by add_env_var(), and create_environ_for_card().

MicEnvVarKind MicEnvVar::get_env_var_kind (char * *env_var_string,* int * *card_number,* char ** *env_var_name,* int * *env_var_name_length,* char ** *env_var_def*)

Definition at line 156 of file offload_env.cpp.
Referenced by analyze_env_var().

void MicEnvVar::mic_parse_env_var_list (int *card_number,* char * *env_var_def*) [private]

Definition at line 228 of file offload_env.cpp.
Referenced by analyze_env_var().

void MicEnvVar::set_prefix (const char * *pref*) [inline]

Definition at line 47 of file offload_env.h.
Referenced by __offload_init_library_once().

6.45.4 Member Data Documentation

const int MicEnvVar::any_card = -1 [static]

Definition at line 80 of file offload_env.h.
Referenced by add_env_var(), MicEnvVar::CardEnvVars::CardEnvVars(), create_environ_for_card(), get_card(), and get_env_var_kind().

std::list<struct CardEnvVars *> MicEnvVar::card_spec_list [private]

Definition at line 87 of file offload_env.h.
Referenced by add_env_var(), get_card(), and ~MicEnvVar().

CardEnvVars MicEnvVar::common_vars [private]

Definition at line 88 of file offload_env.h.
Referenced by add_env_var(), and get_card().

const char* MicEnvVar::prefix [private]

Definition at line 86 of file offload_env.h.
Referenced by create_environ_for_card(), get_env_var_kind(), and set_prefix().
The documentation for this struct was generated from the following files:

- offload_env.h
- offload_env.cpp

6.46 mutex_locker_t Struct Reference

`#include <offload_util.h>`

Public Member Functions

- mutex_locker_t (mutex_t &mutex)
- ~mutex_locker_t ()

Private Attributes

- mutex_t & m_mutex

6.46.1 Detailed Description

Definition at line 94 of file offload_util.h.

6.46.2 Constructor & Destructor Documentation

mutex_locker_t::mutex_locker_t (mutex_t & *mutex*) [inline]

Definition at line 95 of file offload_util.h.

mutex_locker_t::~mutex_locker_t ( ) [inline]

Definition at line 99 of file offload_util.h.

6.46.3 Member Data Documentation

mutex_t& mutex_locker_t::m_mutex [private]

Definition at line 104 of file offload_util.h.
Referenced by mutex_locker_t(), and ~mutex_locker_t().
The documentation for this struct was generated from the following file:

- offload_util.h

6.47 mutex_t Struct Reference

`#include <offload_util.h>`

Public Member Functions

- mutex_t ()
- ~mutex_t ()
- void lock ()
- void unlock ()

Private Attributes

- pthread_mutex_t m_lock

6.47.1 Detailed Description

Definition at line 53 of file offload_util.h.

6.47.2 Constructor & Destructor Documentation

mutex_t::mutex_t ( ) [inline]

Definition at line 54 of file offload_util.h.

mutex_t::~mutex_t ( ) [inline]

Definition at line 62 of file offload_util.h.

6.47.3 Member Function Documentation

void mutex_t::lock ( ) [inline]

Definition at line 70 of file offload_util.h.

Referenced by __offload_register_image(), Engine::add_lib(), Engine::add_signal(), TableList< VarTable >::add_table(), FuncList::dump(), VarList::dump(), FuncList::find_addr(), FuncList::find_name(), Engine::find_ptr_data(), Engine::find_signal(), Engine::insert_ptr_data(), FuncList::max_name_length(), mutex_locker_t::mutex_locker_t(), mic_lib::omp_destroy_lock_target::omp_destroy_lock_target(), mic_lib::omp_destroy_nest_lock_target::omp_destroy_nest_lock_target(), mic_lib::omp_init_lock_target::omp_init_lock_target(), mic_lib::omp_init_nest_lock_target::omp_init_nest_lock_target(), mic_lib::omp_set_lock_target::omp_set_lock_target(), mic_lib::omp_set_nest_lock_target::omp_set_nest_lock_target(), mic_lib::omp_test_lock_target::omp_test_lock_target(), mic_lib::omp_test_nest_lock_target::omp_test_nest_lock_target(), mic_lib::omp_unset_lock_target::omp_unset_lock_target(), mic_lib::omp_unset_nest_lock_target::omp_unset_nest_lock_target(), Engine::remove_ptr_data(), and TableList< VarTable >::remove_table().

void mutex_t::unlock ( ) [inline]

Definition at line 78 of file offload_util.h.

Referenced by __offload_register_image(), Engine::add_lib(), Engine::add_signal(), TableList< VarTable >::add_table(), OffloadDescriptor::alloc_ptr_data(), FuncList::dump(), VarList::dump(), FuncList::find_addr(), FuncList::find_name(), Engine::find_ptr_data(), Engine::find_signal(), Engine::insert_ptr_data(), FuncList::max_name_length(), Engine::remove_ptr_data(), TableList< VarTable >::remove_table(), and mutex_locker_t::~mutex_locker_t().

6.47.4 Member Data Documentation

pthread_mutex_t mutex_t::m_lock [private]

Definition at line 90 of file offload_util.h.

Referenced by lock(), mutex_t(), unlock(), and ~mutex_t().

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

- offload_util.h

6.48 MyoTable Struct Reference

Public Member Functions

- MyoTable (SharedTableEntry *tab, int len)

Public Attributes

- SharedTableEntry * var_tab
- int var_tab_len

6.48.1 Detailed Description

Definition at line 306 of file offload_myo_host.cpp.

6.48.2 Constructor & Destructor Documentation

MyoTable::MyoTable (SharedTableEntry * *tab,* int *len*) [inline]

Definition at line 308 of file offload_myo_host.cpp.

6.48.3 Member Data Documentation

SharedTableEntry* MyoTable::var_tab

Definition at line 311 of file offload_myo_host.cpp.

int MyoTable::var_tab_len

Definition at line 312 of file offload_myo_host.cpp.

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

- offload_myo_host.cpp

6.49 MyoWrapper Class Reference

Public Member Functions

- MyoWrapper ()
- bool is_available () const
- bool LoadLibrary (void)
- void UnloadLibrary (void)
- void LibInit (void *arg, void *func) const
- void LibFini (void) const
- void * SharedMalloc (size_t size) const
- void SharedFree (void *ptr) const
- void * SharedAlignedMalloc (size_t size, size_t align) const
- void SharedAlignedFree (void *ptr) const
- void Acquire (void) const
- void Release (void) const
- void HostVarTablePropagate (void *table, int num_entries) const
- void HostFptrTableRegister (void *table, int num_entries, int ordered) const
- void RemoteThunkCall (void *thunk, void *args, int device)
- MyoiRFuncCallHandle RemoteCall (char *func, void *args, int device) const
- void GetResult (MyoiRFuncCallHandle handle) const

Private Member Functions

- void CheckResult (const char *func, MyoError error) const

Private Attributes

- void * m_lib_handle
- bool m_is_available
- MyoError(* m_lib_init)(void *, void *)
- void(* m_lib_fini)(void)
- void *(* m_shared_malloc)(size_t)
- void(* m_shared_free)(void *)
- void *(* m_shared_aligned_malloc)(size_t, size_t)
- void(* m_shared_aligned_free)(void *)
- MyoError(* m_acquire)(void)
- MyoError(* m_release)(void)
- MyoError(* m_host_var_table_propagate)(void *, int)
- MyoError(* m_host_fptr_table_register)(void *, int, int)
- MyoError(* m_remote_thunk_call)(void *, void *, int)
- MyoiRFuncCallHandle(* m_remote_call)(char *, void *, int)
- MyoError(* m_get_result)(MyoiRFuncCallHandle)

6.49.1 Detailed Description

Definition at line 36 of file offload_myo_host.cpp.

6.49.2 Constructor & Destructor Documentation

MyoWrapper::MyoWrapper ( ) [inline]

Definition at line 38 of file offload_myo_host.cpp.

6.49.3 Member Function Documentation

void MyoWrapper::Acquire (void) const [inline]

Definition at line 91 of file offload_myo_host.cpp.
Referenced by __intel_cilk_for_32_offload(), __intel_cilk_for_64_offload(), and __offload_myoiRemoteIThunkCall().

void MyoWrapper::CheckResult (const char * *func,* MyoError *error*) const [inline], [private]

Definition at line 135 of file offload_myo_host.cpp.
Referenced by Acquire(), GetResult(), HostFptrTableRegister(), HostVarTablePropagate(), LibInit(), Release(), and RemoteThunkCall().

void MyoWrapper::GetResult (MyoiRFuncCallHandle *handle*) const [inline]

Definition at line 129 of file offload_myo_host.cpp.
Referenced by __intel_cilk_for_32_offload(), and __intel_cilk_for_64_offload().

void MyoWrapper::HostFptrTableRegister (void * *table,* int *num_entries,* int *ordered*) const [inline]

Definition at line 108 of file offload_myo_host.cpp.
Referenced by __offload_myo_fptr_table_register().

void MyoWrapper::HostVarTablePropagate (void * *table,* int *num_entries*) const [inline]

Definition at line 103 of file offload_myo_host.cpp.
Referenced by __offload_myoInit().

bool MyoWrapper::is_available ( ) const [inline]

Definition at line 41 of file offload_myo_host.cpp.
Referenced by __offload_myo_fptr_table_register(), and __offload_myoLoadLibrary().

void MyoWrapper::LibFini (void) const [inline]

Definition at line 62 of file offload_myo_host.cpp.
Referenced by __offload_myoFini().

void MyoWrapper::LibInit (void * *arg,* void * *func*) const [inline]

Definition at line 56 of file offload_myo_host.cpp.
Referenced by __offload_myoInit_once().

bool MyoWrapper::LoadLibrary (void)

Definition at line 162 of file offload_myo_host.cpp.
Referenced by __offload_myoLoadLibrary_once().

void MyoWrapper::Release (void) const [inline]

Definition at line 97 of file offload_myo_host.cpp.
Referenced by __intel_cilk_for_32_offload(), __intel_cilk_for_64_offload(), and __offload_myoiRemoteIThunkCall().

MyoiRFuncCallHandle MyoWrapper::RemoteCall (char * *func,* void * *args,* int *device*) const [inline]

Definition at line 123 of file offload_myo_host.cpp.
Referenced by __intel_cilk_for_32_offload(), and __intel_cilk_for_64_offload().

void MyoWrapper::RemoteThunkCall (void * *thunk,* void * *args,* int *device*) [inline]

Definition at line 117 of file offload_myo_host.cpp.
Referenced by __offload_myoiRemoteIThunkCall().

void MyoWrapper::SharedAlignedFree (void * *ptr*) const [inline]

Definition at line 85 of file offload_myo_host.cpp.
Referenced by _Offload_shared_aligned_free().

void* MyoWrapper::SharedAlignedMalloc (size_t *size,* size_t *align*) const [inline]

Definition at line 79 of file offload_myo_host.cpp.
Referenced by _Offload_shared_aligned_malloc().

void MyoWrapper::SharedFree (void * *ptr*) const [inline]

Definition at line 73 of file offload_myo_host.cpp.
Referenced by _Offload_shared_free().

void* MyoWrapper::SharedMalloc (size_t *size*) const [inline]

Definition at line 67 of file offload_myo_host.cpp.
Referenced by _Offload_shared_malloc().

void MyoWrapper::UnloadLibrary (void) [inline]

Definition at line 48 of file offload_myo_host.cpp.
Referenced by LoadLibrary().

6.49.4 Member Data Documentation

MyoError(∗ MyoWrapper::m_acquire)(void) [private]

Definition at line 153 of file offload_myo_host.cpp.
Referenced by Acquire(), and LoadLibrary().

MyoError(∗ MyoWrapper::m_get_result)(MyoiRFuncCallHandle) [private]

Definition at line 159 of file offload_myo_host.cpp.
Referenced by GetResult(), and LoadLibrary().

MyoError(∗ MyoWrapper::m_host_fptr_table_register)(void ∗, int, int) [private]

Definition at line 156 of file offload_myo_host.cpp.
Referenced by HostFptrTableRegister(), and LoadLibrary().

MyoError(∗ MyoWrapper::m_host_var_table_propagate)(void ∗, int) [private]

Definition at line 155 of file offload_myo_host.cpp.
Referenced by HostVarTablePropagate(), and LoadLibrary().

bool MyoWrapper::m_is_available [private]

Definition at line 144 of file offload_myo_host.cpp.
Referenced by is_available(), and LoadLibrary().

void(∗ MyoWrapper::m_lib_fini)(void) [private]

Definition at line 148 of file offload_myo_host.cpp.
Referenced by LibFini(), and LoadLibrary().

void∗ MyoWrapper::m_lib_handle [private]

Definition at line 143 of file offload_myo_host.cpp.
Referenced by LoadLibrary().

MyoError(∗ MyoWrapper::m_lib_init)(void ∗, void ∗) [private]

Definition at line 147 of file offload_myo_host.cpp.
Referenced by LibInit(), and LoadLibrary().

MyoError(∗ MyoWrapper::m_release)(void) [private]

Definition at line 154 of file offload_myo_host.cpp.
Referenced by LoadLibrary(), and Release().

MyoiRFuncCallHandle(∗ MyoWrapper::m_remote_call)(char ∗, void ∗, int) [private]

Definition at line 158 of file offload_myo_host.cpp.
Referenced by LoadLibrary(), and RemoteCall().

MyoError(∗ MyoWrapper::m_remote_thunk_call)(void ∗, void ∗, int) [private]

Definition at line 157 of file offload_myo_host.cpp.
Referenced by LoadLibrary(), and RemoteThunkCall().

void(∗ MyoWrapper::m_shared_aligned_free)(void ∗) [private]

Definition at line 152 of file offload_myo_host.cpp.
Referenced by LoadLibrary(), and SharedAlignedFree().

void*(* MyoWrapper::m_shared_aligned_malloc)(size_t, size_t) [private]

Definition at line 151 of file offload_myo_host.cpp.
Referenced by LoadLibrary(), and SharedAlignedMalloc().

void(* MyoWrapper::m_shared_free)(void *) [private]

Definition at line 150 of file offload_myo_host.cpp.
Referenced by LoadLibrary(), and SharedFree().

void*(* MyoWrapper::m_shared_malloc)(size_t) [private]

Definition at line 149 of file offload_myo_host.cpp.
Referenced by LoadLibrary(), and SharedMalloc().
The documentation for this class was generated from the following file:

- offload_myo_host.cpp

6.50 TableList< T >::Node Struct Reference

`#include <offload_table.h>`

Public Attributes

- Table table
- Node * prev
- Node * next

6.50.1 Detailed Description

template<typename T>struct TableList< T >::Node

Definition at line 28 of file offload_table.h.

6.50.2 Member Data Documentation

template<typename T> Node* TableList< T >::Node::next

Definition at line 31 of file offload_table.h.
Referenced by TableList< VarTable >::remove_table().

template<typename T> Node* TableList< T >::Node::prev

Definition at line 30 of file offload_table.h.
Referenced by TableList< VarTable >::add_table().

template<typename T> Table TableList< T >::Node::table

Definition at line 29 of file offload_table.h.
The documentation for this struct was generated from the following file:

- offload_table.h

6.51 mic_lib::offload_get_device_number Interface Reference

Public Member Functions

- integer(kind=c_int) function offload_get_device_number ()

6.51.1 Detailed Description

Definition at line 70 of file mic_lib.f90.

6.51.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::offload_get_device_number::offload_get_device_number ( )

Definition at line 70 of file mic_lib.f90.

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

- mic_lib.f90

6.52 mic_lib::offload_get_physical_device_number Interface Reference

Public Member Functions

- integer(kind=c_int) function offload_get_physical_device_number ()

6.52.1 Detailed Description

Definition at line 78 of file mic_lib.f90.

6.52.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::offload_get_physical_device_number::offload_get_physical_device_number ( )

Definition at line 78 of file mic_lib.f90.

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

- mic_lib.f90

6.53 mic_lib::offload_number_of_devices Interface Reference

Public Member Functions

- integer(kind=c_int) function offload_number_of_devices ()

6.53.1 Detailed Description

Definition at line 43 of file mic_lib.f90.

6.53.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::offload_number_of_devices::offload_number_of_devices ( )

Definition at line 43 of file mic_lib.f90.

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

- mic_lib.f90

6.54 mic_lib::offload_report Interface Reference

Public Member Functions

- subroutine offload_report (val)

6.54.1 Detailed Description

Definition at line 62 of file mic_lib.f90.

6.54.2 Constructor & Destructor Documentation

subroutine mic_lib::offload_report::offload_report (integer (kind=c_int) *val*)

Definition at line 62 of file mic_lib.f90.

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

- mic_lib.f90

6.55 mic_lib::offload_signaled Interface Reference

Public Member Functions

- integer(kind=c_int) function offload_signaled (target_number, signal)

6.55.1 Detailed Description

Definition at line 52 of file mic_lib.f90.

6.55.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::offload_signaled::offload_signaled (integer (kind=c_int) *target_number,* integer (kind=c_int64_t) *signal*)

Definition at line 52 of file mic_lib.f90.

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

- mic_lib.f90

6.56 mic_lib::offload_status Type Reference

Public Attributes

- integer(kind=c_int) result = OFFLOAD_DISABLED
- integer(kind=c_int) device_number = -1
- integer(kind=c_size_t) data_sent = 0
- integer(kind=c_size_t) data_received = 0

6.56.1 Detailed Description

Definition at line 35 of file mic_lib.f90.

6.56.2 Member Data Documentation

integer(kind=c_size_t) mic_lib::offload_status::data_received = 0

Definition at line 39 of file mic_lib.f90.

integer(kind=c_size_t) mic_lib::offload_status::data_sent = 0

Definition at line 38 of file mic_lib.f90.

integer(kind=c_int) mic_lib::offload_status::device_number = -1

Definition at line 37 of file mic_lib.f90.

integer(kind=c_int) mic_lib::offload_status::result = OFFLOAD_DISABLED

Definition at line 36 of file mic_lib.f90.

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

- mic_lib.f90

6.57 OffloadDescriptor Class Reference

`#include <offload_host.h>`

Classes

- class ReadArrElements
- struct VarExtra

Public Member Functions

- OffloadDescriptor (int index, _Offload_status *status, bool is_mandatory, bool is_openmp, OffloadHostTimerData *timer_data)
- ~OffloadDescriptor ()
- bool offload (const char *name, bool is_empty, VarDesc *vars, VarDesc2 *vars2, int vars_total, const void **waits, int num_waits, const void **signal, int entry_id, const void *stack_addr)
- bool offload_finish ()
- bool is_signaled ()
- OffloadHostTimerData * get_timer_data () const
- ~OffloadDescriptor ()
- void scatter_copyin_data ()
- void gather_copyout_data ()
- void merge_var_descs (VarDesc *vars, VarDesc2 *vars2, int vars_total)
- int get_offload_number () const
- void set_offload_number (int number)

Static Public Member Functions

- static void offload (uint32_t buffer_count, void **buffers, void *misc_data, uint16_t misc_data_len, void *return_data, uint16_t return_data_len)

Private Types

- typedef std::list< COIBUFFER > BufferList
- typedef std::list< void * > BufferList

Private Member Functions

- bool wait_dependencies (const void **waits, int num_waits)
- bool setup_descriptors (VarDesc *vars, VarDesc2 *vars2, int vars_total, int entry_id, const void *stack_addr)
- bool setup_misc_data (const char *name)
- bool send_pointer_data (bool is_async)
- bool send_noncontiguous_pointer_data (int i, PtrData *src_buf, PtrData *dst_buf, COIEVENT *event)
- bool recieve_noncontiguous_pointer_data (int i, char *src_data, COIBUFFER dst_buf, COIEVENT *event)
- bool gather_copyin_data ()
- bool compute ()
- bool receive_pointer_data (bool is_async)
- bool scatter_copyout_data ()
- void cleanup ()
- bool find_ptr_data (PtrData *&ptr_data, void *base, int64_t disp, int64_t length, bool error_does_not_exist=true)

- bool alloc_ptr_data (PtrData *&ptr_data, void *base, int64_t disp, int64_t length, int64_t alloc_disp, int align)
- bool init_static_ptr_data (PtrData *ptr_data)
- bool init_mic_address (PtrData *ptr_data)
- bool offload_stack_memory_manager (const void *stack_begin, int routine_id, int buf_size, int align, bool *is_new)
- bool nullify_target_stack (COIBUFFER targ_buf, uint64_t size)
- bool gen_var_descs_for_pointer_array (int i)
- void report_coi_error (error_types msg, COIRESULT res)
- _Offload_result translate_coi_error (COIRESULT res) const
- OffloadDescriptor ()

Private Attributes

- PtrData * m_stack_ptr_data
- PtrDataList m_destroy_stack
- Engine & m_device
- bool m_is_mandatory
- const bool m_is_openmp
- Marshaller m_in
- Marshaller m_out
- BufferList m_compute_buffers
- BufferList m_destroy_buffers
- VarDesc * m_vars
- VarExtra * m_vars_extra
- int m_vars_total
- _Offload_status * m_status
- FunctionDescriptor * m_func_desc
- uint32_t m_func_desc_size
- COIBUFFER m_inout_buf
- COIEVENT * m_in_deps
- uint32_t m_in_deps_total
- COIEVENT * m_out_deps
- uint32_t m_out_deps_total
- OffloadHostTimerData * m_timer_data
- uint64_t m_in_datalen
- uint64_t m_out_datalen
- bool m_need_runfunction
- BufferList m_buffers
- int m_offload_number

6.57.1 Detailed Description

Definition at line 44 of file offload_host.h.

6.57.2 Member Typedef Documentation

typedef std::list<void*> OffloadDescriptor::BufferList `[private]`

Definition at line 64 of file offload_target.h.

typedef std::list<COIBUFFER> OffloadDescriptor::BufferList `[private]`

Definition at line 141 of file offload_host.h.

6.57.3 Constructor & Destructor Documentation

OffloadDescriptor::OffloadDescriptor (int *index,* _Offload_status * *status,* bool *is_mandatory,* bool *is_openmp,* OffloadHostTimerData * *timer_data*) [inline]

Definition at line 47 of file offload_host.h.

OffloadDescriptor::~OffloadDescriptor ( ) [inline]

Definition at line 70 of file offload_host.h.

OffloadDescriptor::~OffloadDescriptor ( ) [inline]

Definition at line 23 of file offload_target.h.

OffloadDescriptor::OffloadDescriptor ( ) [inline], [private]

Definition at line 60 of file offload_target.h.

6.57.4 Member Function Documentation

bool OffloadDescriptor::alloc_ptr_data (PtrData *& *ptr_data,* void * *base,* int64_t *disp,* int64_t *length,* int64_t *alloc_disp,* int *align*) [private]

Definition at line 261 of file offload_host.cpp.
Referenced by setup_descriptors().

void OffloadDescriptor::cleanup ( ) [private]

Definition at line 2262 of file offload_host.cpp.
Referenced by offload(), and wait_dependencies().

bool OffloadDescriptor::compute ( ) [private]

Definition at line 2872 of file offload_host.cpp.
Referenced by offload().

bool OffloadDescriptor::find_ptr_data (PtrData *& *ptr_data,* void * *base,* int64_t *disp,* int64_t *length,* bool *error_does_not_exist = true*) [private]

Definition at line 414 of file offload_host.cpp.
Referenced by setup_descriptors().

bool OffloadDescriptor::gather_copyin_data ( ) [private]

Definition at line 2726 of file offload_host.cpp.
Referenced by offload().

void OffloadDescriptor::gather_copyout_data ( )

Definition at line 545 of file offload_target.cpp.
Referenced by OFFLOAD_TARGET_LEAVE().

bool OffloadDescriptor::gen_var_descs_for_pointer_array (int *i*) [private]

Definition at line 3490 of file offload_host.cpp.
Referenced by setup_descriptors().

int OffloadDescriptor::get_offload_number ( ) const [inline]

Definition at line 50 of file offload_target.h.
Referenced by gather_copyout_data(), merge_var_descs(), and offload().

OffloadHostTimerData* OffloadDescriptor::get_timer_data ( ) const [inline]

Definition at line 95 of file offload_host.h.
Referenced by alloc_ptr_data(), cleanup(), compute(), gather_copyin_data(), init_static_ptr_data(), offload(), offload_finish(), receive_pointer_data(), scatter_copyout_data(), send_pointer_data(), setup_descriptors(), setup_misc_data(), and wait_dependencies().

bool OffloadDescriptor::init_mic_address (PtrData * *ptr_data*) [private]

Definition at line 514 of file offload_host.cpp.
Referenced by offload_stack_memory_manager(), and setup_descriptors().

bool OffloadDescriptor::init_static_ptr_data (PtrData * *ptr_data*) [private]

Definition at line 465 of file offload_host.cpp.
Referenced by alloc_ptr_data(), and find_ptr_data().

bool OffloadDescriptor::is_signaled ( )

Definition at line 2273 of file offload_host.cpp.
Referenced by _Offload_signaled().

void OffloadDescriptor::merge_var_descs (VarDesc * *vars,* VarDesc2 * *vars2,* int *vars_total*)

Definition at line 254 of file offload_target.cpp.
Referenced by OFFLOAD_TARGET_ENTER().

bool OffloadDescriptor::nullify_target_stack (COIBUFFER *targ_buf,* uint64_t *size*) [private]

Definition at line 532 of file offload_host.cpp.
Referenced by offload_stack_memory_manager().

void OffloadDescriptor::offload (uint32_t *buffer_count,* void ** *buffers,* void * *misc_data,* uint16_t *misc_data_len,* void * *return_data,* uint16_t *return_data_len*) [static]

Definition at line 126 of file offload_target.cpp.

bool OffloadDescriptor::offload (const char * *name,* bool *is_empty,* VarDesc * *vars,* VarDesc2 * *vars2,* int *vars_total,* const void ** *waits,* int *num_waits,* const void ** *signal,* int *entry_id,* const void * *stack_addr*)

Definition at line 2077 of file offload_host.cpp.
Referenced by offload_offload_wrap(), and server_compute().

bool OffloadDescriptor::offload_finish ( )

Definition at line 2186 of file offload_host.cpp.
Referenced by offload(), and wait_dependencies().

bool OffloadDescriptor::offload_stack_memory_manager (const void * *stack_begin,* int *routine_id,* int *buf_size,* int *align,* bool * *is_new*) [private]

Definition at line 559 of file offload_host.cpp.
Referenced by setup_descriptors().

bool OffloadDescriptor::receive_pointer_data (bool *is_async*) [private]

Definition at line 3050 of file offload_host.cpp.
Referenced by offload().

bool OffloadDescriptor::recieve_noncontiguous_pointer_data (int *i,* char * *src_data,* COIBUFFER *dst_buf,* COIEVENT * *event*) [private]

Definition at line 2921 of file offload_host.cpp.
Referenced by receive_pointer_data().

void OffloadDescriptor::report_coi_error (error_types *msg,* COIRESULT *res*) [private]

Definition at line 205 of file offload_host.cpp.
Referenced by alloc_ptr_data(), compute(), gather_copyin_data(), init_mic_address(), init_static_ptr_data(), nullify_target_stack(), offload_finish(), offload_stack_memory_manager(), receive_pointer_data(), recieve_noncontiguous_pointer_data(), scatter_copyout_data(), send_noncontiguous_pointer_data(), send_pointer_data(), and setup_misc_data().

void OffloadDescriptor::scatter_copyin_data ( )

Definition at line 314 of file offload_target.cpp.
Referenced by OFFLOAD_TARGET_ENTER().

bool OffloadDescriptor::scatter_copyout_data ( ) [private]

Definition at line 3347 of file offload_host.cpp.
Referenced by offload_finish().

bool OffloadDescriptor::send_noncontiguous_pointer_data (int *i,* PtrData * *src_buf,* PtrData * *dst_buf,* COIEVENT * *event*) [private]

Definition at line 2294 of file offload_host.cpp.
Referenced by send_pointer_data().

bool OffloadDescriptor::send_pointer_data (bool *is_async*) [private]

Definition at line 2420 of file offload_host.cpp.
Referenced by offload().

void OffloadDescriptor::set_offload_number (int *number*) [inline]

Definition at line 54 of file offload_target.h.
Referenced by offload().

bool OffloadDescriptor::setup_descriptors (VarDesc * *vars,* VarDesc2 * *vars2,* int *vars_total,* int *entry_id,* const void * *stack_addr*) [private]

Definition at line 689 of file offload_host.cpp.
Referenced by offload().

bool OffloadDescriptor::setup_misc_data (const char * *name*) [private]

Definition at line 1966 of file offload_host.cpp.
Referenced by offload().

_Offload_result OffloadDescriptor::translate_coi_error (COIRESULT *res*) const [private]

Definition at line 244 of file offload_host.cpp.

Referenced by alloc_ptr_data(), compute(), gather_copyin_data(), init_mic_address(), init_static_ptr_data(), nullify_target_stack(), offload_finish(), offload_stack_memory_manager(), receive_pointer_data(), recieve_noncontiguous_pointer_data(), scatter_copyout_data(), send_noncontiguous_pointer_data(), send_pointer_data(), and setup_misc_data().

bool OffloadDescriptor::wait_dependencies (const void ** *waits,* int *num_waits*) [private]

Definition at line 2049 of file offload_host.cpp.

Referenced by offload().

6.57.5 Member Data Documentation

BufferList OffloadDescriptor::m_buffers [private]

Definition at line 73 of file offload_target.h.

Referenced by offload(), and scatter_copyin_data().

BufferList OffloadDescriptor::m_compute_buffers [private]

Definition at line 226 of file offload_host.h.

Referenced by compute(), setup_descriptors(), and setup_misc_data().

BufferList OffloadDescriptor::m_destroy_buffers [private]

Definition at line 229 of file offload_host.h.

Referenced by offload_finish(), receive_pointer_data(), and setup_misc_data().

PtrDataList OffloadDescriptor::m_destroy_stack [private]

Definition at line 208 of file offload_host.h.

Referenced by gather_copyin_data(), offload_stack_memory_manager(), receive_pointer_data(), and setup_descriptors().

Engine& OffloadDescriptor::m_device [private]

Definition at line 211 of file offload_host.h.

Referenced by alloc_ptr_data(), cleanup(), compute(), find_ptr_data(), init_static_ptr_data(), offload(), offload_stack_memory_manager(), receive_pointer_data(), report_coi_error(), setup_descriptors(), setup_misc_data(), and wait_dependencies().

FunctionDescriptor* OffloadDescriptor::m_func_desc [private]

Definition at line 240 of file offload_host.h.

Referenced by compute(), gather_copyin_data(), scatter_copyout_data(), setup_misc_data(), and ~OffloadDescriptor().

uint32_t OffloadDescriptor::m_func_desc_size [private]

Definition at line 241 of file offload_host.h.

Referenced by compute(), and setup_misc_data().

Marshaller OffloadDescriptor::m_in [private]

Definition at line 220 of file offload_host.h.

Referenced by gather_copyin_data(), offload(), and scatter_copyin_data().

uint64_t OffloadDescriptor::m_in_datalen [private]

Definition at line 256 of file offload_host.h.

Referenced by compute(), gather_copyin_data(), offload_stack_memory_manager(), setup_descriptors(), and setup_misc_data().

COIEVENT* OffloadDescriptor::m_in_deps [private]

Definition at line 247 of file offload_host.h.

Referenced by compute(), gen_var_descs_for_pointer_array(), is_signaled(), offload_finish(), receive_pointer_data(), recieve_noncontiguous_pointer_data(), send_pointer_data(), setup_descriptors(), and ~OffloadDescriptor().

uint32_t OffloadDescriptor::m_in_deps_total [private]

Definition at line 248 of file offload_host.h.

Referenced by compute(), is_signaled(), offload_finish(), receive_pointer_data(), recieve_noncontiguous_pointer_data(), and send_pointer_data().

COIBUFFER OffloadDescriptor::m_inout_buf [private]

Definition at line 244 of file offload_host.h.

Referenced by gather_copyin_data(), scatter_copyout_data(), and setup_misc_data().

bool OffloadDescriptor::m_is_mandatory [private]

Definition at line 214 of file offload_host.h.

Referenced by alloc_ptr_data(), init_mic_address(), and offload_stack_memory_manager().

const bool OffloadDescriptor::m_is_openmp [private]

Definition at line 217 of file offload_host.h.

Referenced by receive_pointer_data(), and setup_descriptors().

bool OffloadDescriptor::m_need_runfunction [private]

Definition at line 261 of file offload_host.h.

Referenced by compute(), gather_copyin_data(), offload(), scatter_copyout_data(), setup_descriptors(), and setup_misc_data().

int OffloadDescriptor::m_offload_number [private]

Definition at line 78 of file offload_target.h.

Referenced by get_offload_number(), and set_offload_number().

Marshaller OffloadDescriptor::m_out [private]

Definition at line 223 of file offload_host.h.

Referenced by gather_copyout_data(), offload(), and scatter_copyout_data().

uint64_t OffloadDescriptor::m_out_datalen [private]

Definition at line 257 of file offload_host.h.

Referenced by compute(), scatter_copyout_data(), setup_descriptors(), and setup_misc_data().

COIEVENT* OffloadDescriptor::m_out_deps [private]

Definition at line 249 of file offload_host.h.

Referenced by gen_var_descs_for_pointer_array(), is_signaled(), offload_finish(), receive_pointer_data(), setup_descriptors(), and ~OffloadDescriptor().

uint32_t OffloadDescriptor::m_out_deps_total [private]

Definition at line 250 of file offload_host.h.
Referenced by is_signaled(), offload_finish(), and receive_pointer_data().

PtrData* OffloadDescriptor::m_stack_ptr_data [private]

Definition at line 207 of file offload_host.h.
Referenced by offload_stack_memory_manager(), and setup_descriptors().

_Offload_status* OffloadDescriptor::m_status [private]

Definition at line 237 of file offload_host.h.
Referenced by alloc_ptr_data(), compute(), gather_copyin_data(), init_mic_address(), init_static_ptr_data(), nullify_target_stack(), offload(), offload_finish(), offload_stack_memory_manager(), receive_pointer_data(), recieve_noncontiguous_pointer_data(), scatter_copyout_data(), send_noncontiguous_pointer_data(), send_pointer_data(), and setup_misc_data().

OffloadHostTimerData* OffloadDescriptor::m_timer_data [private]

Definition at line 253 of file offload_host.h.
Referenced by get_timer_data().

VarDesc * OffloadDescriptor::m_vars [private]

Definition at line 232 of file offload_host.h.
Referenced by gather_copyin_data(), gather_copyout_data(), gen_var_descs_for_pointer_array(), merge_var_descs(), offload(), receive_pointer_data(), recieve_noncontiguous_pointer_data(), scatter_copyin_data(), scatter_copyout_data(), send_noncontiguous_pointer_data(), send_pointer_data(), setup_descriptors(), and ~OffloadDescriptor().

VarExtra* OffloadDescriptor::m_vars_extra [private]

Definition at line 233 of file offload_host.h.
Referenced by gather_copyin_data(), gen_var_descs_for_pointer_array(), receive_pointer_data(), recieve_noncontiguous_pointer_data(), scatter_copyout_data(), send_noncontiguous_pointer_data(), send_pointer_data(), setup_descriptors(), and ~OffloadDescriptor().

int OffloadDescriptor::m_vars_total [private]

Definition at line 234 of file offload_host.h.
Referenced by gather_copyin_data(), gather_copyout_data(), gen_var_descs_for_pointer_array(), merge_var_descs(), offload(), receive_pointer_data(), scatter_copyin_data(), scatter_copyout_data(), send_pointer_data(), setup_descriptors(), and setup_misc_data().
The documentation for this class was generated from the following files:

- offload_host.h
- offload_target.h
- offload_host.cpp
- offload_target.cpp

6.58 mic_lib::omp_destroy_lock_target Interface Reference

Public Member Functions

- subroutine omp_destroy_lock_target (target_type, target_number, lock)

6.58.1 Detailed Description

Definition at line 175 of file mic_lib.f90.

6.58.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_destroy_lock_target::omp_destroy_lock_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_intptr_t) *lock*)

Definition at line 175 of file mic_lib.f90.

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

- mic_lib.f90

6.59 mic_lib::omp_destroy_nest_lock_target Interface Reference

Public Member Functions

- subroutine omp_destroy_nest_lock_target (target_type, target_number, lock)

6.59.1 Detailed Description

Definition at line 226 of file mic_lib.f90.

6.59.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_destroy_nest_lock_target::omp_destroy_nest_lock_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_intptr_t) *lock*)

Definition at line 226 of file mic_lib.f90.

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

- mic_lib.f90

6.60 mic_lib::omp_get_dynamic_target Interface Reference

Public Member Functions

- integer(kind=c_int) function omp_get_dynamic_target (target_type, target_number)

6.60.1 Detailed Description

Definition at line 120 of file mic_lib.f90.

6.60.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::omp_get_dynamic_target::omp_get_dynamic_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 120 of file mic_lib.f90.

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

- mic_lib.f90

6.61 mic_lib::omp_get_max_threads_target Interface Reference

Public Member Functions

- integer(kind=c_int) function omp_get_max_threads_target (target_type, target_number)

6.61.1 Detailed Description

Definition at line 96 of file mic_lib.f90.

6.61.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::omp_get_max_threads_target::omp_get_max_threads_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 96 of file mic_lib.f90.

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

- mic_lib.f90

6.62 mic_lib::omp_get_nested_target Interface Reference

Public Member Functions

- integer(kind=c_int) function omp_get_nested_target (target_type, target_number)

6.62.1 Detailed Description

Definition at line 136 of file mic_lib.f90.

6.62.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::omp_get_nested_target::omp_get_nested_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 136 of file mic_lib.f90.

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

- mic_lib.f90

6.63 mic_lib::omp_get_num_procs_target Interface Reference

Public Member Functions

- integer(kind=c_int) function omp_get_num_procs_target (target_type, target_number)

6.63.1 Detailed Description

Definition at line 104 of file mic_lib.f90.

6.63.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::omp_get_num_procs_target::omp_get_num_procs_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number*)

Definition at line 104 of file mic_lib.f90.

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

- mic_lib.f90

6.64 mic_lib::omp_get_schedule_target Interface Reference

Public Member Functions

- subroutine omp_get_schedule_target (target_type, target_number, kind, modifier)

6.64.1 Detailed Description

Definition at line 153 of file mic_lib.f90.

6.64.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_get_schedule_target::omp_get_schedule_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_intptr_t) *kind,* integer (kind=c_intptr_t) *modifier*)

Definition at line 153 of file mic_lib.f90.

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

- mic_lib.f90

6.65 mic_lib::omp_init_lock_target Interface Reference

Public Member Functions

- subroutine omp_init_lock_target (target_type, target_number, lock)

6.65.1 Detailed Description

Definition at line 165 of file mic_lib.f90.

6.65.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_init_lock_target::omp_init_lock_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_intptr_t) *lock*)

Definition at line 165 of file mic_lib.f90.

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

- mic_lib.f90

6.66 mic_lib::omp_init_nest_lock_target Interface Reference

Public Member Functions

- subroutine omp_init_nest_lock_target (target_type, target_number, lock)

6.66.1 Detailed Description

Definition at line 217 of file mic_lib.f90.

6.66.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_init_nest_lock_target::omp_init_nest_lock_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_intptr_t) *lock*)

Definition at line 217 of file mic_lib.f90.

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

- mic_lib.f90

6.67 omp_lock_target_t Struct Reference

`#include <offload.h>`

Public Attributes

- omp_lock_t lock

6.67.1 Detailed Description

Definition at line 197 of file offload.h.

6.67.2 Member Data Documentation

omp_lock_t omp_lock_target_t::lock

Definition at line 198 of file offload.h.

Referenced by omp_destroy_lock_lrb(), omp_init_lock_lrb(), omp_set_lock_lrb(), omp_test_lock_lrb(), and omp_unset_lock_lrb().

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

- offload.h

6.68 omp_nest_lock_target_t Struct Reference

`#include <offload.h>`

Public Attributes

- omp_nest_lock_t lock

6.68.1 Detailed Description

Definition at line 233 of file offload.h.

6.68.2 Member Data Documentation

omp_nest_lock_t omp_nest_lock_target_t::lock

Definition at line 234 of file offload.h.

Referenced by omp_destroy_nest_lock_lrb(), omp_init_nest_lock_lrb(), omp_set_nest_lock_lrb(), omp_test_nest_lock_lrb(), and omp_unset_nest_lock_lrb().

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

- offload.h

6.69 mic_lib::omp_set_dynamic_target Interface Reference

Public Member Functions

- subroutine omp_set_dynamic_target (target_type, target_number, num_threads)

6.69.1 Detailed Description

Definition at line 112 of file mic_lib.f90.

6.69.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_set_dynamic_target::omp_set_dynamic_target (integer (kind=c_int) *target_type,* integer (kind=c_int) *target_number,* integer (kind=c_int) *num_threads*)

Definition at line 112 of file mic_lib.f90.

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

- mic_lib.f90

6.70 mic_lib::omp_set_lock_target Interface Reference

Public Member Functions

- subroutine omp_set_lock_target (target_type, target_number, lock)

6.70.1 Detailed Description

Definition at line 185 of file mic_lib.f90.

6.70.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_set_lock_target::omp_set_lock_target (integer (kind=c_int) ***target_type,*** **integer (kind=c_int)** ***target_number,*** **integer (kind=c_intptr_t)** ***lock*** **)**

Definition at line 185 of file mic_lib.f90.

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

- mic_lib.f90

6.71 mic_lib::omp_set_nest_lock_target Interface Reference

Public Member Functions

- subroutine omp_set_nest_lock_target (target_type, target_number, lock)

6.71.1 Detailed Description

Definition at line 235 of file mic_lib.f90.

6.71.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_set_nest_lock_target::omp_set_nest_lock_target (integer (kind=c_int) ***target_type,*** **integer (kind=c_int)** ***target_number,*** **integer (kind=c_intptr_t)** ***lock*** **)**

Definition at line 235 of file mic_lib.f90.

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

- mic_lib.f90

6.72 mic_lib::omp_set_nested_target Interface Reference

Public Member Functions

- subroutine omp_set_nested_target (target_type, target_number, nested)

6.72.1 Detailed Description

Definition at line 128 of file mic_lib.f90.

6.72.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_set_nested_target::omp_set_nested_target (integer (kind=c_int) ***target_type,*** **integer (kind=c_int)** ***target_number,*** **integer (kind=c_int)** ***nested*** **)**

Definition at line 128 of file mic_lib.f90.

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

- mic_lib.f90

6.73 mic_lib::omp_set_num_threads_target Interface Reference

Public Member Functions

- subroutine omp_set_num_threads_target (target_type, target_number, num_threads)

6.73.1 Detailed Description

Definition at line 88 of file mic_lib.f90.

6.73.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_set_num_threads_target::omp_set_num_threads_target (integer (kind=c_int) ***target_type,*** **integer (kind=c_int)** ***target_number,*** **integer (kind=c_int)** ***num_threads*** **)**

Definition at line 88 of file mic_lib.f90.

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

- mic_lib.f90

6.74 mic_lib::omp_set_schedule_target Interface Reference

Public Member Functions

- subroutine omp_set_schedule_target (target_type, target_number, kind, modifier)

6.74.1 Detailed Description

Definition at line 144 of file mic_lib.f90.

6.74.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_set_schedule_target::omp_set_schedule_target (integer (kind=c_int) ***target_type,*** **integer (kind=c_int)** ***target_number,*** **integer (kind=c_int)** ***kind,*** **integer (kind=c_int)** ***modifier*** **)**

Definition at line 144 of file mic_lib.f90.

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

- mic_lib.f90

6.75 mic_lib::omp_test_lock_target Interface Reference

Public Member Functions

- integer(kind=c_int) function omp_test_lock_target (target_type, target_number, lock)

6.75.1 Detailed Description

Definition at line 205 of file mic_lib.f90.

6.75.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::omp_test_lock_target::omp_test_lock_target (integer (kind=c_int) ***target_type,*** **integer (kind=c_int)** ***target_number,*** **integer (kind=c_intptr_t)** ***lock*** **)**

Definition at line 205 of file mic_lib.f90.

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

- mic_lib.f90

6.76 mic_lib::omp_test_nest_lock_target Interface Reference

Public Member Functions

- integer(kind=c_int) function omp_test_nest_lock_target (target_type, target_number, lock)

6.76.1 Detailed Description

Definition at line 253 of file mic_lib.f90.

6.76.2 Constructor & Destructor Documentation

integer (kind=c_int) function mic_lib::omp_test_nest_lock_target::omp_test_nest_lock_target (integer (kind=c_int) ***target_type,*** **integer (kind=c_int)** ***target_number,*** **integer (kind=c_intptr_t)** ***lock*** **)**

Definition at line 253 of file mic_lib.f90.

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

- mic_lib.f90

6.77 mic_lib::omp_unset_lock_target Interface Reference

Public Member Functions

- subroutine omp_unset_lock_target (target_type, target_number, lock)

6.77.1 Detailed Description

Definition at line 195 of file mic_lib.f90.

6.77.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_unset_lock_target::omp_unset_lock_target (integer (kind=c_int) ***target_type,*** **integer (kind=c_int)** ***target_number,*** **integer (kind=c_intptr_t)** ***lock*** **)**

Definition at line 195 of file mic_lib.f90.

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

- mic_lib.f90

6.78 mic_lib::omp_unset_nest_lock_target Interface Reference

Public Member Functions

- subroutine omp_unset_nest_lock_target (target_type, target_number, lock)

6.78.1 Detailed Description

Definition at line 244 of file mic_lib.f90.

6.78.2 Constructor & Destructor Documentation

subroutine mic_lib::omp_unset_nest_lock_target::omp_unset_nest_lock_target (integer (kind=c_int) ***target_type,*** **integer (kind=c_int)** ***target_number,*** **integer (kind=c_intptr_t)** ***lock*** **)**

Definition at line 244 of file mic_lib.f90.

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

- mic_lib.f90

6.79 ORSLBusySet Struct Reference

`#include <orsl-lite/include/orsl-lite.h>`

Public Attributes

- BusySetType type

6.79.1 Detailed Description

ORSLBusySet encapsulation
Definition at line 32 of file orsl-lite.h.

6.79.2 Member Data Documentation

BusySetType ORSLBusySet::type

Set type
Definition at line 33 of file orsl-lite.h.
Referenced by ORSL::release(), ORSL::reserve(), and ORSL::try_reserve().
The documentation for this struct was generated from the following file:

- orsl-lite/include/orsl-lite.h

6.80 PersistData Struct Reference

`#include <offload_engine.h>`

Public Member Functions

- PersistData (const void *addr, uint64_t routine_num, uint64_t size)

Public Attributes

- const void * stack_cpu_addr
- uint64_t routine_id
- PtrData * stack_ptr_data
- char * cpu_stack_addr

6.80.1 Detailed Description

Definition at line 210 of file offload_engine.h.

6.80.2 Constructor & Destructor Documentation

PersistData::PersistData (const void * *addr,* uint64_t *routine_num,* uint64_t *size*) [inline]

Definition at line 212 of file offload_engine.h.

6.80.3 Member Data Documentation

char* PersistData::cpu_stack_addr

Definition at line 224 of file offload_engine.h.

uint64_t PersistData::routine_id

Definition at line 220 of file offload_engine.h.

const void* PersistData::stack_cpu_addr

Definition at line 218 of file offload_engine.h.

PtrData* PersistData::stack_ptr_data

Definition at line 222 of file offload_engine.h.
Referenced by OffloadDescriptor::offload_stack_memory_manager(), and PersistData().
The documentation for this struct was generated from the following file:

- offload_engine.h

6.81 PtrData Class Reference

`#include <offload_engine.h>`

Public Member Functions

- PtrData (const void *addr, uint64_t len)
- PtrData (const PtrData &ptr)
- bool operator< (const PtrData &o) const
- long add_reference ()
- long remove_reference ()
- long get_reference () const

Public Attributes

- const MemRange cpu_addr
- COIBUFFER cpu_buf
- COIBUFFER mic_buf
- uint64_t mic_addr
- uint64_t alloc_disp
- uint32_t mic_offset
- bool is_static
- mutex_t alloc_ptr_data_lock

Private Attributes

- long ref_count

6.81.1 Detailed Description

Definition at line 58 of file offload_engine.h.

6.81.2 Constructor & Destructor Documentation

PtrData::PtrData (const void * *addr,* uint64_t *len*) [inline]

Definition at line 60 of file offload_engine.h.

PtrData::PtrData (const PtrData & *ptr*) [inline]

Definition at line 69 of file offload_engine.h.

6.81.3 Member Function Documentation

long PtrData::add_reference ( ) [inline]

Definition at line 83 of file offload_engine.h.
Referenced by OffloadDescriptor::setup_descriptors().

long PtrData::get_reference ( ) const [inline]

Definition at line 105 of file offload_engine.h.
Referenced by OffloadDescriptor::setup_descriptors().

bool PtrData::operator< (const PtrData & *o*) const [inline]

Definition at line 76 of file offload_engine.h.

long PtrData::remove_reference ( ) [inline]

Definition at line 94 of file offload_engine.h.
Referenced by OffloadDescriptor::receive_pointer_data().

6.81.4 Member Data Documentation

uint64_t PtrData::alloc_disp

Definition at line 123 of file offload_engine.h.
Referenced by OffloadDescriptor::alloc_ptr_data(), OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

mutex_t PtrData::alloc_ptr_data_lock

Definition at line 131 of file offload_engine.h.
Referenced by OffloadDescriptor::alloc_ptr_data(), and Engine::insert_ptr_data().

const MemRange PtrData::cpu_addr

Definition at line 114 of file offload_engine.h.
Referenced by OffloadDescriptor::alloc_ptr_data(), OffloadDescriptor::find_ptr_data(), OffloadDescriptor::init_static_ptr_data(), operator<(), OffloadDescriptor::receive_pointer_data(), and OffloadDescriptor::setup_descriptors().

COIBUFFER PtrData::cpu_buf

Definition at line 117 of file offload_engine.h.
Referenced by OffloadDescriptor::alloc_ptr_data(), OffloadDescriptor::init_static_ptr_data(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), and OffloadDescriptor::send_pointer_data().

bool PtrData::is_static

Definition at line 130 of file offload_engine.h.
Referenced by add_reference(), OffloadDescriptor::alloc_ptr_data(), OffloadDescriptor::find_ptr_data(), get_reference(), Engine::init_ptr_data(), remove_reference(), and OffloadDescriptor::setup_descriptors().

uint64_t PtrData::mic_addr

Definition at line 121 of file offload_engine.h.
Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::init_mic_address(), Engine::init_ptr_data(), OffloadDescriptor::init_static_ptr_data(), OffloadDescriptor::offload_stack_memory_manager(), OffloadDescriptor::receive_pointer_data(), and OffloadDescriptor::setup_descriptors().

COIBUFFER PtrData::mic_buf

Definition at line 118 of file offload_engine.h.

Referenced by OffloadDescriptor::alloc_ptr_data(), OffloadDescriptor::init_mic_address(), OffloadDescriptor::init_static_ptr_data(), OffloadDescriptor::offload_stack_memory_manager(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

uint32_t PtrData::mic_offset

Definition at line 127 of file offload_engine.h.

Referenced by OffloadDescriptor::alloc_ptr_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

long PtrData::ref_count `[private]`

Definition at line 135 of file offload_engine.h.

Referenced by add_reference(), get_reference(), and remove_reference().

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

- offload_engine.h

6.82 OffloadDescriptor::ReadArrElements< T > Class Template Reference

Public Member Functions

- ReadArrElements ()
- bool read_next (bool flag)

Public Attributes

- CeanReadRanges * ranges
- T val
- int el_size
- int64_t size
- int64_t offset
- int64_t length_cur
- bool is_empty
- int count
- char * base

6.82.1 Detailed Description

template<typename T>class OffloadDescriptor::ReadArrElements< T >

Definition at line 156 of file offload_host.h.

6.82.2 Constructor & Destructor Documentation

template<typename T> OffloadDescriptor::ReadArrElements< T >::ReadArrElements ( ) `[inline]`

Definition at line 158 of file offload_host.h.

6.82.3 Member Function Documentation

template<typename T> bool OffloadDescriptor::ReadArrElements< T >::read_next (bool *flag*) [inline]

Definition at line 167 of file offload_host.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array().

6.82.4 Member Data Documentation

template<typename T> char* OffloadDescriptor::ReadArrElements< T >::base

Definition at line 203 of file offload_host.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::ReadArrElements< T >::read_next().

template<typename T> int OffloadDescriptor::ReadArrElements< T >::count

Definition at line 202 of file offload_host.h.

Referenced by OffloadDescriptor::ReadArrElements< T >::read_next().

template<typename T> int OffloadDescriptor::ReadArrElements< T >::el_size

Definition at line 197 of file offload_host.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::ReadArrElements< T >::read_next().

template<typename T> bool OffloadDescriptor::ReadArrElements< T >::is_empty

Definition at line 201 of file offload_host.h.

Referenced by OffloadDescriptor::ReadArrElements< T >::read_next().

template<typename T> int64_t OffloadDescriptor::ReadArrElements< T >::length_cur

Definition at line 198 of file offload_host.h.

Referenced by OffloadDescriptor::ReadArrElements< T >::read_next().

template<typename T> int64_t OffloadDescriptor::ReadArrElements< T >::offset

Definition at line 198 of file offload_host.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::ReadArrElements< T >::read_next().

template<typename T> CeanReadRanges* OffloadDescriptor::ReadArrElements< T >::ranges

Definition at line 195 of file offload_host.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::ReadArrElements< T >::read_next().

template<typename T> int64_t OffloadDescriptor::ReadArrElements< T >::size

Definition at line 198 of file offload_host.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::ReadArrElements< T >::read_next().

template<typename T> T OffloadDescriptor::ReadArrElements< T >::val

Definition at line 196 of file offload_host.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::ReadArrElements< T >::read_next().

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

- offload_host.h

6.83 RefInfo Struct Reference

`#include <offload_target.h>`

Public Member Functions

- RefInfo (bool is_add, long amount)

Public Attributes

- bool is_added
- long count

6.83.1 Detailed Description

Definition at line 93 of file offload_target.h.

6.83.2 Constructor & Destructor Documentation

RefInfo::RefInfo (bool *is_add,* long *amount*) [inline]

Definition at line 94 of file offload_target.h.

6.83.3 Member Data Documentation

long RefInfo::count

Definition at line 97 of file offload_target.h.

bool RefInfo::is_added

Definition at line 96 of file offload_target.h.

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

- offload_target.h

6.84 TableList< T > Class Template Reference

`#include <offload_table.h>`

Classes

- struct Node

Public Types

- typedef T Table

Public Member Functions

- TableList (Node *node=0)
- void add_table (Node *node)
- void remove_table (Node *node)

Protected Attributes

- Node * m_head
- mutex_t m_lock

6.84.1 Detailed Description

template<typename T>class TableList< T >

Definition at line 22 of file offload_table.h.

6.84.2 Member Typedef Documentation

template<typename T> typedef T TableList< T >::Table

Definition at line 25 of file offload_table.h.

6.84.3 Constructor & Destructor Documentation

template<typename T> TableList< T >::TableList (Node * *node = 0*) [inline], [explicit]

Definition at line 35 of file offload_table.h.

6.84.4 Member Function Documentation

template<typename T> void TableList< T >::add_table (Node * *node*) [inline]

Definition at line 37 of file offload_table.h.
Referenced by __offload_register_tables(), and FuncList::add_table().

template<typename T> void TableList< T >::remove_table (Node * *node*) [inline]

Definition at line 49 of file offload_table.h.
Referenced by __offload_unregister_tables().

6.84.5 Member Data Documentation

template<typename T> Node* TableList< T >::m_head [protected]

Definition at line 66 of file offload_table.h.
Referenced by TableList< VarTable >::add_table(), and TableList< VarTable >::remove_table().

template<typename T> mutex_t TableList< T >::m_lock [protected]

Definition at line 67 of file offload_table.h.
Referenced by TableList< VarTable >::add_table(), and TableList< VarTable >::remove_table().
The documentation for this class was generated from the following file:

- offload_table.h

6.85 TargetImage Struct Reference

`#include <offload_engine.h>`

Public Member Functions

- TargetImage (const char *_name, const void *_data, uint64_t _size, const char *_origin, uint64_t _offset)

Public Attributes

- const char * name
- const void * data
- uint64_t size
- const char * origin
- uint64_t offset

6.85.1 Detailed Description

Definition at line 187 of file offload_engine.h.

6.85.2 Constructor & Destructor Documentation

TargetImage::TargetImage (const char * *_name*, const void * *_data*, uint64_t *_size*, const char * *_origin*, uint64_t *_offset*) [inline]

Definition at line 189 of file offload_engine.h.

6.85.3 Member Data Documentation

const void* TargetImage::data

Definition at line 199 of file offload_engine.h.
Referenced by Engine::init_process().

const char* TargetImage::name

Definition at line 196 of file offload_engine.h.
Referenced by Engine::init_process().

uint64_t TargetImage::offset

Definition at line 204 of file offload_engine.h.
Referenced by Engine::init_process().

const char* TargetImage::origin

Definition at line 203 of file offload_engine.h.
Referenced by Engine::init_process().

uint64_t TargetImage::size

Definition at line 200 of file offload_engine.h.
Referenced by Engine::init_process().
The documentation for this struct was generated from the following file:

- offload_engine.h

6.86 Thread Struct Reference

Public Member Functions

- Thread (long *addr_coipipe_counter)
- ~Thread ()
- COIPIPELINE get_pipeline (int index) const
- void set_pipeline (int index, COIPIPELINE pipeline)
- AutoSet & get_auto_vars ()

Private Attributes

- long * m_addr_coipipe_counter
- AutoSet m_auto_vars
- COIPIPELINE m_pipelines [MIC_ENGINES_MAX]

6.86.1 Detailed Description

Definition at line 448 of file offload_engine.cpp.

6.86.2 Constructor & Destructor Documentation

Thread::Thread (long * *addr_coipipe_counter*) [inline]

Definition at line 449 of file offload_engine.cpp.

Thread::~Thread ( ) [inline]

Definition at line 454 of file offload_engine.cpp.

6.86.3 Member Function Documentation

AutoSet& Thread::get_auto_vars ( ) [inline]

Definition at line 475 of file offload_engine.cpp.
Referenced by Engine::get_auto_vars().

COIPIPELINE Thread::get_pipeline (int *index*) const [inline]

Definition at line 467 of file offload_engine.cpp.
Referenced by Engine::get_pipeline().

void Thread::set_pipeline (int *index,* COIPIPELINE *pipeline*) [inline]

Definition at line 471 of file offload_engine.cpp.
Referenced by Engine::get_pipeline().

6.86.4 Member Data Documentation

long* Thread::m_addr_coipipe_counter [private]

Definition at line 480 of file offload_engine.cpp.
Referenced by Thread(), and ~Thread().

AutoSet Thread::m_auto_vars [private]

Definition at line 481 of file offload_engine.cpp.
Referenced by get_auto_vars().

COIPIPELINE Thread::m_pipelines[MIC_ENGINES_MAX] [private]

Definition at line 482 of file offload_engine.cpp.
Referenced by get_pipeline(), set_pipeline(), Thread(), and ~Thread().
The documentation for this struct was generated from the following file:

- offload_engine.cpp

6.87 VarDesc Struct Reference

An Offload Variable descriptor.
`#include <offload_common.h>`

Public Attributes

- union {
  struct {
  uint8_t dst: 4
  OffloadItemType of destination.
  uint8_t src: 4
  OffloadItemType of source.
  }
  uint8_t bits
  } type

  OffloadItemTypes of source and destination.
- union {
  struct {
  uint8_t in: 1
  Set if IN or INOUT.
  uint8_t out: 1
  Set if OUT or INOUT.
  }
  uint8_t bits
  } direction

  OffloadParameterType that describes direction of data transfer.
- uint8_t alloc_if

  alloc_if modifier value
- uint8_t free_if

  free_if modifier value
- uint32_t align
- uint32_t mic_offset

  Not used by compiler; set to 0.
- union {
  struct {
  uint32_t is_static: 1
  source variable has persistent storage
  uint32_t is_static_dstn: 1
  destination variable has persistent storage
  uint32_t has_length: 1
  has length for c_dv && c_dv_ptr
  uint32_t is_stack_buf: 1
  persisted local scalar is in stack buffer
  uint32_t sink_addr: 1
  buffer address is sent in data
  uint32_t alloc_disp: 1

alloc displacement is sent in data
uint32_t is_noncont_src: 1
source data is noncontiguous
uint32_t is_noncont_dst: 1
destination data is noncontiguous
}
uint32_t bits
} flags

Flags describing this variable.

- int64_t offset

  Not used by compiler; set to 0.

- int64_t size

  Element byte-size of data to be transferred.

- union {
  int64_t count
  Set to 0 for array expressions and dope-vectors.
  int64_t disp
  Displacement not used by compiler.
  };

- union {
  void * alloc
  int64_t ptr_arr_offset
  };

  This field not used by OpenMP 4.0.

- void * into

  This field not used by OpenMP 4.0.

- void * ptr

  For an ordinary variable, address of the variable.

6.87.1 Detailed Description

An Offload Variable descriptor.
Definition at line 176 of file offload_common.h.

6.87.2 Member Data Documentation

union { **...** }

union { **...** }

This field not used by OpenMP 4.0.
The alloc section expression in #pragma offload

uint32_t VarDesc::align

MIC alignment requested for pointer data
Definition at line 197 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), OffloadDescriptor::merge_var_descs(), and OffloadDescriptor::setup_descriptors().

void* VarDesc::alloc

Definition at line 241 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

uint32_t VarDesc::alloc_disp

alloc displacement is sent in data

Definition at line 215 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::setup_descriptors().

uint8_t VarDesc::alloc_if

alloc_if modifier value

Definition at line 195 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), kmp_set_defaults_target(), OffloadDescriptor::merge_var_descs(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::setup_descriptors().

uint8_t VarDesc::bits

Definition at line 183 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), kmp_create_affinity_mask_lrb(), kmp_create_affinity_mask_target(), kmp_destroy_affinity_mask_lrb(), kmp_destroy_affinity_mask_target(), kmp_get_affinity_lrb(), kmp_get_affinity_mask_proc_lrb(), kmp_get_affinity_mask_proc_target(), kmp_get_affinity_target(), kmp_set_affinity_lrb(), kmp_set_affinity_mask_proc_lrb(), kmp_set_affinity_mask_proc_target(), kmp_set_affinity_target(), kmp_set_defaults_lrb(), kmp_set_defaults_target(), kmp_unset_affinity_mask_proc_lrb(), kmp_unset_affinity_mask_proc_target(), OffloadDescriptor::merge_var_descs(), omp_destroy_lock_lrb(), omp_destroy_lock_target(), omp_destroy_nest_lock_lrb(), omp_destroy_nest_lock_target(), omp_get_int_from_host(), omp_get_int_target(), omp_get_schedule_lrb(), omp_get_schedule_target(), omp_init_lock_lrb(), omp_init_lock_target(), omp_init_nest_lock_lrb(), omp_init_nest_lock_target(), omp_send_int_to_host(), omp_set_int_target(), omp_set_lock_lrb(), omp_set_lock_target(), omp_set_nest_lock_lrb(), omp_set_nest_lock_target(), omp_set_schedule_lrb(), omp_set_schedule_target(), omp_test_lock_lrb(), omp_test_lock_target(), omp_test_nest_lock_lrb(), omp_test_nest_lock_target(), omp_unset_lock_lrb(), omp_unset_lock_target(), omp_unset_nest_lock_lrb(), omp_unset_nest_lock_target(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::setup_descriptors().

uint32_t VarDesc::bits

Definition at line 221 of file offload_common.h.

int64_t VarDesc::count

Set to 0 for array expressions and dope-vectors.

Set to 1 for scalars

Set to value of length modifier for pointers

Definition at line 233 of file offload_common.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), kmp_create_affinity_mask_target(), kmp_destroy_affinity_mask_target(), kmp_get_affinity_mask_proc_target(), kmp_get_affinity_target(), kmp_set_affinity_mask_proc_target(), kmp_set_affinity_target(), kmp_unset_affinity_mask_proc_target(), OffloadDescriptor::merge_var_descs(), omp_destroy_lock_target(), omp_destroy_nest_lock_target(), omp_get_int_target(), omp_get_schedule_target(), omp_init_lock_target(), omp_init_nest_lock_target(), omp_set_int_target(), omp_set_lock_target(), omp_set_nest_lock_target(), omp_set_schedule_target(), omp_test_lock_target(), omp_test_nest_lock_target(), omp_unset_lock_target(), omp_unset_nest_lock_target(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::setup_descriptors().

union { ... } VarDesc::direction

OffloadParameterType that describes direction of data transfer.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), kmp_create_affinity_mask_lrb(), kmp_create_affinity_mask_target(),

kmp_destroy_affinity_mask_lrb(), kmp_destroy_affinity_mask_target(), kmp_get_affinity_lrb(), kmp_get_affinity_mask_proc_lrb(), kmp_get_affinity_mask_proc_target(), kmp_get_affinity_target(), kmp_set_affinity_lrb(), kmp_set_affinity_mask_proc_lrb(), kmp_set_affinity_mask_proc_target(), kmp_set_affinity_target(), kmp_set_defaults_lrb(), kmp_set_defaults_target(), kmp_unset_affinity_mask_proc_lrb(), kmp_unset_affinity_mask_proc_target(), OffloadDescriptor::merge_var_descs(), omp_destroy_lock_lrb(), omp_destroy_lock_target(), omp_destroy_nest_lock_lrb(), omp_destroy_nest_lock_target(), omp_get_int_from_host(), omp_get_int_target(), omp_get_schedule_lrb(), omp_get_schedule_target(), omp_init_lock_lrb(), omp_init_lock_target(), omp_init_nest_lock_lrb(), omp_init_nest_lock_target(), omp_send_int_to_host(), omp_set_int_target(), omp_set_lock_lrb(), omp_set_lock_target(), omp_set_nest_lock_lrb(), omp_set_nest_lock_target(), omp_set_schedule_lrb(), omp_set_schedule_target(), omp_test_lock_lrb(), omp_test_lock_target(), omp_test_nest_lock_lrb(), omp_test_nest_lock_target(), omp_unset_lock_lrb(), omp_unset_lock_target(), omp_unset_nest_lock_lrb(), omp_unset_nest_lock_target(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::setup_descriptors().

int64_t VarDesc::disp

Displacement not used by compiler.

Definition at line 235 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

uint8_t VarDesc::dst

OffloadItemType of destination.

Definition at line 180 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), kmp_create_affinity_mask_lrb(), kmp_create_affinity_mask_target(), kmp_destroy_affinity_mask_lrb(), kmp_destroy_affinity_mask_target(), kmp_get_affinity_lrb(), kmp_get_affinity_mask_proc_lrb(), kmp_get_affinity_mask_proc_target(), kmp_get_affinity_target(), kmp_set_affinity_lrb(), kmp_set_affinity_mask_proc_lrb(), kmp_set_affinity_mask_proc_target(), kmp_set_affinity_target(), kmp_set_defaults_lrb(), kmp_set_defaults_target(), kmp_unset_affinity_mask_proc_lrb(), kmp_unset_affinity_mask_proc_target(), OffloadDescriptor::merge_var_descs(), omp_destroy_lock_lrb(), omp_destroy_lock_target(), omp_destroy_nest_lock_lrb(), omp_destroy_nest_lock_target(), omp_get_int_from_host(), omp_get_int_target(), omp_get_schedule_lrb(), omp_get_schedule_target(), omp_init_lock_lrb(), omp_init_lock_target(), omp_init_nest_lock_lrb(), omp_init_nest_lock_target(), omp_send_int_to_host(), omp_set_int_target(), omp_set_lock_lrb(), omp_set_lock_target(), omp_set_nest_lock_lrb(), omp_set_nest_lock_target(), omp_set_schedule_lrb(), omp_set_schedule_target(), omp_test_lock_lrb(), omp_test_lock_target(), omp_test_nest_lock_lrb(), omp_test_nest_lock_target(), omp_unset_lock_lrb(), omp_unset_lock_target(), omp_unset_nest_lock_lrb(), omp_unset_nest_lock_target(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::scatter_copyout_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

union { ... } VarDesc::flags

Flags describing this variable.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), OffloadDescriptor::merge_var_descs(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::scatter_copyout_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

uint8_t VarDesc::free_if

free_if modifier value

Definition at line 196 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), kmp_set_defaults_target(), OffloadDescriptor::merge_var_descs(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::setup_descriptors().

uint32_t VarDesc::has_length

has length for c_dv && c_dv_ptr

Definition at line 209 of file offload_common.h.

Referenced by OffloadDescriptor::setup_descriptors().

uint8_t VarDesc::in

Set if IN or INOUT.

Definition at line 189 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

void* VarDesc::into

This field not used by OpenMP 4.0.

The into section expression in #pragma offload

For c_data_ptr_array this is the into ptr array

Definition at line 248 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), OffloadDescriptor::merge_var_descs(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

uint32_t VarDesc::is_noncont_dst

destination data is noncontiguous

Definition at line 219 of file offload_common.h.

Referenced by OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

uint32_t VarDesc::is_noncont_src

source data is noncontiguous

Definition at line 217 of file offload_common.h.

Referenced by OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

uint32_t VarDesc::is_stack_buf

persisted local scalar is in stack buffer

Definition at line 211 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::setup_descriptors().

uint32_t VarDesc::is_static

source variable has persistent storage

Definition at line 205 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::scatter_copyout_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

uint32_t VarDesc::is_static_dstn

destination variable has persistent storage

Definition at line 207 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

uint32_t VarDesc::mic_offset

Not used by compiler; set to 0.

Used by runtime as offset to data from start of MIC buffer

Definition at line 200 of file offload_common.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), OffloadDescriptor::merge_var_descs(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

int64_t VarDesc::offset

Not used by compiler; set to 0.

Used by runtime as offset to base from data stored in a buffer

Definition at line 225 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), OffloadDescriptor::merge_var_descs(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

uint8_t VarDesc::out

Set if OUT or INOUT.

Definition at line 190 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::scatter_copyout_data(), and OffloadDescriptor::setup_descriptors().

void* VarDesc::ptr

For an ordinary variable, address of the variable.

For c_cean_var (C/C++ array expression), pointer to arr_desc, which is an array descriptor.

For c_data_ptr_array (array of data pointers), pointer to ptr_array_descriptor, which is a descriptor for pointer array transfers.

Definition at line 256 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), kmp_create_affinity_mask_lrb(), kmp_create_affinity_mask_target(), kmp_destroy_affinity_mask_lrb(), kmp_destroy_affinity_mask_target(), kmp_get_affinity_lrb(), kmp_get_affinity_mask_proc_lrb(), kmp_get_affinity_mask_proc_target(), kmp_get_affinity_target(), kmp_set_affinity_lrb(), kmp_set_affinity_mask_proc_lrb(), kmp_set_affinity_mask_proc_target(), kmp_set_affinity_target(), kmp_set_defaults_lrb(), kmp_set_defaults_target(), kmp_unset_affinity_mask_proc_lrb(), kmp_unset_affinity_mask_proc_target(), OffloadDescriptor::merge_var_descs(), omp_destroy_lock_lrb(), omp_destroy_lock_target(), omp_destroy_nest_lock_lrb(), omp_destroy_nest_lock_target(), omp_get_int_from_host(), omp_get_int_target(), omp_get_schedule_lrb(), omp_get_schedule_target(), omp_init_lock_lrb(), omp_init_lock_target(), omp_init_nest_lock_lrb(), omp_init_nest_lock_target(), omp_send_int_to_host(), omp_set_int_target(), omp_set_lock_lrb(), omp_set_lock_target(), omp_set_nest_lock_lrb(), omp_set_nest_lock_target(), omp_set_schedule_lrb(), omp_set_schedule_target(), omp_test_lock_lrb(), omp_test_lock_target(), omp_test_nest_lock_lrb(), omp_test_nest_lock_target(), omp_unset_lock_lrb(), omp_unset_lock_target(), omp_unset_nest_lock_lrb(), omp_unset_nest_lock_target(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::setup_descriptors().

int64_t VarDesc::ptr_arr_offset

Definition at line 242 of file offload_common.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::send_pointer_data().

uint32_t VarDesc::sink_addr

buffer address is sent in data

Definition at line 213 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::setup_descriptors().

int64_t VarDesc::size

Element byte-size of data to be transferred.

For dope-vector, the size of the dope-vector

Definition at line 228 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), kmp_create_affinity_mask_target(), kmp_destroy_affinity_mask_target(), kmp_get_affinity_mask_proc_target(), kmp_get_affinity_target(), kmp_set_affinity_mask_proc_target(), kmp_set_affinity_target(), kmp_unset_affinity_mask_proc_target(), OffloadDescriptor::merge_var_descs(), omp_destroy_lock_target(), omp_destroy_nest_lock_target(), omp_get_int_target(), omp_get_schedule_target(), omp_init_lock_target(), omp_init_nest_lock_target(), omp_set_int_target(), omp_set_lock_target(), omp_set_nest_lock_target(), omp_set_schedule_target(), omp_test_lock_target(), omp_test_nest_lock_target(), omp_unset_lock_target(), omp_unset_nest_lock_target(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

uint8_t VarDesc::src

OffloadItemType of source.

Definition at line 181 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), kmp_create_affinity_mask_lrb(), kmp_create_affinity_mask_target(), kmp_destroy_affinity_mask_lrb(), kmp_destroy_affinity_mask_target(), kmp_get_affinity_lrb(), kmp_get_affinity_mask_proc_lrb(), kmp_get_affinity_mask_proc_target(), kmp_get_affinity_target(), kmp_set_affinity_lrb(), kmp_set_affinity_mask_proc_lrb(), kmp_set_affinity_mask_proc_target(), kmp_set_affinity_target(), kmp_set_defaults_lrb(), kmp_set_defaults_target(), kmp_unset_affinity_mask_proc_lrb(), kmp_unset_affinity_mask_proc_target(), OffloadDescriptor::merge_var_descs(), omp_destroy_lock_lrb(), omp_destroy_lock_target(), omp_destroy_nest_lock_lrb(), omp_destroy_nest_lock_target(), omp_get_int_from_host(), omp_get_int_target(), omp_get_schedule_lrb(), omp_get_schedule_target(), omp_init_lock_lrb(), omp_init_lock_target(), omp_init_nest_lock_lrb(), omp_init_nest_lock_target(), omp_send_int_to_host(), omp_set_int_target(), omp_set_lock_lrb(), omp_set_lock_target(), omp_set_nest_lock_lrb(), omp_set_nest_lock_target(), omp_set_schedule_lrb(), omp_set_schedule_target(), omp_test_lock_lrb(), omp_test_lock_target(), omp_test_nest_lock_lrb(), omp_test_nest_lock_target(), omp_unset_lock_lrb(), omp_unset_lock_target(), omp_unset_nest_lock_lrb(), omp_unset_nest_lock_target(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::scatter_copyout_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

union { ... } VarDesc::type

OffloadItemTypes of source and destination.

Referenced by OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), kmp_create_affinity_mask_lrb(), kmp_create_affinity_mask_target(), kmp_destroy_affinity_mask_lrb(), kmp_destroy_affinity_mask_target(), kmp_get_affinity_lrb(), kmp_get_affinity_mask_proc_lrb(), kmp_get_affinity_mask_proc_target(), kmp_get_affinity_target(), kmp_set_affinity_lrb(), kmp_set_affinity_mask_proc_lrb(), kmp_set_affinity_mask_proc_target(), kmp_set_affinity_target(), kmp_set_defaults_lrb(), kmp_set_defaults_target(), kmp_unset_affinity_mask_proc_lrb(), kmp_unset_affinity_mask_proc_target(), OffloadDescriptor::merge_var_descs(), omp_destroy_lock_lrb(), omp_destroy_lock_target(), omp_destroy_nest_lock_lrb(), omp_destroy_nest_lock_target(), omp_get_int_from_host(),

omp_get_int_target(), omp_get_schedule_lrb(), omp_get_schedule_target(), omp_init_lock_lrb(), omp_init_lock_target(), omp_init_nest_lock_lrb(), omp_init_nest_lock_target(), omp_send_int_to_host(), omp_set_int_target(), omp_set_lock_lrb(), omp_set_lock_target(), omp_set_nest_lock_lrb(), omp_set_nest_lock_target(), omp_set_schedule_lrb(), omp_set_schedule_target(), omp_test_lock_lrb(), omp_test_lock_target(), omp_test_nest_lock_lrb(), omp_test_nest_lock_target(), omp_unset_lock_lrb(), omp_unset_lock_target(), omp_unset_nest_lock_lrb(), omp_unset_nest_lock_target(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::setup_descriptors().

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

- offload_common.h

6.88 VarDesc2 Struct Reference

Auxiliary struct used when -g is enabled that holds variable names.

`#include <offload_common.h>`

Public Attributes

- const char * sname

  Source name.
- const char * dname

  Destination name (when "into" is used)

6.88.1 Detailed Description

Auxiliary struct used when -g is enabled that holds variable names.

Definition at line 260 of file offload_common.h.

6.88.2 Member Data Documentation

const char* VarDesc2::dname

Destination name (when "into" is used)

Definition at line 262 of file offload_common.h.

const char* VarDesc2::sname

Source name.

Definition at line 261 of file offload_common.h.

Referenced by OffloadDescriptor::merge_var_descs(), and OffloadDescriptor::setup_descriptors().

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

- offload_common.h

6.89 VarDesc3 Struct Reference

`#include <offload_common.h>`

Public Attributes

- void * ptr_array

  Pointer to arr_desc of array of pointers.
- void * align_array

  Scalar value or pointer to arr_desc.
- void * alloc_if_array

  Scalar value or pointer to arr_desc.
- void * free_if_array

Scalar value or pointer to arr_desc.

- void * extent_start

  Scalar value or pointer to arr_desc.

- void * extent_elements

  Scalar value or pointer to arr_desc.

- void * into_start

  Scalar value or pointer to arr_desc.

- void * into_elements

  Scalar value or pointer to arr_desc.

- void * alloc_start

  Scalar value or pointer to arr_desc.

- void * alloc_elements

  Scalar value or pointer to arr_desc.

- uint32_t array_fields

6.89.1 Detailed Description

When the OffloadItemType is c_data_ptr_array the ptr field of the main descriptor points to this struct.

The type in VarDesc1 merely says c_cean_data_ptr, but the pointer type can be c_data_ptr, c_func_ptr, c_void_ptr, or c_string_ptr. Therefore the actual pointer type is in the flags field of VarDesc3.

If flag_align_is_array/flag_alloc_if_is_array/flag_free_if_is_array is 0 then alignment/alloc_if/free_if are specified in VarDesc1.

If flag_align_is_array/flag_alloc_if_is_array/flag_free_if_is_array is 1 then align_array/alloc_if_array/free_if_array specify the set of alignment/alloc_if/free_if values.

For the other fields, if neither the scalar nor the array flag is set, then that modifier was not specified. If the bits are set they specify which modifier was set and whether it was a scalar or an array expression.

Definition at line 279 of file offload_common.h.

6.89.2 Member Data Documentation

void* VarDesc3::align_array

Scalar value or pointer to arr_desc.

Definition at line 282 of file offload_common.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

void* VarDesc3::alloc_elements

Scalar value or pointer to arr_desc.

Definition at line 290 of file offload_common.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

void* VarDesc3::alloc_if_array

Scalar value or pointer to arr_desc.

Definition at line 283 of file offload_common.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

void* VarDesc3::alloc_start

Scalar value or pointer to arr_desc.

Definition at line 289 of file offload_common.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

uint32_t VarDesc3::array_fields

Flags that describe the pointer type and whether each field is a scalar value or an array expression.
First 6 bits are pointer array element type: c_data_ptr, c_func_ptr, c_void_ptr, c_string_ptr
Then single bits specify:
align_array is an array
alloc_if_array is an array
free_if_array is an array
extent_start is a scalar expression
extent_start is an array expression
extent_elements is a scalar expression
extent_elements is an array expression
into_start is a scalar expression
into_start is an array expression
into_elements is a scalar expression
into_elements is an array expression
alloc_start is a scalar expression
alloc_start is an array expression
alloc_elements is a scalar expression
alloc_elements is an array expression
Definition at line 311 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

void* VarDesc3::extent_elements

Scalar value or pointer to arr_desc.
Definition at line 286 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

void* VarDesc3::extent_start

Scalar value or pointer to arr_desc.
Definition at line 285 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

void* VarDesc3::free_if_array

Scalar value or pointer to arr_desc.
Definition at line 284 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

void* VarDesc3::into_elements

Scalar value or pointer to arr_desc.
Definition at line 288 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

void* VarDesc3::into_start

Scalar value or pointer to arr_desc.
Definition at line 287 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

void* VarDesc3::ptr_array

Pointer to arr_desc of array of pointers.
Definition at line 281 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().
The documentation for this struct was generated from the following file:

- offload_common.h

6.90 OffloadDescriptor::VarExtra Struct Reference

Public Attributes

- PtrData * src_data
- PtrData * dst_data
- AutoData * auto_data
- int64_t cpu_disp
- int64_t cpu_offset
- CeanReadRanges * read_rng_src
- CeanReadRanges * read_rng_dst
- int64_t ptr_arr_offset
- bool is_arr_ptr_el

6.90.1 Detailed Description

Definition at line 144 of file offload_host.h.

6.90.2 Member Data Documentation

AutoData* OffloadDescriptor::VarExtra::auto_data

Definition at line 147 of file offload_host.h.

Referenced by OffloadDescriptor::receive_pointer_data(), and OffloadDescriptor::setup_descriptors().

int64_t OffloadDescriptor::VarExtra::cpu_disp

Definition at line 148 of file offload_host.h.

Referenced by OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

int64_t OffloadDescriptor::VarExtra::cpu_offset

Definition at line 149 of file offload_host.h.

Referenced by OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

PtrData* OffloadDescriptor::VarExtra::dst_data

Definition at line 146 of file offload_host.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

bool OffloadDescriptor::VarExtra::is_arr_ptr_el

Definition at line 153 of file offload_host.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

int64_t OffloadDescriptor::VarExtra::ptr_arr_offset

Definition at line 152 of file offload_host.h.

Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::send_pointer_data().

CeanReadRanges* OffloadDescriptor::VarExtra::read_rng_dst

Definition at line 151 of file offload_host.h.

Referenced by OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), and OffloadDescriptor::setup_descriptors().

CeanReadRanges* OffloadDescriptor::VarExtra::read_rng_src

Definition at line 150 of file offload_host.h.

Referenced by OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), and OffloadDescriptor::setup_descriptors().

PtrData* OffloadDescriptor::VarExtra::src_data

Definition at line 145 of file offload_host.h.

Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

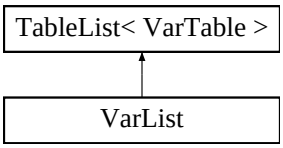
The documentation for this struct was generated from the following file:

- offload_host.h

6.91 VarList Class Reference

`#include <offload_table.h>`

Inheritance diagram for VarList:



Classes

- struct BufEntry
- class Iterator

Public Member Functions

- VarList ()
- void dump ()
- Iterator begin () const
- Iterator end () const
- int64_t table_size (int64_t &nelems)
- void table_copy (void *buf, int64_t nelems)

Static Public Member Functions

- static void table_patch_names (void *buf, int64_t nelems)

Additional Inherited Members

6.91.1 Detailed Description

Definition at line 155 of file offload_table.h.

6.91.2 Constructor & Destructor Documentation

VarList::VarList ( ) [inline]

Definition at line 157 of file offload_table.h.

6.91.3 Member Function Documentation

Iterator VarList::begin ( ) const [inline]

Definition at line 221 of file offload_table.h.
Referenced by Engine::init_ptr_data().

void VarList::dump (void)

Definition at line 264 of file offload_table.cpp.

Iterator VarList::end ( ) const [inline]

Definition at line 225 of file offload_table.h.
Referenced by Engine::init_ptr_data().

void VarList::table_copy (void * *buf,* int64_t *nelems*)

Definition at line 309 of file offload_table.cpp.
Referenced by server_var_table_copy().

void VarList::table_patch_names (void * *buf,* int64_t *nelems*) [static]

Definition at line 335 of file offload_table.cpp.
Referenced by Engine::init_ptr_data().

int64_t VarList::table_size (int64_t & *nelems*)

Definition at line 288 of file offload_table.cpp.
Referenced by server_var_table_size().
The documentation for this class was generated from the following files:

- offload_table.h
- offload_table.cpp

6.92 VarTable Struct Reference

`#include <offload_table.h>`

Classes

- struct Entry

  Variable table entry.

Public Attributes

- const Entry * entries

6.92.1 Detailed Description

Definition at line 125 of file offload_table.h.

6.92.2 Member Data Documentation

const Entry* VarTable::entries

Definition at line 151 of file offload_table.h.

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

- offload_table.h

6.93 MicEnvVar::VarValue Struct Reference

`#include <offload_env.h>`

Public Member Functions

- VarValue (char *var, int ln, char *value)
- ~VarValue ()

Public Attributes

- char * env_var
- int length
- char * env_var_value

6.93.1 Detailed Description

Definition at line 51 of file offload_env.h.

6.93.2 Constructor & Destructor Documentation

MicEnvVar::VarValue::VarValue (char * *var,* int *ln,* char * *value*) `[inline]`

Definition at line 57 of file offload_env.h.

MicEnvVar::VarValue::~VarValue ( )

Definition at line 30 of file offload_env.cpp.

6.93.3 Member Data Documentation

char* MicEnvVar::VarValue::env_var

Definition at line 53 of file offload_env.h.

Referenced by MicEnvVar::create_environ_for_card(), MicEnvVar::CardEnvVars::find_var(), and VarValue().

char* MicEnvVar::VarValue::env_var_value

Definition at line 55 of file offload_env.h.

Referenced by MicEnvVar::create_environ_for_card(), VarValue(), and ~VarValue().

int MicEnvVar::VarValue::length

Definition at line 54 of file offload_env.h.

Referenced by MicEnvVar::create_environ_for_card(), MicEnvVar::CardEnvVars::find_var(), and VarValue().

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

- offload_env.h
- offload_env.cpp

Chapter 7

File Documentation

7.1 cean_util.cpp File Reference

```
#include "cean_util.h"
#include "offload_common.h"
```

Typedefs

- typedef void(* fpp)(const char *spaces, uint64_t low, uint64_t high, int esize)

Functions

- CeanReadRanges * init_read_ranges_arr_desc (const arr_desc *ap)
- bool cean_ranges_match (CeanReadRanges *read_rng1, CeanReadRanges *read_rng2)
- bool get_next_range (CeanReadRanges *read_rng, int64_t *offset)
- bool is_arr_desc_contiguous (const arr_desc *ap)
- int64_t cean_get_transf_size (CeanReadRanges *read_rng)
- static void generate_one_range (const char *spaces, uint64_t lrange, uint64_t rrange, fpp fp, int esize)
- static void generate_mem_ranges_one_rank (const char *spaces, uint64_t base, uint64_t rank, const struct dim_desc *ddp, fpp fp, int esize)
- static void generate_mem_ranges (const char *spaces, const arr_desc *adp, bool deref, fpp fp)
- void __arr_data_offset_and_length (const arr_desc *adp, int64_t &offset, int64_t &length)

Variables

- static uint64_t last_left
- static uint64_t last_right

7.1.1 Typedef Documentation

typedef void(* fpp)(const char *spaces, uint64_t low, uint64_t high, int esize)

Definition at line 127 of file cean_util.cpp.

7.1.2 Function Documentation

void __arr_data_offset_and_length (const arr_desc * *adp,* int64_t & *offset,* int64_t & *length*)

Definition at line 227 of file cean_util.cpp.
Referenced by get_arr_desc_numbers(), and OffloadDescriptor::setup_descriptors().

int64_t cean_get_transf_size (CeanReadRanges * *read_rng*)

Definition at line 121 of file cean_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

bool cean_ranges_match (CeanReadRanges * *read_rng1*, CeanReadRanges * *read_rng2*)

Definition at line 65 of file cean_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

static void generate_mem_ranges (const char * *spaces*, const arr_desc * *adp*, bool *deref*, fpp *fp*) [static]

Definition at line 199 of file cean_util.cpp.

static void generate_mem_ranges_one_rank (const char * *spaces*, uint64_t *base*, uint64_t *rank*, const struct dim_desc * *ddp*, fpp *fp*, int *esize*) [static]

Definition at line 156 of file cean_util.cpp.
Referenced by generate_mem_ranges().

static void generate_one_range (const char * *spaces*, uint64_t *lrange*, uint64_t *rrange*, fpp *fp*, int *esize*) [static]

Definition at line 129 of file cean_util.cpp.
Referenced by generate_mem_ranges_one_rank().

bool get_next_range (CeanReadRanges * *read_rng*, int64_t * *offset*)

Definition at line 77 of file cean_util.cpp.
Referenced by OffloadDescriptor::ReadArrElements< T >::read_next(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), and OffloadDescriptor::send_noncontiguous_pointer_data().

CeanReadRanges* init_read_ranges_arr_desc (const arr_desc * *ap*)

Definition at line 17 of file cean_util.cpp.
Referenced by get_arr_desc_numbers(), and OffloadDescriptor::setup_descriptors().

bool is_arr_desc_contiguous (const arr_desc * *ap*)

Definition at line 99 of file cean_util.cpp.
Referenced by get_arr_desc_numbers(), and OffloadDescriptor::setup_descriptors().

7.1.3 Variable Documentation

uint64_t last_left [static]

Definition at line 126 of file cean_util.cpp.
Referenced by generate_mem_ranges(), and generate_one_range().

uint64_t last_right [static]

Definition at line 126 of file cean_util.cpp.
Referenced by generate_mem_ranges(), and generate_one_range().

7.2 cean_util.h File Reference

```
#include <stdint.h>
```

Classes

- struct dim_desc
- struct arr_desc
- struct CeanReadDim
- struct CeanReadRanges

Macros

- #define __arr_desc_length(rank) (sizeof(int64_t) + sizeof(dim_desc) ∗ (rank))
- #define __arr_desc_dump(spaces, name, adp, dereference)

Functions

- void __arr_data_offset_and_length (const arr_desc ∗adp, int64_t &offset, int64_t &length)
- bool is_arr_desc_contiguous (const arr_desc ∗ap)
- CeanReadRanges ∗ init_read_ranges_arr_desc (const arr_desc ∗ap)
- bool cean_ranges_match (CeanReadRanges ∗read_rng1, CeanReadRanges ∗read_rng2)
- bool get_next_range (CeanReadRanges ∗read_rng, int64_t ∗offset)
- int64_t cean_get_transf_size (CeanReadRanges ∗read_rng)

7.2.1 Macro Definition Documentation

#define __arr_desc_dump(*spaces, name, adp, dereference*)

Definition at line 93 of file cean_util.h.
Referenced by OffloadDescriptor::setup_descriptors().

#define __arr_desc_length(*rank*) (sizeof(int64_t) + sizeof(dim_desc) ∗ (rank))

Definition at line 48 of file cean_util.h.

7.2.2 Function Documentation

void __arr_data_offset_and_length (const arr_desc ∗ *adp,* int64_t & *offset,* int64_t & *length*)

Definition at line 227 of file cean_util.cpp.
Referenced by get_arr_desc_numbers(), and OffloadDescriptor::setup_descriptors().

int64_t cean_get_transf_size (CeanReadRanges ∗ *read_rng*)

Definition at line 121 of file cean_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

bool cean_ranges_match (CeanReadRanges ∗ *read_rng1,* CeanReadRanges ∗ *read_rng2*)

Definition at line 65 of file cean_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

bool get_next_range (CeanReadRanges ∗ *read_rng,* int64_t ∗ *offset*)

Definition at line 77 of file cean_util.cpp.
Referenced by OffloadDescriptor::ReadArrElements< T >::read_next(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), and OffloadDescriptor::send_noncontiguous_pointer_data().

CeanReadRanges∗ init_read_ranges_arr_desc (const arr_desc ∗ *ap*)

Definition at line 17 of file cean_util.cpp.
Referenced by get_arr_desc_numbers(), and OffloadDescriptor::setup_descriptors().

bool is_arr_desc_contiguous (const arr_desc * *ap*)

Definition at line 99 of file cean_util.cpp.

Referenced by get_arr_desc_numbers(), and OffloadDescriptor::setup_descriptors().

7.3 coi/coi_client.cpp File Reference

```
#include "coi_client.h"
#include "../offload_common.h"
```

Namespaces

- COI

Macros

- #define COI_VERSION1 "COI_1.0"
- #define COI_VERSION2 "COI_2.0"

Functions

- bool COI::init (void)
- void COI::fini (void)

Variables

- bool COI::is_available
- static void * COI::lib_handle
- COIRESULT(* COI::EngineGetCount)(COI_ISA_TYPE, uint32_t *)
- COIRESULT(* COI::EngineGetHandle)(COI_ISA_TYPE, uint32_t, COIENGINE *)
- COIRESULT(* COI::ProcessCreateFromMemory)(COIENGINE, const char *, const void *, uint64_t, int, const char **, uint8_t, const char **, uint8_t, const char *, uint64_t, const char *, const char *, uint64_t, COIPROCESS *)
- COIRESULT(* COI::ProcessDestroy)(COIPROCESS, int32_t, uint8_t, int8_t *, uint32_t *)
- COIRESULT(* COI::ProcessGetFunctionHandles)(COIPROCESS, uint32_t, const char **, COIFUNCTION *)
- COIRESULT(* COI::ProcessLoadLibraryFromMemory)(COIPROCESS, const void *, uint64_t, const char *, const char *, const char *, uint64_t, uint32_t, COILIBRARY *)
- COIRESULT(* COI::ProcessRegisterLibraries)(uint32_t, const void **, const uint64_t *, const char **, const uint64_t *)
- COIRESULT(* COI::PipelineCreate)(COIPROCESS, COI_CPU_MASK, uint32_t, COIPIPELINE *)
- COIRESULT(* COI::PipelineDestroy)(COIPIPELINE)
- COIRESULT(* COI::PipelineRunFunction)(COIPIPELINE, COIFUNCTION, uint32_t, const COIBUFFER *, const COI_ACCESS_FLAGS *, uint32_t, const COIEVENT *, const void *, uint16_t, void *, uint16_t, COIEVENT *)
- COIRESULT(* COI::BufferCreate)(uint64_t, COI_BUFFER_TYPE, uint32_t, const void *, uint32_t, const COIPROCESS *, COIBUFFER *)
- COIRESULT(* COI::BufferCreateFromMemory)(uint64_t, COI_BUFFER_TYPE, uint32_t, void *, uint32_t, const COIPROCESS *, COIBUFFER *)
- COIRESULT(* COI::BufferDestroy)(COIBUFFER)
- COIRESULT(* COI::BufferMap)(COIBUFFER, uint64_t, uint64_t, COI_MAP_TYPE, uint32_t, const COIEVENT *, COIEVENT *, COIMAPINSTANCE *, void **)
- COIRESULT(* COI::BufferUnmap)(COIMAPINSTANCE, uint32_t, const COIEVENT *, COIEVENT *)
- COIRESULT(* COI::BufferWrite)(COIBUFFER, uint64_t, const void *, uint64_t, COI_COPY_TYPE, uint32_t, const COIEVENT *, COIEVENT *)

- COIRESULT(∗ COI::BufferRead)(COIBUFFER, uint64_t, void ∗, uint64_t, COI_COPY_TYPE, uint32_t, const COIEVENT ∗, COIEVENT ∗)
- COIRESULT(∗ COI::BufferCopy)(COIBUFFER, COIBUFFER, uint64_t, uint64_t, uint64_t, COI_COPY_TYPE, uint32_t, const COIEVENT ∗, COIEVENT ∗)
- COIRESULT(∗ COI::BufferGetSinkAddress)(COIBUFFER, uint64_t ∗)
- COIRESULT(∗ COI::BufferSetState)(COIBUFFER, COIPROCESS, COI_BUFFER_STATE, COI_BUFFER_MOVE_FLAG, uint32_t, const COIEVENT ∗, COIEVENT ∗)
- COIRESULT(∗ COI::EventWait)(uint16_t, const COIEVENT ∗, int32_t, uint8_t, uint32_t ∗, uint32_t ∗)
- uint64_t(∗ COI::PerfGetCycleFrequency)(void)

7.3.1 Macro Definition Documentation

#define COI_VERSION1 "COI_1.0"

Definition at line 18 of file coi_client.cpp.
Referenced by COI::init().

#define COI_VERSION2 "COI_2.0"

Definition at line 19 of file coi_client.cpp.
Referenced by COI::init().

7.4 coi/coi_client.h File Reference

```
#include <common/COIPerf_common.h>
#include <source/COIEngine_source.h>
#include <source/COIProcess_source.h>
#include <source/COIPipeline_source.h>
#include <source/COIBuffer_source.h>
#include <source/COIEvent_source.h>
#include <string.h>
#include "../liboffload_error_codes.h"
#include "../offload_util.h"
```

Namespaces

- COI

Macros

- #define MIC_ENGINES_MAX 128

Functions

- bool COI::init (void)
- void COI::fini (void)

7.4.1 Macro Definition Documentation

#define MIC_ENGINES_MAX 128

Definition at line 28 of file coi_client.h.
Referenced by __offload_init_library_once(), __offload_myoFini(), and __offload_myoInit_once().

7.5 coi/coi_server.cpp File Reference

```
#include "coi_server.h"
#include "../offload_target.h"
#include "../offload_timer.h"
```

Functions

- COINATIVELIBEXPORT void server_compute (uint32_t buffer_count, void **buffers, uint64_t *buffers_len, void *misc_data, uint16_t misc_data_len, void *return_data, uint16_t return_data_len)
- COINATIVELIBEXPORT void server_init (uint32_t buffer_count, void **buffers, uint64_t *buffers_len, void *misc_data, uint16_t misc_data_len, void *return_data, uint16_t return_data_len)
- COINATIVELIBEXPORT void server_var_table_size (uint32_t buffer_count, void **buffers, uint64_t *buffers_len, void *misc_data, uint16_t misc_data_len, void *return_data, uint16_t return_data_len)
- COINATIVELIBEXPORT void server_var_table_copy (uint32_t buffer_count, void **buffers, uint64_t *buffers_len, void *misc_data, uint16_t misc_data_len, void *return_data, uint16_t return_data_len)

7.5.1 Function Documentation

COINATIVELIBEXPORT void server_compute (uint32_t *buffer_count,* void ** *buffers,* uint64_t * *buffers_len,* void * *misc_data,* uint16_t *misc_data_len,* void * *return_data,* uint16_t *return_data_len*)

Definition at line 22 of file coi_server.cpp.

COINATIVELIBEXPORT void server_init (uint32_t *buffer_count,* void ** *buffers,* uint64_t * *buffers_len,* void * *misc_data,* uint16_t *misc_data_len,* void * *return_data,* uint16_t *return_data_len*)

Definition at line 38 of file coi_server.cpp.

COINATIVELIBEXPORT void server_var_table_copy (uint32_t *buffer_count,* void ** *buffers,* uint64_t * *buffers_len,* void * *misc_data,* uint16_t *misc_data_len,* void * *return_data,* uint16_t *return_data_len*)

Definition at line 88 of file coi_server.cpp.

COINATIVELIBEXPORT void server_var_table_size (uint32_t *buffer_count,* void ** *buffers,* uint64_t * *buffers_len,* void * *misc_data,* uint16_t *misc_data_len,* void * *return_data,* uint16_t *return_data_len*)

Definition at line 68 of file coi_server.cpp.

7.6 coi/coi_server.h File Reference

```
#include <common/COIEngine_common.h>
#include <common/COIPerf_common.h>
#include <sink/COIProcess_sink.h>
#include <sink/COIPipeline_sink.h>
#include <sink/COIBuffer_sink.h>
#include <list>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include "../liboffload_error_codes.h"
```

Macros

- #define PipelineStartExecutingRunFunctions()
- #define ProcessWaitForShutdown()
- #define BufferAddRef(buf)
- #define BufferReleaseRef(buf)
- #define EngineGetIndex(index)

7.6.1 Macro Definition Documentation

#define BufferAddRef(*buf*)

Value:

```
{ \
        COIRESULT res = COIBufferAddRef(buf); \
        if (res != COI_SUCCESS) { \
            LIBOFFLOAD_ERROR(c_buf_add_ref,
      mic_index, res); \
            exit(1); \
        } \
    }
```

Definition at line 46 of file coi_server.h.
Referenced by OffloadDescriptor::scatter_copyin_data().

#define BufferReleaseRef(*buf*)

Value:

```
{ \
        COIRESULT res = COIBufferReleaseRef(buf); \
        if (res != COI_SUCCESS) { \
            LIBOFFLOAD_ERROR(c_buf_release_ref,
      mic_index, res); \
            exit(1); \
        } \
    }
```

Definition at line 55 of file coi_server.h.
Referenced by OffloadDescriptor::scatter_copyin_data().

#define EngineGetIndex(*index*)

Value:

```
{ \
        COI_ISA_TYPE isa_type; \
        COIRESULT res = COIEngineGetIndex(&isa_type, index); \
        if (res != COI_SUCCESS) { \
            LIBOFFLOAD_ERROR(c_get_engine_index,
      mic_index, res); \
            exit(1); \
        } \
    }
```

Definition at line 64 of file coi_server.h.
Referenced by _Offload_get_physical_device_number().

#define PipelineStartExecutingRunFunctions( )

Value:

```
{ \
        COIRESULT res = COIPipelineStartExecutingRunFunctions(); \
        if (res != COI_SUCCESS) { \
            LIBOFFLOAD_ERROR(
      c_pipeline_start_run_funcs, mic_index, res); \
            exit(1); \
        } \
    }
```

Definition at line 28 of file coi_server.h.
Referenced by OFFLOAD_TARGET_MAIN().

#define ProcessWaitForShutdown( )

Value:

```
{ \
        COIRESULT res = COIProcessWaitForShutdown(); \
        if (res != COI_SUCCESS) { \
            LIBOFFLOAD_ERROR(c_process_wait_shutdown,
      mic_index, res); \
            exit(1); \
        } \
    }
```

Definition at line 37 of file coi_server.h.
Referenced by OFFLOAD_TARGET_MAIN().

7.7 compiler_if_host.cpp File Reference

```
#include "compiler_if_host.h"
#include <malloc.h>
#include <alloca.h>
```

Functions

- OFFLOAD OFFLOAD_TARGET_ACQUIRE (TARGET_TYPE target_type, int target_number, int is_optional, _Offload_status *status, const char *file, uint64_t line)
- OFFLOAD OFFLOAD_TARGET_ACQUIRE1 (const int *device_num, const char *file, uint64_t line)
- int offload_offload_wrap (OFFLOAD ofld, const char *name, int is_empty, int num_vars, VarDesc *vars, VarDesc2 *vars2, int num_waits, const void **waits, const void **signal, int entry_id, const void *stack_addr)
- int OFFLOAD_OFFLOAD1 (OFFLOAD ofld, const char *name, int is_empty, int num_vars, VarDesc *vars, VarDesc2 *vars2, int num_waits, const void **waits, const void **signal)
- int OFFLOAD_OFFLOAD2 (OFFLOAD ofld, const char *name, int is_empty, int num_vars, VarDesc *vars, VarDesc2 *vars2, int num_waits, const void **waits, const void **signal, int entry_id, const void *stack_addr)
- int OFFLOAD_OFFLOAD (OFFLOAD ofld, const char *name, int is_empty, int num_vars, VarDesc *vars, VarDesc2 *vars2, int num_waits, const void **waits, const void *signal, int entry_id, const void *stack_addr)
- int OFFLOAD_CALL_COUNT ()

Variables

- static int offload_call_count = 0

7.7.1 Function Documentation

int OFFLOAD_CALL_COUNT ( )

Definition at line 319 of file compiler_if_host.cpp.

int OFFLOAD_OFFLOAD (OFFLOAD *ofld,* const char * *name,* int *is_empty,* int *num_vars,* VarDesc * *vars,* VarDesc2 * *vars2,* int *num_waits,* const void ** *waits,* const void * *signal,* int *entry_id,* const void * *stack_addr*)

Definition at line 284 of file compiler_if_host.cpp.

int OFFLOAD_OFFLOAD1 (OFFLOAD *ofld,* const char * *name,* int *is_empty,* int *num_vars,* VarDesc * *vars,* VarDesc2 * *vars2,* int *num_waits,* const void ** *waits,* const void ** *signal*)

Definition at line 246 of file compiler_if_host.cpp.
Referenced by OFFLOAD_OFFLOAD().

int OFFLOAD_OFFLOAD2 (OFFLOAD *ofld,* const char * *name,* int *is_empty,* int *num_vars,* VarDesc * *vars,* VarDesc2 * *vars2,* int *num_waits,* const void ** *waits,* const void ** *signal,* int *entry_id,* const void * *stack_addr*)

Definition at line 264 of file compiler_if_host.cpp.

int offload_offload_wrap (OFFLOAD *ofld,* const char * *name,* int *is_empty,* int *num_vars,* VarDesc * *vars,* VarDesc2 * *vars2,* int *num_waits,* const void ** *waits,* const void ** *signal,* int *entry_id,* const void * *stack_addr*)

Definition at line 224 of file compiler_if_host.cpp.
Referenced by OFFLOAD_OFFLOAD1(), and OFFLOAD_OFFLOAD2().

OFFLOAD OFFLOAD_TARGET_ACQUIRE (TARGET_TYPE *target_type,* int *target_number,* int *is_optional,* _Offload_status * *status,* const char * *file,* uint64_t *line*)

Definition at line 25 of file compiler_if_host.cpp.

OFFLOAD OFFLOAD_TARGET_ACQUIRE1 (const int * *device_num,* const char * *file,* uint64_t *line*)

Definition at line 144 of file compiler_if_host.cpp.

7.7.2 Variable Documentation

int offload_call_count = 0 [static]

Definition at line 23 of file compiler_if_host.cpp.
Referenced by OFFLOAD_CALL_COUNT().

7.8 compiler_if_host.h File Reference

The interface between compiler-generated host code and runtime library.

```
#include "offload_host.h"
```

Macros

- #define OFFLOAD_TARGET_ACQUIRE OFFLOAD_PREFIX(target_acquire)

  Attempt to acquire the target.
- #define OFFLOAD_TARGET_ACQUIRE1 OFFLOAD_PREFIX(target_acquire1)

  Acquire the target for offload (OpenMP).
- #define OFFLOAD_OFFLOAD OFFLOAD_PREFIX(offload)
- #define OFFLOAD_OFFLOAD1 OFFLOAD_PREFIX(offload1)

  Run function on target using interface for old data persistence.
- #define OFFLOAD_OFFLOAD2 OFFLOAD_PREFIX(offload2)

  Run function on target using interface for new data persistence.
- #define OFFLOAD_CALL_COUNT OFFLOAD_PREFIX(offload_call_count)

Functions

- OFFLOAD OFFLOAD_TARGET_ACQUIRE (TARGET_TYPE target_type, int target_number, int is_optional, _Offload_status *status, const char *file, uint64_t line)
- OFFLOAD OFFLOAD_TARGET_ACQUIRE1 (const int *device_number, const char *file, uint64_t line)
- int OFFLOAD_OFFLOAD1 (OFFLOAD o, const char *name, int is_empty, int num_vars, VarDesc *vars, VarDesc2 *vars2, int num_waits, const void **waits, const void **signal)
- int OFFLOAD_OFFLOAD2 (OFFLOAD o, const char *name, int is_empty, int num_vars, VarDesc *vars, VarDesc2 *vars2, int num_waits, const void **waits, const void **signal, int entry_id, const void *stack_addr)

- int OFFLOAD_OFFLOAD (OFFLOAD o, const char *name, int is_empty, int num_vars, VarDesc *vars, VarDesc2 *vars2, int num_waits, const void **waits, const void *signal, int entry_id=0, const void *stack_addr=NULL)
- int OFFLOAD_CALL_COUNT ()

7.8.1 Detailed Description

The interface between compiler-generated host code and runtime library.

Definition in file compiler_if_host.h.

7.8.2 Macro Definition Documentation

#define OFFLOAD_CALL_COUNT OFFLOAD_PREFIX(offload_call_count)

Definition at line 25 of file compiler_if_host.h.

#define OFFLOAD_OFFLOAD OFFLOAD_PREFIX(offload)

Definition at line 22 of file compiler_if_host.h.

Referenced by kmp_create_affinity_mask_target(), kmp_destroy_affinity_mask_target(), kmp_get_affinity_mask_proc_target(), kmp_get_affinity_target(), kmp_set_affinity_mask_proc_target(), kmp_set_affinity_target(), kmp_set_defaults_target(), kmp_set_library_serial_target(), kmp_set_library_throughput_target(), kmp_set_library_turnaround_target(), kmp_unset_affinity_mask_proc_target(), omp_destroy_lock_target(), omp_destroy_nest_lock_target(), omp_get_int_target(), omp_get_schedule_target(), omp_init_lock_target(), omp_init_nest_lock_target(), omp_set_int_target(), omp_set_lock_target(), omp_set_nest_lock_target(), omp_set_schedule_target(), omp_test_lock_target(), omp_test_nest_lock_target(), omp_unset_lock_target(), and omp_unset_nest_lock_target().

OFFLOAD_OFFLOAD1 OFFLOAD_PREFIX(offload1)

Run function on target using interface for old data persistence.

Parameters

o	Offload descriptor created by OFFLOAD_TARGET_ACQUIRE.
name	Name of offload entry point.
is_empty	If no code to execute (e.g. offload_transfer)
num_vars	Number of variable descriptors.
vars	Pointer to VarDesc array.
vars2	Pointer to VarDesc2 array.
num_waits	Number of "wait" values.
waits	Pointer to array of wait values.
signal	Pointer to signal value or NULL.

Definition at line 23 of file compiler_if_host.h.

OFFLOAD_OFFLOAD2 OFFLOAD_PREFIX(offload2)

Run function on target using interface for new data persistence.

Parameters

o	Offload descriptor created by OFFLOAD_TARGET_ACQUIRE.
name	Name of offload entry point.
is_empty	If no code to execute (e.g. offload_transfer)
num_vars	Number of variable descriptors.
vars	Pointer to VarDesc array.

vars2	Pointer to VarDesc2 array.
num_waits	Number of ”wait” values.
waits	Pointer to array of wait values.
signal	Pointer to signal value or NULL.
entry_id	A signature for the function doing the offload.
stack_addr	The stack frame address of the function doing offload.

Definition at line 24 of file compiler_if_host.h.

OFFLOAD_TARGET_ACQUIRE OFFLOAD_PREFIX(target_acquire)

Attempt to acquire the target.

Parameters

target_type	The type of target.
target_number	The device number.
is_optional	Whether CPU fall-back is allowed.
status	Address of variable to hold offload status.
file	Filename in which this offload occurred.
line	Line number in the file where this offload occurred.

Definition at line 20 of file compiler_if_host.h.

Referenced by kmp_create_affinity_mask_target(), kmp_destroy_affinity_mask_target(), kmp_get_affinity_mask_proc_target(), kmp_get_affinity_target(), kmp_set_affinity_mask_proc_target(), kmp_set_affinity_target(), kmp_set_defaults_target(), kmp_set_library_serial_target(), kmp_set_library_throughput_target(), kmp_set_library_turnaround_target(), kmp_unset_affinity_mask_proc_target(), omp_destroy_lock_target(), omp_destroy_nest_lock_target(), omp_get_int_target(), omp_get_schedule_target(), omp_init_lock_target(), omp_init_nest_lock_target(), omp_set_int_target(), omp_set_lock_target(), omp_set_nest_lock_target(), omp_set_schedule_target(), omp_test_lock_target(), omp_test_nest_lock_target(), omp_unset_lock_target(), and omp_unset_nest_lock_target().

OFFLOAD_TARGET_ACQUIRE1 OFFLOAD_PREFIX(target_acquire1)

Acquire the target for offload (OpenMP).

Parameters

device_number	Device number or null if not specified.
file	Filename in which this offload occurred
line	Line number in the file where this offload occurred.

Definition at line 21 of file compiler_if_host.h.

7.8.3 Function Documentation

int OFFLOAD_CALL_COUNT ( )

Definition at line 319 of file compiler_if_host.cpp.

int OFFLOAD_OFFLOAD (OFFLOAD *o*, const char * *name*, int *is_empty*, int *num_vars*, VarDesc * *vars*, VarDesc2 * *vars2*, int *num_waits*, const void ** *waits*, const void * *signal*, int *entry_id = 0*, const void * *stack_addr = NULL*)

Definition at line 284 of file compiler_if_host.cpp.

int OFFLOAD_OFFLOAD1 (OFFLOAD *o*, const char * *name*, int *is_empty*, int *num_vars*, VarDesc * *vars*, VarDesc2 * *vars2*, int *num_waits*, const void ** *waits*, const void ** *signal*)

Definition at line 246 of file compiler_if_host.cpp.

Referenced by OFFLOAD_OFFLOAD().

int OFFLOAD_OFFLOAD2 (OFFLOAD *o,* const char * *name,* int *is_empty,* int *num_vars,* VarDesc * *vars,* VarDesc2 * *vars2,* int *num_waits,* const void ** *waits,* const void ** *signal,* int *entry_id,* const void * *stack_addr*)

Definition at line 264 of file compiler_if_host.cpp.

OFFLOAD OFFLOAD_TARGET_ACQUIRE (TARGET_TYPE *target_type,* int *target_number,* int *is_optional,* _Offload_status * *status,* const char * *file,* uint64_t *line*)

Definition at line 25 of file compiler_if_host.cpp.

OFFLOAD OFFLOAD_TARGET_ACQUIRE1 (const int * *device_number,* const char * *file,* uint64_t *line*)

Definition at line 144 of file compiler_if_host.cpp.

7.9 compiler_if_target.cpp File Reference

```
#include "compiler_if_target.h"
```

Functions

- void OFFLOAD_TARGET_ENTER (OFFLOAD ofld, int vars_total, VarDesc *vars, VarDesc2 *vars2)
- void OFFLOAD_TARGET_LEAVE (OFFLOAD ofld)
- void OFFLOAD_TARGET_MAIN (void)

7.9.1 Function Documentation

void OFFLOAD_TARGET_ENTER (OFFLOAD *ofld,* int *vars_total,* VarDesc * *vars,* VarDesc2 * *vars2*)

Definition at line 13 of file compiler_if_target.cpp.

void OFFLOAD_TARGET_LEAVE (OFFLOAD *ofld*)

Definition at line 26 of file compiler_if_target.cpp.

void OFFLOAD_TARGET_MAIN (void)

Definition at line 34 of file compiler_if_target.cpp.

7.10 compiler_if_target.h File Reference

The interface between compiler-generated target code and runtime library.

```
#include "offload_target.h"
```

Macros

- #define OFFLOAD_TARGET_ENTER OFFLOAD_PREFIX(target_enter)

  Fill in variable addresses using VarDesc array.
- #define OFFLOAD_TARGET_LEAVE OFFLOAD_PREFIX(target_leave)

  Call back the runtime library to gather outputs using VarDesc array.
- #define OFFLOAD_TARGET_MAIN OFFLOAD_PREFIX(target_main)

Functions

- void OFFLOAD_TARGET_ENTER (OFFLOAD ofld, int var_desc_num, VarDesc *var_desc, VarDesc2 *var_desc2)
- void OFFLOAD_TARGET_LEAVE (OFFLOAD ofld)
- void OFFLOAD_TARGET_MAIN (void)

7.10.1 Detailed Description

The interface between compiler-generated target code and runtime library.

Definition in file compiler_if_target.h.

7.10.2 Macro Definition Documentation

OFFLOAD_TARGET_ENTER OFFLOAD_PREFIX(target_enter)

Fill in variable addresses using VarDesc array.

Then call back the runtime library to fetch data.

Parameters

ofld	Offload descriptor created by runtime.
var_desc_num	Number of variable descriptors.
var_desc	Pointer to VarDesc array.
var_desc2	Pointer to VarDesc2 array.

Definition at line 20 of file compiler_if_target.h.

Referenced by kmp_create_affinity_mask_lrb(), kmp_destroy_affinity_mask_lrb(), kmp_get_affinity_lrb(), kmp_get_affinity_mask_proc_lrb(), kmp_set_affinity_lrb(), kmp_set_affinity_mask_proc_lrb(), kmp_set_defaults_lrb(), kmp_set_library_serial_lrb(), kmp_set_library_throughput_lrb(), kmp_set_library_turnaround_lrb(), kmp_unset_affinity_mask_proc_lrb(), omp_destroy_lock_lrb(), omp_destroy_nest_lock_lrb(), omp_get_int_from_host(), omp_get_schedule_lrb(), omp_init_lock_lrb(), omp_init_nest_lock_lrb(), omp_send_int_to_host(), omp_set_lock_lrb(), omp_set_nest_lock_lrb(), omp_set_schedule_lrb(), omp_test_lock_lrb(), omp_test_nest_lock_lrb(), omp_unset_lock_lrb(), and omp_unset_nest_lock_lrb().

OFFLOAD_TARGET_LEAVE OFFLOAD_PREFIX(target_leave)

Call back the runtime library to gather outputs using VarDesc array.

Parameters

ofld	Offload descriptor created by OFFLOAD_TARGET_ACQUIRE.

Definition at line 21 of file compiler_if_target.h.

Referenced by kmp_create_affinity_mask_lrb(), kmp_destroy_affinity_mask_lrb(), kmp_get_affinity_lrb(), kmp_get_affinity_mask_proc_lrb(), kmp_set_affinity_lrb(), kmp_set_affinity_mask_proc_lrb(), kmp_set_defaults_lrb(), kmp_set_library_serial_lrb(), kmp_set_library_throughput_lrb(), kmp_set_library_turnaround_lrb(), kmp_unset_affinity_mask_proc_lrb(), omp_destroy_lock_lrb(), omp_destroy_nest_lock_lrb(), omp_get_int_from_host(), omp_get_schedule_lrb(), omp_init_lock_lrb(), omp_init_nest_lock_lrb(), omp_send_int_to_host(), omp_set_lock_lrb(), omp_set_nest_lock_lrb(), omp_set_schedule_lrb(), omp_test_lock_lrb(), omp_test_nest_lock_lrb(), omp_unset_lock_lrb(), and omp_unset_nest_lock_lrb().

#define OFFLOAD_TARGET_MAIN OFFLOAD_PREFIX(target_main)

Definition at line 22 of file compiler_if_target.h.

Referenced by main(), and MAIN__().

7.10.3 Function Documentation

void OFFLOAD_TARGET_ENTER (OFFLOAD *ofld*, int *var_desc_num*, VarDesc * *var_desc*, VarDesc2 * *var_desc2*)

Definition at line 13 of file compiler_if_target.cpp.

void OFFLOAD_TARGET_LEAVE (OFFLOAD *ofld*)

Definition at line 26 of file compiler_if_target.cpp.

void OFFLOAD_TARGET_MAIN (void)

Definition at line 34 of file compiler_if_target.cpp.

7.11 dv_util.cpp File Reference

```
#include "offload_common.h"
```

Functions

- bool __dv_is_contiguous (const ArrDesc *dvp)
- bool __dv_is_allocated (const ArrDesc *dvp)
- uint64_t __dv_data_length (const ArrDesc *dvp)
- uint64_t __dv_data_length (const ArrDesc *dvp, int64_t count)
- CeanReadRanges * init_read_ranges_dv (const ArrDesc *dvp)

7.11.1 Function Documentation

uint64_t __dv_data_length (const ArrDesc * *dvp*)

Definition at line 38 of file dv_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

uint64_t __dv_data_length (const ArrDesc * *dvp*, int64_t *count*)

Definition at line 54 of file dv_util.cpp.

bool __dv_is_allocated (const ArrDesc * *dvp*)

Definition at line 33 of file dv_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

bool __dv_is_contiguous (const ArrDesc * *dvp*)

Definition at line 13 of file dv_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

CeanReadRanges* init_read_ranges_dv (const ArrDesc * *dvp*)

Definition at line 65 of file dv_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

7.12 dv_util.h File Reference

```
#include <stdint.h>
```

Classes

- struct DimDesc
- struct ArrDesc

Macros

- #define ArrDescMaxArrayRank 31
- #define ArrDescFlagsDefined 1
- #define ArrDescFlagsNodealloc 2
- #define ArrDescFlagsContiguous 4
- #define __dv_desc_dump(name, dvp)

Typedefs

- typedef int64_t dv_size
- typedef struct DimDesc DimDesc
- typedef struct ArrDesc ArrDesc
- typedef ArrDesc * pArrDesc

Functions

- bool __dv_is_contiguous (const ArrDesc *dvp)
- bool __dv_is_allocated (const ArrDesc *dvp)
- uint64_t __dv_data_length (const ArrDesc *dvp)
- uint64_t __dv_data_length (const ArrDesc *dvp, int64_t nelems)
- CeanReadRanges * init_read_ranges_dv (const ArrDesc *dvp)

7.12.1 Macro Definition Documentation

#define __dv_desc_dump(*name, dvp*)

Definition at line 60 of file dv_util.h.
Referenced by OffloadDescriptor::setup_descriptors().

#define ArrDescFlagsContiguous 4

Definition at line 22 of file dv_util.h.
Referenced by __dv_is_contiguous().

#define ArrDescFlagsDefined 1

Definition at line 20 of file dv_util.h.
Referenced by __dv_is_allocated().

#define ArrDescFlagsNodealloc 2

Definition at line 21 of file dv_util.h.

#define ArrDescMaxArrayRank 31

Definition at line 17 of file dv_util.h.

7.12.2 Typedef Documentation

typedef struct ArrDesc ArrDesc

typedef struct DimDesc DimDesc

typedef int64_t dv_size

Definition at line 24 of file dv_util.h.

typedef ArrDesc* pArrDesc

Definition at line 45 of file dv_util.h.

7.12.3 Function Documentation

uint64_t __dv_data_length (const ArrDesc * *dvp*)

Definition at line 38 of file dv_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

uint64_t __dv_data_length (const ArrDesc * *dvp,* int64_t *nelems*)

Definition at line 54 of file dv_util.cpp.

bool __dv_is_allocated (const ArrDesc * *dvp*)

Definition at line 33 of file dv_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

bool __dv_is_contiguous (const ArrDesc * *dvp*)

Definition at line 13 of file dv_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

CeanReadRanges* init_read_ranges_dv (const ArrDesc * *dvp*)

Definition at line 65 of file dv_util.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

7.13 liboffload_error.c File Reference

```
#include <stdio.h>
#include <stdarg.h>
#include "liboffload_msg.h"
#include "liboffload_error_codes.h"
```

Macros

- #define va_copy(dst, src) ((dst) = (src))

Functions

- void __liboffload_error_support (error_types input_tag,...)
- char const * report_get_message_str (error_types input_tag)
- char const * report_get_host_stage_str (int i)
- char const * report_get_target_stage_str (int i)

7.13.1 Macro Definition Documentation

#define va_copy(*dst, src*) ((dst) = (src))

Definition at line 14 of file liboffload_error.c.

7.13.2 Function Documentation

void __liboffload_error_support (error_types *input_tag, ...*)

Definition at line 25 of file liboffload_error.c.

char const* report_get_host_stage_str (int *i*)

Definition at line 361 of file liboffload_error.c.

char const* report_get_message_str (error_types *input_tag*)

Definition at line 242 of file liboffload_error.c.

Referenced by __offload_init_library_once(), __offload_target_init(), offload_signal(), offload_stage(), and offload_stage_print().

char const* report_get_target_stage_str (int *i*)

Definition at line 422 of file liboffload_error.c.

7.14 liboffload_error_codes.h File Reference

```
#include <stdarg.h>
#include <stdlib.h>
#include <stdio.h>
```

Macros

- #define test_msg_cat(nm, msg)
- #define test_msg_cat1(nm, msg,...)
- #define LIBOFFLOAD_ERROR __liboffload_error_support
- #define LIBOFFLOAD_ABORT abort()

Enumerations

- enum error_types {
  c_device_is_not_available = 0, c_invalid_device_number, c_offload1, c_unknown_var_type,
  c_send_func_ptr, c_receive_func_ptr, c_offload_malloc, c_invalid_env_var_value,
  c_invalid_env_var_int_value, c_invalid_env_report_value, c_offload_signaled1, c_offload_signaled2,
  c_myotarget_checkresult, c_myowrapper_checkresult, c_offload_descriptor_offload, c_merge_var_descs1,
  c_merge_var_descs2, c_mic_parse_env_var_list1, c_mic_parse_env_var_list2, c_mic_process_exit_ret,
  c_mic_process_exit_sig, c_mic_process_exit, c_mic_init3, c_mic_init4,
  c_mic_init5, c_mic_init6, c_no_static_var_data, c_no_ptr_data,
  c_get_engine_handle, c_get_engine_index, c_process_create, c_process_get_func_handles,
  c_process_wait_shutdown, c_process_proxy_flush, c_load_library, c_pipeline_create,
  c_pipeline_run_func, c_pipeline_start_run_funcs, c_buf_create, c_buf_create_out_of_mem,
  c_buf_create_from_mem, c_buf_destroy, c_buf_map, c_buf_unmap,
  c_buf_read, c_buf_write, c_buf_copy, c_buf_get_address,
  c_buf_add_ref, c_buf_release_ref, c_buf_set_state, c_event_wait,
  c_zero_or_neg_ptr_len, c_zero_or_neg_transfer_size, c_bad_ptr_mem_range, c_different_src_and_dstn_sizes,
  c_ranges_dont_match, c_destination_is_over, c_slice_of_noncont_array, c_non_contiguous_dope_vector,
  c_pointer_array_mismatch, c_omp_invalid_device_num_env, c_omp_invalid_device_num, c_unknown_binary_type,
  c_multiple_target_exes, c_no_target_exe, c_report_host, c_report_target,
  c_report_title, c_report_from_file, c_report_file, c_report_line,
  c_report_tag, c_report_seconds, c_report_bytes, c_report_mic,
  c_report_cpu_time, c_report_cpu_to_mic_data, c_report_mic_time, c_report_mic_to_cpu_data,
  c_report_unknown_timer_node, c_report_unknown_trace_node, c_report_offload, c_report_w_tag,
  c_report_state, c_report_start, c_report_init, c_report_logical_card,
  c_report_physical_card, c_report_register, c_report_init_func, c_report_create_buf_host,
  c_report_create_buf_mic, c_report_send_pointer_data, c_report_sent_pointer_data, c_report_gather_copyin_data,
  c_report_copyin_data, c_report_state_signal, c_report_signal, c_report_wait,
  c_report_compute, c_report_receive_pointer_data, c_report_received_pointer_data, c_report_start_target_func,
  c_report_var, c_report_scatter_copyin_data, c_report_gather_copyout_data, c_report_scatter_copyout_data,
  c_report_copyout_data, c_report_unregister, c_report_destroy, c_report_myoinit,
  c_report_myoregister, c_report_myofini, c_report_mic_myo_shared, c_report_mic_myo_fptr,
  c_report_myosharedmalloc, c_report_myosharedfree, c_report_myosharedalignedmalloc, c_report_myosharedalignedfree,

c_report_myoacquire, c_report_myorelease, c_coipipe_max_number }
- enum OffloadHostPhase {
c_offload_host_total_offload = 0, c_offload_host_initialize, c_offload_host_target_acquire, c_offload_host_wait_deps,
c_offload_host_setup_buffers, c_offload_host_alloc_buffers, c_offload_host_setup_misc_data, c_offload_host_alloc_data_buffer,
c_offload_host_send_pointers, c_offload_host_gather_inputs, c_offload_host_map_in_data_buffer, c_offload_host_unmap_in_data_buffer,
c_offload_host_start_compute, c_offload_host_wait_compute, c_offload_host_start_buffers_reads, c_offload_host_scatter_outputs,
c_offload_host_map_out_data_buffer, c_offload_host_unmap_out_data_buffer, c_offload_host_wait_buffers_reads,
c_offload_host_destroy_buffers,
c_offload_host_max_phase }
- enum OffloadTargetPhase {
c_offload_target_total_time = 0, c_offload_target_descriptor_setup, c_offload_target_func_lookup, c_offload_target_func_time,
c_offload_target_scatter_inputs, c_offload_target_add_buffer_refs, c_offload_target_compute, c_offload_target_gather_outputs,
c_offload_target_release_buffer_refs, c_offload_target_max_phase }

Functions

- void __liboffload_error_support (error_types input_tag,...)
- void __liboffload_report_support (error_types input_tag,...)
- char const * offload_get_message_str (int msgCode)
- char const * report_get_message_str (error_types input_tag)
- char const * report_get_host_stage_str (int i)
- char const * report_get_target_stage_str (int i)
- void write_message (FILE *file, int msgCode, va_list args_p)

7.14.1 Macro Definition Documentation

#define LIBOFFLOAD_ABORT abort()

Definition at line 272 of file liboffload_error_codes.h.

Referenced by _Offload_signaled(), OffloadDescriptor::gen_var_descs_for_pointer_array(), Engine::get_pipeline(), OffloadDescriptor::setup_descriptors(), and OffloadDescriptor::wait_dependencies().

#define LIBOFFLOAD_ERROR __liboffload_error_support

Definition at line 265 of file liboffload_error_codes.h.

Referenced by __offload_init_library_once(), __offload_myoFini(), __offload_myoInit_once(), __offload_myoIsAvailable(), __offload_register_image(), _Offload_signaled(), OffloadDescriptor::alloc_ptr_data(), CheckResult(), MyoWrapper::CheckResult(), OffloadDescriptor::find_ptr_data(), Engine::fini_process(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::gen_var_descs_for_pointer_array(), Engine::get_pipeline(), ORSL::init(), Engine::init_process(), OffloadDescriptor::merge_var_descs(), MicEnvVar::mic_parse_env_var_list(), OffloadDescriptor::offload(), OFFLOAD_MALLOC(), offload_stage_print(), OFFLOAD_TARGET_ACQUIRE(), OFFLOAD_TARGET_ACQUIRE1(), Marshaller::receive_func_ptr(), OffloadDescriptor::recieve_noncontiguous_pointer_data(), OffloadDescriptor::report_coi_error(), report_get_host_stage_str(), report_get_message_str(), report_get_target_stage_str(), OffloadDescriptor::scatter_copyin_data(), Marshaller::send_func_ptr(), OffloadDescriptor::send_noncontiguous_pointer_data(), OffloadDescriptor::setup_descriptors(), and OffloadDescriptor::wait_dependencies().

#define test_msg_cat(*nm, msg*)

Value:

```
fprintf(stderr, "\t TEST for %s \n \t", nm); \
    __liboffload_error_support(msg);
```

Definition at line 255 of file liboffload_error_codes.h.

#define test_msg_cat1(*nm, msg, ...*)

Value:

```
fprintf(stderr, "\t TEST for %s \n \t", nm); \
    __liboffload_error_support(msg, __VA_ARGS__);
```

Definition at line 259 of file liboffload_error_codes.h.

7.14.2 Enumeration Type Documentation

enum error_types

Enumerator

c_device_is_not_available

c_invalid_device_number

c_offload1

c_unknown_var_type

c_send_func_ptr

c_receive_func_ptr

c_offload_malloc

c_invalid_env_var_value

c_invalid_env_var_int_value

c_invalid_env_report_value

c_offload_signaled1

c_offload_signaled2

c_myotarget_checkresult

c_myowrapper_checkresult

c_offload_descriptor_offload

c_merge_var_descs1

c_merge_var_descs2

c_mic_parse_env_var_list1

c_mic_parse_env_var_list2

c_mic_process_exit_ret

c_mic_process_exit_sig

c_mic_process_exit

c_mic_init3

c_mic_init4

c_mic_init5

c_mic_init6

c_no_static_var_data

c_no_ptr_data

c_get_engine_handle

c_get_engine_index

c_process_create

c_process_get_func_handles

c_process_wait_shutdown

c_process_proxy_flush

c_load_library

c_pipeline_create
c_pipeline_run_func
c_pipeline_start_run_funcs
c_buf_create
c_buf_create_out_of_mem
c_buf_create_from_mem
c_buf_destroy
c_buf_map
c_buf_unmap
c_buf_read
c_buf_write
c_buf_copy
c_buf_get_address
c_buf_add_ref
c_buf_release_ref
c_buf_set_state
c_event_wait
c_zero_or_neg_ptr_len
c_zero_or_neg_transfer_size
c_bad_ptr_mem_range
c_different_src_and_dstn_sizes
c_ranges_dont_match
c_destination_is_over
c_slice_of_noncont_array
c_non_contiguous_dope_vector
c_pointer_array_mismatch
c_omp_invalid_device_num_env
c_omp_invalid_device_num
c_unknown_binary_type
c_multiple_target_exes
c_no_target_exe
c_report_host
c_report_target
c_report_title
c_report_from_file
c_report_file
c_report_line
c_report_tag
c_report_seconds
c_report_bytes
c_report_mic
c_report_cpu_time
c_report_cpu_to_mic_data
c_report_mic_time
c_report_mic_to_cpu_data

c_report_unknown_timer_node

c_report_unknown_trace_node

c_report_offload

c_report_w_tag

c_report_state

c_report_start

c_report_init

c_report_logical_card

c_report_physical_card

c_report_register

c_report_init_func

c_report_create_buf_host

c_report_create_buf_mic

c_report_send_pointer_data

c_report_sent_pointer_data

c_report_gather_copyin_data

c_report_copyin_data

c_report_state_signal

c_report_signal

c_report_wait

c_report_compute

c_report_receive_pointer_data

c_report_received_pointer_data

c_report_start_target_func

c_report_var

c_report_scatter_copyin_data

c_report_gather_copyout_data

c_report_scatter_copyout_data

c_report_copyout_data

c_report_unregister

c_report_destroy

c_report_myoinit

c_report_myoregister

c_report_myofini

c_report_mic_myo_shared

c_report_mic_myo_fptr

c_report_myosharedmalloc

c_report_myosharedfree

c_report_myosharedalignedmalloc

c_report_myosharedalignedfree

c_report_myoacquire

c_report_myorelease

c_coipipe_max_number

Definition at line 17 of file liboffload_error_codes.h.

enum OffloadHostPhase

Enumerator

c_offload_host_total_offload

c_offload_host_initialize

c_offload_host_target_acquire

c_offload_host_wait_deps

c_offload_host_setup_buffers

c_offload_host_alloc_buffers

c_offload_host_setup_misc_data

c_offload_host_alloc_data_buffer

c_offload_host_send_pointers

c_offload_host_gather_inputs

c_offload_host_map_in_data_buffer

c_offload_host_unmap_in_data_buffer

c_offload_host_start_compute

c_offload_host_wait_compute

c_offload_host_start_buffers_reads

c_offload_host_scatter_outputs

c_offload_host_map_out_data_buffer

c_offload_host_unmap_out_data_buffer

c_offload_host_wait_buffers_reads

c_offload_host_destroy_buffers

c_offload_host_max_phase

Definition at line 144 of file liboffload_error_codes.h.

enum OffloadTargetPhase

Enumerator

c_offload_target_total_time

c_offload_target_descriptor_setup

c_offload_target_func_lookup

c_offload_target_func_time

c_offload_target_scatter_inputs

c_offload_target_add_buffer_refs

c_offload_target_compute

c_offload_target_gather_outputs

c_offload_target_release_buffer_refs

c_offload_target_max_phase

Definition at line 210 of file liboffload_error_codes.h.

7.14.3 Function Documentation

void __liboffload_error_support (error_types *input_tag,* ...)

Definition at line 25 of file liboffload_error.c.

void __liboffload_report_support (error_types *input_tag,* ...)

char const* offload_get_message_str (int *msgCode*)

Definition at line 33 of file liboffload_msg.c.
Referenced by report_get_host_stage_str(), report_get_message_str(), and report_get_target_stage_str().

char const* report_get_host_stage_str (int *i*)

Definition at line 361 of file liboffload_error.c.

char const* report_get_message_str (error_types *input_tag*)

Definition at line 242 of file liboffload_error.c.
Referenced by __offload_init_library_once(), __offload_target_init(), offload_signal(), offload_stage(), and offload_stage_print().

char const* report_get_target_stage_str (int *i*)

Definition at line 422 of file liboffload_error.c.

void write_message (FILE * *file,* int *msgCode,* va_list *args_p*)

Referenced by __liboffload_error_support().

7.15 liboffload_msg.c File Reference

```
#include <stdio.h>
#include "liboffload_msg.h"
```

Macros

- #define DYNART_STDERR_PUTS(__message_text__) fputs((__message_text__),stderr)

Functions

- void write_message (FILE *file, int msgCode)
- char const * offload_get_message_str (int msgCode)

7.15.1 Macro Definition Documentation

#define DYNART_STDERR_PUTS(*__message_text__*) fputs((__message_text__),stderr)

Definition at line 21 of file liboffload_msg.c.

7.15.2 Function Documentation

char const* offload_get_message_str (int *msgCode*)

Definition at line 33 of file liboffload_msg.c.
Referenced by report_get_host_stage_str(), report_get_message_str(), and report_get_target_stage_str().

void write_message (FILE * *file,* int *msgCode*)

Definition at line 28 of file liboffload_msg.c.

7.16 liboffload_msg.h File Reference

Macros

- #define MESSAGE_TABLE_NAME __liboffload_message_table

Enumerations

- enum {
__dummy__ = 0, msg_c_device_is_not_available, msg_c_invalid_device_number, msg_c_send_func_ptr, msg_c_receive_func_ptr, msg_c_offload_malloc, msg_c_offload1, msg_c_unknown_var_type, msg_c_invalid_env_var_value, msg_c_invalid_env_var_int_value, msg_c_invalid_env_report_value, msg_c_offload_signaled1, msg_c_offload_signaled2, msg_c_myowrapper_checkresult, msg_c_myotarget_checkresult, msg_c_offload_descriptor_offload, msg_c_merge_var_descs1, msg_c_merge_var_descs2, msg_c_mic_parse_env_var_list1, msg_c_mic_parse_env_var_list2, msg_c_mic_process_exit_ret, msg_c_mic_process_exit_sig, msg_c_mic_process_exit, msg_c_mic_init3, msg_c_mic_init4, msg_c_mic_init5, msg_c_mic_init6, msg_c_no_static_var_data, msg_c_no_ptr_data, msg_c_get_engine_handle, msg_c_get_engine_index, msg_c_process_create, msg_c_process_get_func_handles, msg_c_process_wait_shutdown, msg_c_process_proxy_flush, msg_c_load_library, msg_c_pipeline_create, msg_c_pipeline_run_func, msg_c_pipeline_start_run_funcs, msg_c_buf_create, msg_c_buf_create_out_of_mem, msg_c_buf_create_from_mem, msg_c_buf_destroy, msg_c_buf_map, msg_c_buf_unmap, msg_c_buf_read, msg_c_buf_write, msg_c_buf_copy, msg_c_buf_get_address, msg_c_buf_add_ref, msg_c_buf_release_ref, msg_c_buf_set_state, msg_c_event_wait, msg_c_zero_or_neg_ptr_len, msg_c_zero_or_neg_transfer_size, msg_c_bad_ptr_mem_range, msg_c_different_src_and_dstn_sizes, msg_c_non_contiguous_dope_vector, msg_c_omp_invalid_device_num_env, msg_c_omp_invalid_device_num, msg_c_unknown_binary_type, msg_c_multiple_target_exes, msg_c_no_target_exe, msg_c_report_unknown_timer_node, msg_c_report_unknown_trace_node, msg_c_report_host, msg_c_report_mic, msg_c_report_title, msg_c_report_seconds, msg_c_report_bytes, msg_c_report_cpu_time, msg_c_report_mic_time, msg_c_report_tag, msg_c_report_from_file, msg_c_report_file, msg_c_report_line, msg_c_report_cpu_to_mic_data, msg_c_report_mic_to_cpu_data, msg_c_report_offload, msg_c_report_w_tag, msg_c_report_state, msg_c_report_start, msg_c_report_init, msg_c_report_logical_card, msg_c_report_physical_card, msg_c_report_register, msg_c_report_init_func, msg_c_report_create_buf_host, msg_c_report_create_buf_mic, msg_c_report_send_pointer_data, msg_c_report_sent_pointer_data, msg_c_report_gather_copyin_data, msg_c_report_copyin_data, msg_c_report_state_signal, msg_c_report_signal, msg_c_report_wait, msg_c_report_compute, msg_c_report_receive_pointer_data, msg_c_report_received_pointer_data, msg_c_report_start_target_func, msg_c_report_var, msg_c_report_scatter_copyin_data, msg_c_report_gather_copyout_data, msg_c_report_scatter_copyout_data, msg_c_report_copyout_data, msg_c_report_unregister, msg_c_report_destroy, msg_c_report_myoinit, msg_c_report_myoregister, msg_c_report_myofini, msg_c_report_mic_myo_shared, msg_c_report_mic_myo_fptr, msg_c_report_myosharedmalloc, msg_c_report_myosharedfree, msg_c_report_myosharedalignedmalloc, msg_c_report_myosharedalignedfree, msg_c_report_myoacquire, msg_c_report_myorelease, msg_c_report_host_total_offload_time, msg_c_report_host_initialize, msg_c_report_host_target_acquire, msg_c_report_host_wait_deps, msg_c_report_host_setup_buffers, msg_c_report_host_alloc_buffers, msg_c_report_host_setup_misc_data, msg_c_report_host_alloc_data_buffer, msg_c_report_host_send_pointers, msg_c_report_host_gather_inputs, msg_c_report_host_map_in_data_buffer, msg_c_report_host_unmap_in_data_buffer, msg_c_report_host_start_compute, msg_c_report_host_wait_compute, msg_c_report_host_start_buffers_reads, msg_c_report_host_scatter_outputs, msg_c_report_host_map_out_data_buffer, msg_c_report_host_unmap_out_data_buffer, msg_c_report_host_wait_buffers_reads, msg_c_report_host_destroy_buffers, msg_c_report_target_total_time,

msg_c_report_target_descriptor_setup,
msg_c_report_target_func_lookup, msg_c_report_target_func_time, msg_c_report_target_scatter_inputs, msg_c_report_target_add_buffer_refs,
msg_c_report_target_compute, msg_c_report_target_gather_outputs, msg_c_report_target_release_buffer_refs,
msg_c_coi_pipeline_max_number,
msg_c_ranges_dont_match, msg_c_destination_is_over, msg_c_slice_of_noncont_array, msg_c_pointer_array_mismatch,
lastMsg = 152, firstMsg = 1 }

Variables

- static char const * MESSAGE_TABLE_NAME []

7.16.1 Macro Definition Documentation

#define MESSAGE_TABLE_NAME __liboffload_message_table

Definition at line 170 of file liboffload_msg.h.
Referenced by offload_get_message_str(), and write_message().

7.16.2 Enumeration Type Documentation

anonymous enum

Enumerator

__dummy__
msg_c_device_is_not_available
msg_c_invalid_device_number
msg_c_send_func_ptr
msg_c_receive_func_ptr
msg_c_offload_malloc
msg_c_offload1
msg_c_unknown_var_type
msg_c_invalid_env_var_value
msg_c_invalid_env_var_int_value
msg_c_invalid_env_report_value
msg_c_offload_signaled1
msg_c_offload_signaled2
msg_c_myowrapper_checkresult
msg_c_myotarget_checkresult
msg_c_offload_descriptor_offload
msg_c_merge_var_descs1
msg_c_merge_var_descs2
msg_c_mic_parse_env_var_list1
msg_c_mic_parse_env_var_list2
msg_c_mic_process_exit_ret
msg_c_mic_process_exit_sig
msg_c_mic_process_exit
msg_c_mic_init3
msg_c_mic_init4
msg_c_mic_init5

msg_c_mic_init6
msg_c_no_static_var_data
msg_c_no_ptr_data
msg_c_get_engine_handle
msg_c_get_engine_index
msg_c_process_create
msg_c_process_get_func_handles
msg_c_process_wait_shutdown
msg_c_process_proxy_flush
msg_c_load_library
msg_c_pipeline_create
msg_c_pipeline_run_func
msg_c_pipeline_start_run_funcs
msg_c_buf_create
msg_c_buf_create_out_of_mem
msg_c_buf_create_from_mem
msg_c_buf_destroy
msg_c_buf_map
msg_c_buf_unmap
msg_c_buf_read
msg_c_buf_write
msg_c_buf_copy
msg_c_buf_get_address
msg_c_buf_add_ref
msg_c_buf_release_ref
msg_c_buf_set_state
msg_c_event_wait
msg_c_zero_or_neg_ptr_len
msg_c_zero_or_neg_transfer_size
msg_c_bad_ptr_mem_range
msg_c_different_src_and_dstn_sizes
msg_c_non_contiguous_dope_vector
msg_c_omp_invalid_device_num_env
msg_c_omp_invalid_device_num
msg_c_unknown_binary_type
msg_c_multiple_target_exes
msg_c_no_target_exe
msg_c_report_unknown_timer_node
msg_c_report_unknown_trace_node
msg_c_report_host
msg_c_report_mic
msg_c_report_title
msg_c_report_seconds
msg_c_report_bytes
msg_c_report_cpu_time

msg_c_report_mic_time
msg_c_report_tag
msg_c_report_from_file
msg_c_report_file
msg_c_report_line
msg_c_report_cpu_to_mic_data
msg_c_report_mic_to_cpu_data
msg_c_report_offload
msg_c_report_w_tag
msg_c_report_state
msg_c_report_start
msg_c_report_init
msg_c_report_logical_card
msg_c_report_physical_card
msg_c_report_register
msg_c_report_init_func
msg_c_report_create_buf_host
msg_c_report_create_buf_mic
msg_c_report_send_pointer_data
msg_c_report_sent_pointer_data
msg_c_report_gather_copyin_data
msg_c_report_copyin_data
msg_c_report_state_signal
msg_c_report_signal
msg_c_report_wait
msg_c_report_compute
msg_c_report_receive_pointer_data
msg_c_report_received_pointer_data
msg_c_report_start_target_func
msg_c_report_var
msg_c_report_scatter_copyin_data
msg_c_report_gather_copyout_data
msg_c_report_scatter_copyout_data
msg_c_report_copyout_data
msg_c_report_unregister
msg_c_report_destroy
msg_c_report_myoinit
msg_c_report_myoregister
msg_c_report_myofini
msg_c_report_mic_myo_shared
msg_c_report_mic_myo_fptr
msg_c_report_myosharedmalloc
msg_c_report_myosharedfree
msg_c_report_myosharedalignedmalloc
msg_c_report_myosharedalignedfree

msg_c_report_myoacquire

msg_c_report_myorelease

msg_c_report_host_total_offload_time

msg_c_report_host_initialize

msg_c_report_host_target_acquire

msg_c_report_host_wait_deps

msg_c_report_host_setup_buffers

msg_c_report_host_alloc_buffers

msg_c_report_host_setup_misc_data

msg_c_report_host_alloc_data_buffer

msg_c_report_host_send_pointers

msg_c_report_host_gather_inputs

msg_c_report_host_map_in_data_buffer

msg_c_report_host_unmap_in_data_buffer

msg_c_report_host_start_compute

msg_c_report_host_wait_compute

msg_c_report_host_start_buffers_reads

msg_c_report_host_scatter_outputs

msg_c_report_host_map_out_data_buffer

msg_c_report_host_unmap_out_data_buffer

msg_c_report_host_wait_buffers_reads

msg_c_report_host_destroy_buffers

msg_c_report_target_total_time

msg_c_report_target_descriptor_setup

msg_c_report_target_func_lookup

msg_c_report_target_func_time

msg_c_report_target_scatter_inputs

msg_c_report_target_add_buffer_refs

msg_c_report_target_compute

msg_c_report_target_gather_outputs

msg_c_report_target_release_buffer_refs

msg_c_coi_pipeline_max_number

msg_c_ranges_dont_match

msg_c_destination_is_over

msg_c_slice_of_noncont_array

msg_c_pointer_array_mismatch

lastMsg

firstMsg

Definition at line 11 of file liboffload_msg.h.

7.16.3 Variable Documentation

char const* MESSAGE_TABLE_NAME[] `[static]`

Definition at line 173 of file liboffload_msg.h.

7.17 mic_lib.f90 File Reference

Data Types

- module mic_lib
- type mic_lib::offload_status
- interface mic_lib::offload_number_of_devices
- interface mic_lib::offload_signaled
- interface mic_lib::offload_report
- interface mic_lib::offload_get_device_number
- interface mic_lib::offload_get_physical_device_number
- interface mic_lib::omp_set_num_threads_target
- interface mic_lib::omp_get_max_threads_target
- interface mic_lib::omp_get_num_procs_target
- interface mic_lib::omp_set_dynamic_target
- interface mic_lib::omp_get_dynamic_target
- interface mic_lib::omp_set_nested_target
- interface mic_lib::omp_get_nested_target
- interface mic_lib::omp_set_schedule_target
- interface mic_lib::omp_get_schedule_target
- interface mic_lib::omp_init_lock_target
- interface mic_lib::omp_destroy_lock_target
- interface mic_lib::omp_set_lock_target
- interface mic_lib::omp_unset_lock_target
- interface mic_lib::omp_test_lock_target
- interface mic_lib::omp_init_nest_lock_target
- interface mic_lib::omp_destroy_nest_lock_target
- interface mic_lib::omp_set_nest_lock_target
- interface mic_lib::omp_unset_nest_lock_target
- interface mic_lib::omp_test_nest_lock_target
- interface mic_lib::kmp_set_stacksize_target
- interface mic_lib::kmp_get_stacksize_target
- interface mic_lib::kmp_set_stacksize_s_target
- interface mic_lib::kmp_get_stacksize_s_target
- interface mic_lib::kmp_set_blocktime_target
- interface mic_lib::kmp_get_blocktime_target
- interface mic_lib::kmp_set_library_serial_target
- interface mic_lib::kmp_set_library_turnaround_target
- interface mic_lib::kmp_set_library_throughput_target
- interface mic_lib::kmp_set_library_target
- interface mic_lib::kmp_get_library_target
- interface mic_lib::kmp_set_defaults_target
- interface mic_lib::kmp_create_affinity_mask_target
- interface mic_lib::kmp_destroy_affinity_mask_target
- interface mic_lib::kmp_set_affinity_target
- interface mic_lib::kmp_get_affinity_target
- interface mic_lib::kmp_get_affinity_max_proc_target
- interface mic_lib::kmp_set_affinity_mask_proc_target
- interface mic_lib::kmp_unset_affinity_mask_proc_target
- interface mic_lib::kmp_get_affinity_mask_proc_target

7.18 offload.h File Reference

```
#include <stddef.h>
#include <omp.h>
```

Classes

- struct _Offload_status
- struct omp_lock_target_t
- struct omp_nest_lock_target_t
- struct kmp_affinity_mask_target_t

Macros

- #define TARGET_ATTRIBUTE __declspec(target(mic))
- #define DEFAULT_TARGET_TYPE TARGET_MIC
- #define DEFAULT_TARGET_NUMBER 0
- #define OFFLOAD_STATUS_INIT(x) ((x).result = OFFLOAD_DISABLED)
- #define OFFLOAD_STATUS_INITIALIZER { OFFLOAD_DISABLED, -1, 0, 0 }

Typedefs

- typedef enum TARGET_TYPE TARGET_TYPE

Enumerations

- enum TARGET_TYPE { TARGET_NONE, TARGET_HOST, TARGET_MIC }
- enum _Offload_result {
  OFFLOAD_SUCCESS = 0, OFFLOAD_DISABLED, OFFLOAD_UNAVAILABLE, OFFLOAD_OUT_OF_MEMORY,
  OFFLOAD_PROCESS_DIED, OFFLOAD_ERROR }

Functions

- int _Offload_number_of_devices (void)
- int _Offload_get_device_number (void)
- int _Offload_get_physical_device_number (void)
- void * _Offload_shared_malloc (size_t size)
- void _Offload_shared_free (void *ptr)
- void * _Offload_shared_aligned_malloc (size_t size, size_t align)
- void _Offload_shared_aligned_free (void *ptr)
- int _Offload_signaled (int index, void *signal)
- void _Offload_report (int val)
- void omp_set_default_device (int num)
- int omp_get_default_device (void)
- int omp_get_num_devices (void)
- void omp_set_num_threads_target (TARGET_TYPE target_type, int target_number, int num_threads)
- int omp_get_max_threads_target (TARGET_TYPE target_type, int target_number)
- int omp_get_num_procs_target (TARGET_TYPE target_type, int target_number)
- void omp_set_dynamic_target (TARGET_TYPE target_type, int target_number, int num_threads)
- int omp_get_dynamic_target (TARGET_TYPE target_type, int target_number)
- void omp_set_nested_target (TARGET_TYPE target_type, int target_number, int nested)
- int omp_get_nested_target (TARGET_TYPE target_type, int target_number)
- void omp_set_schedule_target (TARGET_TYPE target_type, int target_number, omp_sched_t kind, int modifier)
- void omp_get_schedule_target (TARGET_TYPE target_type, int target_number, omp_sched_t *kind, int *modifier)
- void omp_init_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- void omp_destroy_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- void omp_set_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- void omp_unset_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- int omp_test_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)

- void omp_init_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void omp_destroy_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void omp_set_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void omp_unset_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- int omp_test_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void kmp_set_stacksize_target (TARGET_TYPE target_type, int target_number, int size)
- int kmp_get_stacksize_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_stacksize_s_target (TARGET_TYPE target_type, int target_number, size_t size)
- size_t kmp_get_stacksize_s_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_blocktime_target (TARGET_TYPE target_type, int target_number, int time)
- int kmp_get_blocktime_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_serial_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_turnaround_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_throughput_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_target (TARGET_TYPE target_type, int target_number, int mode)
- int kmp_get_library_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_defaults_target (TARGET_TYPE target_type, int target_number, char const *defaults)
- void kmp_create_affinity_mask_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- void kmp_destroy_affinity_mask_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- int kmp_set_affinity_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- int kmp_get_affinity_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- int kmp_get_affinity_max_proc_target (TARGET_TYPE target_type, int target_number)
- int kmp_set_affinity_mask_proc_target (TARGET_TYPE target_type, int target_number, int proc, kmp_affinity_mask_target_t *mask)
- int kmp_unset_affinity_mask_proc_target (TARGET_TYPE target_type, int target_number, int proc, kmp_affinity_mask_target_t *mask)
- int kmp_get_affinity_mask_proc_target (TARGET_TYPE target_type, int target_number, int proc, kmp_affinity_mask_target_t *mask)

7.18.1 Macro Definition Documentation

#define DEFAULT_TARGET_NUMBER 0

Definition at line 48 of file offload.h.

#define DEFAULT_TARGET_TYPE TARGET_MIC

Definition at line 43 of file offload.h.

#define OFFLOAD_STATUS_INIT(*x*) ((x).result = OFFLOAD_DISABLED)

Definition at line 69 of file offload.h.

#define OFFLOAD_STATUS_INITIALIZER { OFFLOAD_DISABLED, -1, 0, 0 }

Definition at line 72 of file offload.h.

#define TARGET_ATTRIBUTE __declspec(target(mic))

Definition at line 29 of file offload.h.

7.18.2 Typedef Documentation

typedef enum TARGET_TYPE TARGET_TYPE

7.18.3 Enumeration Type Documentation

enum _Offload_result

Enumerator

OFFLOAD_SUCCESS

OFFLOAD_DISABLED

OFFLOAD_UNAVAILABLE

OFFLOAD_OUT_OF_MEMORY

OFFLOAD_PROCESS_DIED

OFFLOAD_ERROR

Definition at line 53 of file offload.h.

enum TARGET_TYPE

Enumerator

TARGET_NONE

TARGET_HOST

TARGET_MIC

Definition at line 34 of file offload.h.

7.18.4 Function Documentation

int _Offload_get_device_number (void)

Definition at line 4308 of file offload_host.cpp.

int _Offload_get_physical_device_number (void)

Definition at line 4313 of file offload_host.cpp.

int _Offload_number_of_devices (void)

Definition at line 4302 of file offload_host.cpp.

void _Offload_report (int *val*)

Definition at line 4339 of file offload_host.cpp.

void _Offload_shared_aligned_free (void * *ptr*)

Definition at line 693 of file offload_myo_host.cpp.

void* _Offload_shared_aligned_malloc (size_t *size,* size_t *align*)

Definition at line 678 of file offload_myo_host.cpp.

void _Offload_shared_free (void * *ptr*)

Definition at line 666 of file offload_myo_host.cpp.
Referenced by __intel_cilk_for_32_offload(), and __intel_cilk_for_64_offload().

void* _Offload_shared_malloc (size_t *size*)

Definition at line 654 of file offload_myo_host.cpp.
Referenced by __intel_cilk_for_32_offload(), and __intel_cilk_for_64_offload().

int _Offload_signaled (int *index,* void * *signal*)

Definition at line 4318 of file offload_host.cpp.

void kmp_create_affinity_mask_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 609 of file offload_omp_host.cpp.

void kmp_destroy_affinity_mask_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 632 of file offload_omp_host.cpp.

int kmp_get_affinity_mask_proc_target (TARGET_TYPE *target_type,* int *target_number,* int *proc,* kmp_affinity_mask_target_t * *mask*)

Definition at line 812 of file offload_omp_host.cpp.

int kmp_get_affinity_max_proc_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 721 of file offload_omp_host.cpp.

int kmp_get_affinity_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 688 of file offload_omp_host.cpp.

int kmp_get_blocktime_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 517 of file offload_omp_host.cpp.

int kmp_get_library_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 575 of file offload_omp_host.cpp.

size_t kmp_get_stacksize_s_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 498 of file offload_omp_host.cpp.

int kmp_get_stacksize_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 479 of file offload_omp_host.cpp.

int kmp_set_affinity_mask_proc_target (TARGET_TYPE *target_type,* int *target_number,* int *proc,* kmp_affinity_mask_target_t * *mask*)

Definition at line 730 of file offload_omp_host.cpp.

int kmp_set_affinity_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 655 of file offload_omp_host.cpp.

void kmp_set_blocktime_target (TARGET_TYPE *target_type,* int *target_number,* int *time*)

Definition at line 507 of file offload_omp_host.cpp.

void kmp_set_defaults_target (TARGET_TYPE *target_type,* int *target_number,* char const * *defaults*)

Definition at line 584 of file offload_omp_host.cpp.

void kmp_set_library_serial_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 526 of file offload_omp_host.cpp.

void kmp_set_library_target (TARGET_TYPE *target_type,* int *target_number,* int *mode*)

Definition at line 565 of file offload_omp_host.cpp.

void kmp_set_library_throughput_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 552 of file offload_omp_host.cpp.

void kmp_set_library_turnaround_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 539 of file offload_omp_host.cpp.

void kmp_set_stacksize_s_target (TARGET_TYPE *target_type,* int *target_number,* size_t *size*)

Definition at line 488 of file offload_omp_host.cpp.

void kmp_set_stacksize_target (TARGET_TYPE *target_type,* int *target_number,* int *size*)

Definition at line 469 of file offload_omp_host.cpp.

int kmp_unset_affinity_mask_proc_target (TARGET_TYPE *target_type,* int *target_number,* int *proc,* kmp_affinity_mask_target_t * *mask*)

Definition at line 771 of file offload_omp_host.cpp.

void omp_destroy_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 238 of file offload_omp_host.cpp.

void omp_destroy_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 365 of file offload_omp_host.cpp.

int omp_get_default_device (void)

Definition at line 24 of file offload_omp_host.cpp.

int omp_get_dynamic_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 123 of file offload_omp_host.cpp.

int omp_get_max_threads_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 95 of file offload_omp_host.cpp.

int omp_get_nested_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 142 of file offload_omp_host.cpp.

int omp_get_num_devices (void)

Definition at line 29 of file offload_omp_host.cpp.

int omp_get_num_procs_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 104 of file offload_omp_host.cpp.

void omp_get_schedule_target (TARGET_TYPE *target_type,* int *target_number,* omp_sched_t * *kind,* int * *modifier*)

Definition at line 182 of file offload_omp_host.cpp.

void omp_init_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 215 of file offload_omp_host.cpp.

void omp_init_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 342 of file offload_omp_host.cpp.

void omp_set_default_device (int *num*)

Definition at line 17 of file offload_omp_host.cpp.

void omp_set_dynamic_target (TARGET_TYPE *target_type,* int *target_number,* int *num_threads*)

Definition at line 113 of file offload_omp_host.cpp.

void omp_set_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 261 of file offload_omp_host.cpp.

void omp_set_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 388 of file offload_omp_host.cpp.

void omp_set_nested_target (TARGET_TYPE *target_type,* int *target_number,* int *nested*)

Definition at line 132 of file offload_omp_host.cpp.

void omp_set_num_threads_target (TARGET_TYPE *target_type,* int *target_number,* int *num_threads*)

Definition at line 85 of file offload_omp_host.cpp.

void omp_set_schedule_target (TARGET_TYPE *target_type,* int *target_number,* omp_sched_t *kind,* int *modifier*)

Definition at line 151 of file offload_omp_host.cpp.

int omp_test_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 307 of file offload_omp_host.cpp.

int omp_test_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 434 of file offload_omp_host.cpp.

void omp_unset_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 284 of file offload_omp_host.cpp.

void omp_unset_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 411 of file offload_omp_host.cpp.

7.19 offload_common.cpp File Reference

```
#include "offload_common.h"
```

Functions

- void * OFFLOAD_MALLOC (size_t size, size_t align)

7.19.1 Function Documentation

void* OFFLOAD_MALLOC (size_t *size,* size_t *align*)

Definition at line 147 of file offload_common.cpp.

7.20 offload_common.h File Reference

The parts of the runtime library common to host and target.

```
   #include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <memory.h>
#include "offload.h"
#include "offload_table.h"
#include "offload_trace.h"
#include "offload_timer.h"
#include "offload_util.h"
#include "cean_util.h"
#include "dv_util.h"
#include "liboffload_error_codes.h"
#include <stdarg.h>
```

Classes

- struct VarDesc

  An Offload Variable descriptor.

- struct VarDesc2

  Auxiliary struct used when -g is enabled that holds variable names.

- struct VarDesc3
- class Marshaller
- struct FunctionDescriptor

Macros

- #define OFFLOAD_DO_TRACE (offload_report_level == 3)
- #define OFFLOAD_DEBUG_PRINT_PREFIX() printf("%s%d: ", prefix, mic_index);
- #define OFFLOAD_TRACE(trace_level,...)
- #define OFFLOAD_DEBUG_LOG(level,...)
- #define OFFLOAD_DEBUG_DUMP_BYTES(level, a, b)
- #define OFFLOAD_PREFIX(a) __offload_##a
- #define OFFLOAD_MALLOC OFFLOAD_PREFIX(malloc)
- #define OFFLOAD_FREE(a) _mm_free(a)
- #define VAR_TYPE_IS_PTR(t)
- #define VAR_TYPE_IS_SCALAR(t)
- #define VAR_TYPE_IS_DV_DATA(t)
- #define VAR_TYPE_IS_DV_DATA_SLICE(t)

Typedefs

- typedef struct OffloadDescriptor * OFFLOAD

Enumerations

- enum OffloadItemType {
  c_data = 1, c_data_ptr, c_func_ptr, c_void_ptr,
  c_string_ptr, c_dv, c_dv_data, c_dv_data_slice,
  c_dv_ptr, c_dv_ptr_data, c_dv_ptr_data_slice, c_cean_var,
  c_cean_var_ptr, c_data_ptr_array, c_func_ptr_array, c_void_ptr_array,
  c_string_ptr_array }
- enum OffloadParameterType {
  c_parameter_unknown = -1, c_parameter_nocopy, c_parameter_in, c_parameter_out,
  c_parameter_inout }

Functions

- void * OFFLOAD_MALLOC (size_t size, size_t align)

Variables

- int console_enabled
- int offload_report_level
- const char * prefix
- int offload_number
- int mic_index
- const int flag_align_is_array = 6
- const int flag_alloc_if_is_array = 7
- const int flag_free_if_is_array = 8
- const int flag_extent_start_is_scalar = 9
- const int flag_extent_start_is_array = 10
- const int flag_extent_elements_is_scalar = 11
- const int flag_extent_elements_is_array = 12
- const int flag_into_start_is_scalar = 13
- const int flag_into_start_is_array = 14
- const int flag_into_elements_is_scalar = 15
- const int flag_into_elements_is_array = 16
- const int flag_alloc_start_is_scalar = 17
- const int flag_alloc_start_is_array = 18
- const int flag_alloc_elements_is_scalar = 19
- const int flag_alloc_elements_is_array = 20

7.20.1 Detailed Description

The parts of the runtime library common to host and target.

Definition in file offload_common.h.

7.20.2 Macro Definition Documentation

#define OFFLOAD_DEBUG_DUMP_BYTES(*level, a, b*)

Definition at line 111 of file offload_common.h.

Referenced by OffloadDescriptor::gather_copyout_data(), and OffloadDescriptor::scatter_copyin_data().

#define OFFLOAD_DEBUG_LOG(*level, ...*)

Definition at line 110 of file offload_common.h.

#define OFFLOAD_DEBUG_PRINT_PREFIX( ) printf("%s%d: ", prefix, mic_index);

Definition at line 68 of file offload_common.h.

#define OFFLOAD_DO_TRACE (offload_report_level == 3)

Definition at line 40 of file offload_common.h.

#define OFFLOAD_FREE(*a*) _mm_free(a)

Definition at line 120 of file offload_common.h.

#define OFFLOAD_MALLOC OFFLOAD_PREFIX(malloc)

Definition at line 119 of file offload_common.h.

#define OFFLOAD_PREFIX(*a*) __offload_##a

Definition at line 117 of file offload_common.h.

#define OFFLOAD_TRACE(*trace_level, ...*)

Value:

```
if (console_enabled >= trace_level) { \
        OFFLOAD_DEBUG_PRINT_PREFIX(); \
        printf(__VA_ARGS__); \
        fflush(NULL); \
    }
```

Definition at line 72 of file offload_common.h.

Referenced by OffloadDescriptor::alloc_ptr_data(), OffloadDescriptor::find_ptr_data(), generate_mem_ranges(), generate_mem_ranges_one_rank(), generate_one_range(), OffloadDescriptor::init_static_ptr_data(), OffloadDescriptor::merge_var_descs(), OffloadDescriptor::offload_stack_memory_manager(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::scatter_copyout_data(), and OffloadDescriptor::setup_descriptors().

#define VAR_TYPE_IS_DV_DATA(*t*)

Value:

```
((t) == c_dv_data || \
                                (t) == c_dv_ptr_data)
```

Definition at line 159 of file offload_common.h.

Referenced by offload_get_src_base(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyin_data(), and OffloadDescriptor::send_pointer_data().

#define VAR_TYPE_IS_DV_DATA_SLICE(*t*)

Value:

```
((t) == c_dv_data_slice || \
                                        (t) == c_dv_ptr_data_slice)
```

Definition at line 162 of file offload_common.h.

Referenced by offload_get_src_base(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyin_data(), OffloadDescriptor::send_pointer_data(), and OffloadDescriptor::setup_descriptors().

#define VAR_TYPE_IS_PTR(*t*)

Value:

```
((t) == c_string_ptr || \
                              (t) == c_data_ptr || \
                              (t) == c_cean_var_ptr || \
                              (t) == c_dv_ptr)
```

Definition at line 149 of file offload_common.h.

Referenced by offload_get_src_base(), OffloadDescriptor::receive_pointer_data(), and OffloadDescriptor::send_pointer_data().

#define VAR_TYPE_IS_SCALAR(*t*)

Value:

```
((t) == c_data || \
                                    (t) == c_void_ptr || \
                                    (t) == c_cean_var || \
                                    (t) == c_dv)
```

Definition at line 154 of file offload_common.h.

Referenced by offload_get_src_base(), OffloadDescriptor::receive_pointer_data(), and OffloadDescriptor::send_pointer_data().

7.20.3 Typedef Documentation

typedef struct OffloadDescriptor* OFFLOAD

Definition at line 442 of file offload_common.h.

7.20.4 Enumeration Type Documentation

enum OffloadItemType

Enumerator

c_data Plain data.

c_data_ptr Pointer data.

c_func_ptr Function pointer.

c_void_ptr void*

c_string_ptr C string.

c_dv Dope vector variable.

c_dv_data Dope-vector data.

c_dv_data_slice Dope-vector data's slice.

c_dv_ptr Dope-vector variable pointer.

c_dv_ptr_data Dope-vector pointer data.

c_dv_ptr_data_slice Dope-vector pointer data's slice.

c_cean_var CEAN variable.

c_cean_var_ptr Pointer to CEAN variable.

c_data_ptr_array Pointer to data pointer array.

c_func_ptr_array Pointer to function pointer array.

c_void_ptr_array Pointer to void* pointer array.

c_string_ptr_array Pointer to char* pointer array.

Definition at line 129 of file offload_common.h.

enum OffloadParameterType

Enumerator

c_parameter_unknown Unknown clause.

c_parameter_nocopy Variable listed in "nocopy" clause.

c_parameter_in Variable listed in "in" clause.

c_parameter_out Variable listed in "out" clause.

c_parameter_inout Variable listed in "inout" clause.

Definition at line 167 of file offload_common.h.

7.20.5 Function Documentation

void* OFFLOAD_MALLOC (size_t *size,* size_t *align*)

Definition at line 147 of file offload_common.cpp.

7.20.6 Variable Documentation

int console_enabled

Definition at line 82 of file offload_host.cpp.
Referenced by __offload_console_trace(), __offload_init_library_once(), Engine::init_device(), OffloadDescriptor::offload(), Marshaller::receive_func_ptr(), Marshaller::send_func_ptr(), server_init(), and OffloadDescriptor::setup_misc_data().

const int flag_align_is_array = 6

Definition at line 313 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_alloc_elements_is_array = 20

Definition at line 327 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_alloc_elements_is_scalar = 19

Definition at line 326 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_alloc_if_is_array = 7

Definition at line 314 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_alloc_start_is_array = 18

Definition at line 325 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_alloc_start_is_scalar = 17

Definition at line 324 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_extent_elements_is_array = 12

Definition at line 319 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_extent_elements_is_scalar = 11

Definition at line 318 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_extent_start_is_array = 10

Definition at line 317 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_extent_start_is_scalar = 9

Definition at line 316 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_free_if_is_array = 8

Definition at line 315 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_into_elements_is_array = 16

Definition at line 323 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_into_elements_is_scalar = 15

Definition at line 322 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_into_start_is_array = 14

Definition at line 321 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

const int flag_into_start_is_scalar = 13

Definition at line 320 of file offload_common.h.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array(), and OffloadDescriptor::setup_descriptors().

int mic_index

Definition at line 56 of file offload_target.cpp.
Referenced by omp_get_default_device(), and server_init().

int offload_number

Definition at line 83 of file offload_host.cpp.
Referenced by OffloadDescriptor::offload().

int offload_report_level

Definition at line 29 of file offload_target.cpp.
Referenced by __offload_init_library_once(), Engine::init_device(), OffloadDescriptor::offload(), server_init(), and OffloadDescriptor::setup_misc_data().

const char* prefix

Definition at line 81 of file offload_host.cpp.

7.21 offload_engine.cpp File Reference

```
#include "offload_engine.h"
#include <signal.h>
#include <errno.h>
#include <algorithm>
#include <vector>
#include "offload_host.h"
#include "offload_table.h"
```

Classes

- struct Thread

Functions

- static bool target_entry_cmp (const VarList::BufEntry &l, const VarList::BufEntry &r)
- static bool host_entry_cmp (const VarTable::Entry *l, const VarTable::Entry *r)

7.21.1 Function Documentation

static bool host_entry_cmp (const VarTable::Entry * *l*, const VarTable::Entry * *r*) [static]

Definition at line 242 of file offload_engine.cpp.
Referenced by Engine::init_ptr_data().

static bool target_entry_cmp (const VarList::BufEntry & *l*, const VarList::BufEntry & *r*) [static]

Definition at line 232 of file offload_engine.cpp.
Referenced by Engine::init_ptr_data().

7.22 offload_engine.h File Reference

```
#include <limits.h>
#include <list>
#include <set>
#include <map>
#include "offload_common.h"
#include "coi/coi_client.h"
```

Classes

- class MemRange
- class PtrData
- class AutoData

- struct TargetImage
- struct PersistData
- struct Engine

Macros

- #define check_result(res, tag,...)

Typedefs

- typedef std::list< PtrData * > PtrDataList
- typedef std::set< AutoData > AutoSet
- typedef std::list< TargetImage > TargetImageList
- typedef std::list< PersistData > PersistDataList

7.22.1 Macro Definition Documentation

#define check_result(*res, tag, ...*)

Value:

```
{ \
        if (res == COI_PROCESS_DIED) { \
            fini_process(true); \
            exit(1); \
        } \
        if (res != COI_SUCCESS) { \
            __liboffload_error_support(tag, __VA_ARGS__); \
            exit(1); \
        } \
    }
```

Definition at line 234 of file offload_engine.h.

Referenced by Engine::get_pipeline(), Engine::init_device(), Engine::init_process(), Engine::init_ptr_data(), and Engine::load_libraries().

7.22.2 Typedef Documentation

typedef std::set<AutoData> AutoSet

Definition at line 184 of file offload_engine.h.

typedef std::list<PersistData> PersistDataList

Definition at line 227 of file offload_engine.h.

typedef std::list<PtrData*> PtrDataList

Definition at line 138 of file offload_engine.h.

typedef std::list<TargetImage> TargetImageList

Definition at line 207 of file offload_engine.h.

7.23 offload_env.cpp File Reference

```
#include "offload_env.h"
#include <string.h>
#include <ctype.h>
#include "offload_util.h"
#include "liboffload_error_codes.h"
```

7.24 offload_env.h File Reference

```
#include <list>
```

Classes

- struct MicEnvVar
- struct MicEnvVar::VarValue
- struct MicEnvVar::CardEnvVars

Enumerations

- enum MicEnvVarKind { c_no_mic, c_mic_var, c_mic_card_var, c_mic_card_env }

7.24.1 Enumeration Type Documentation

enum MicEnvVarKind

Enumerator

c_no_mic

c_mic_var

c_mic_card_var

c_mic_card_env

Definition at line 18 of file offload_env.h.

7.25 offload_host.cpp File Reference

```
#include "offload_host.h"
#include <malloc.h>
#include <alloca.h>
#include <elf.h>
#include <errno.h>
#include <fcntl.h>
#include <stdlib.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <algorithm>
#include <bitset>
```

Macros

- #define PATH_SEPARATOR ":"
- #define GET_OFFLOAD_NUMBER(timer_data) timer_data? timer_data->offload_number : 0

Functions

- static void __offload_init_library_once (void)
- static void __offload_fini_library (void)
- static char * offload_get_src_base (void *ptr, uint8_t type)
- void get_arr_desc_numbers (const arr_desc *ap, int64_t el_size, int64_t &offset, int64_t &size, int &el_number, CeanReadRanges *&ptr_ranges)
- arr_desc * make_arr_desc (void *ptr_val, int64_t extent_start_val, int64_t extent_elements_val, int64_t size)
- int __offload_init_library (void)

- void __offload_register_image (const void *target_image)
- void __offload_unregister_image (const void *target_image)
- void __offload_console_trace (int level)
- int _Offload_number_of_devices (void)
- int _Offload_get_device_number (void)
- int _Offload_get_physical_device_number (void)
- int _Offload_signaled (int index, void *signal)
- void _Offload_report (int val)
- void __dbg_target_so_loaded ()
- void __dbg_target_so_unloaded ()

Variables

- const char * prefix
- int console_enabled = 0
- int offload_number = 0
- static const char * htrace_envname = "H_TRACE"
- static const char * offload_report_envname = "OFFLOAD_REPORT"
- static char * timer_envname = "H_TIME"
- static const char * vardesc_direction_as_string []
- static const char * vardesc_type_as_string []
- Engine * mic_engines = 0
- uint32_t mic_engines_total = 0
- pthread_key_t mic_thread_key
- MicEnvVar mic_env_vars
- uint64_t cpu_frequency = 0
- uint32_t mic_stack_size = 12 * 1024 * 1024
- uint64_t mic_buffer_size = 0
- char * mic_library_path = 0
- bool mic_proxy_io = true
- char * mic_proxy_fs_root = 0
- uint64_t __offload_use_2mb_buffers = 0xffffffffffffffffULL
- static const char * mic_use_2mb_buffers_envname
- static uint64_t __offload_use_async_buffer_write = 2 * 1024 * 1024
- static const char * mic_use_async_buffer_write_envname
- static uint64_t __offload_use_async_buffer_read = 2 * 1024 * 1024
- static const char * mic_use_async_buffer_read_envname
- OffloadInitType __offload_init_type = c_init_on_offload_all
- static const char * offload_init_envname = "OFFLOAD_INIT"
- static bool __offload_active_wait = true
- static const char * offload_active_wait_envname = "OFFLOAD_ACTIVE_WAIT"
- int __omp_device_num = 0
- static const char * omp_device_num_envname = "OMP_DEFAULT_DEVICE"
- static bool __target_libs
- static TargetImageList __target_libs_list
- static mutex_t __target_libs_lock
- static mutex_t stack_alloc_lock
- TargetImage * __target_exe
- int __dbg_is_attached = 0
- int __dbg_target_id = -1
- pid_t __dbg_target_so_pid = -1
- char __dbg_target_exe_name [MAX_TARGET_NAME] = {0}
- const int __dbg_api_major_version = 1
- const int __dbg_api_minor_version = 0

7.25.1 Macro Definition Documentation

#define GET_OFFLOAD_NUMBER(*timer_data*) timer_data? timer_data->offload_number : 0

Definition at line 43 of file offload_host.cpp.

Referenced by OffloadDescriptor::alloc_ptr_data(), OffloadDescriptor::compute(), OffloadDescriptor::gather-_copyin_data(), OffloadDescriptor::offload(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::send_-pointer_data(), and OffloadDescriptor::setup_misc_data().

#define PATH_SEPARATOR ":"

Definition at line 40 of file offload_host.cpp.

7.25.2 Function Documentation

void __dbg_target_so_loaded ( )

Definition at line 4355 of file offload_host.cpp.

Referenced by Engine::init_process().

void __dbg_target_so_unloaded ( )

Definition at line 4358 of file offload_host.cpp.

Referenced by Engine::fini_process().

void __offload_console_trace (int *level*)

Definition at line 4295 of file offload_host.cpp.

static void __offload_fini_library (void) [static]

Definition at line 3873 of file offload_host.cpp.

Referenced by __offload_unregister_image().

int __offload_init_library (void)

Definition at line 4171 of file offload_host.cpp.

Referenced by __offload_myoLoadLibrary_once(), __offload_register_image(), _Offload_number_of_devices(), _Offload_signaled(), OFFLOAD_TARGET_ACQUIRE(), OFFLOAD_TARGET_ACQUIRE1(), and omp_get_num_-devices().

static void __offload_init_library_once (void) [static]

Definition at line 3901 of file offload_host.cpp.

Referenced by __offload_init_library().

void __offload_register_image (const void * *target_image*)

Definition at line 4203 of file offload_host.cpp.

Referenced by offload_init().

void __offload_unregister_image (const void * *target_image*)

Definition at line 4261 of file offload_host.cpp.

Referenced by offload_fini().

int _Offload_get_device_number (void)

Definition at line 4308 of file offload_host.cpp.

int _Offload_get_physical_device_number (void)

Definition at line 4313 of file offload_host.cpp.

int _Offload_number_of_devices (void)

Definition at line 4302 of file offload_host.cpp.

void _Offload_report (int *val*)

Definition at line 4339 of file offload_host.cpp.

int _Offload_signaled (int *index,* void * *signal*)

Definition at line 4318 of file offload_host.cpp.

void get_arr_desc_numbers (const arr_desc * *ap,* int64_t *el_size,* int64_t & *offset,* int64_t & *size,* int & *el_number,* CeanReadRanges *& *ptr_ranges*)

Definition at line 3449 of file offload_host.cpp.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array().

arr_desc* make_arr_desc (void * *ptr_val,* int64_t *extent_start_val,* int64_t *extent_elements_val,* int64_t *size*)

Definition at line 3471 of file offload_host.cpp.
Referenced by OffloadDescriptor::gen_var_descs_for_pointer_array().

static char* offload_get_src_base (void * *ptr,* uint8_t *type*) [static]

Definition at line 175 of file offload_host.cpp.
Referenced by OffloadDescriptor::gather_copyin_data(), OffloadDescriptor::receive_pointer_data(), OffloadDescriptor::scatter_copyout_data(), OffloadDescriptor::send_noncontiguous_pointer_data(), and OffloadDescriptor::send_pointer_data().

7.25.3 Variable Documentation

const int __dbg_api_major_version = 1

Definition at line 4352 of file offload_host.cpp.

const int __dbg_api_minor_version = 0

Definition at line 4353 of file offload_host.cpp.

int __dbg_is_attached = 0

Definition at line 4348 of file offload_host.cpp.
Referenced by Engine::fini_process(), and Engine::init_process().

char __dbg_target_exe_name[MAX_TARGET_NAME] = {0}

Definition at line 4351 of file offload_host.cpp.
Referenced by Engine::init_process().

int __dbg_target_id = -1

Definition at line 4349 of file offload_host.cpp.
Referenced by Engine::init_process().

pid_t __dbg_target_so_pid = -1

Definition at line 4350 of file offload_host.cpp.
Referenced by Engine::init_process().

bool __offload_active_wait = true [static]

Definition at line 159 of file offload_host.cpp.
Referenced by __offload_init_library_once(), and OffloadDescriptor::offload_finish().

OffloadInitType __offload_init_type = c_init_on_offload_all

Definition at line 155 of file offload_host.cpp.
Referenced by __offload_init_library_once(), __offload_register_image(), OFFLOAD_TARGET_ACQUIRE(), and OFFLOAD_TARGET_ACQUIRE1().

uint64_t __offload_use_2mb_buffers = 0xffffffffffffffffULL

Definition at line 142 of file offload_host.cpp.
Referenced by __offload_init_library_once(), OffloadDescriptor::alloc_ptr_data(), and OffloadDescriptor::offload_stack_memory_manager().

uint64_t __offload_use_async_buffer_read = 2 * 1024 * 1024 [static]

Definition at line 150 of file offload_host.cpp.
Referenced by __offload_init_library_once(), and OffloadDescriptor::receive_pointer_data().

uint64_t __offload_use_async_buffer_write = 2 * 1024 * 1024 [static]

Definition at line 146 of file offload_host.cpp.
Referenced by __offload_init_library_once(), and OffloadDescriptor::send_pointer_data().

int __omp_device_num = 0

Definition at line 163 of file offload_host.cpp.
Referenced by __offload_init_library_once(), OFFLOAD_TARGET_ACQUIRE1(), omp_get_default_device(), and omp_set_default_device().

TargetImage* __target_exe

Definition at line 173 of file offload_host.cpp.
Referenced by Engine::init_process().

bool __target_libs [static]

Definition at line 167 of file offload_host.cpp.
Referenced by __offload_init_library(), and __offload_register_image().

TargetImageList __target_libs_list [static]

Definition at line 168 of file offload_host.cpp.
Referenced by __offload_init_library(), and __offload_register_image().

mutex_t __target_libs_lock [static]

Definition at line 169 of file offload_host.cpp.

int console_enabled = 0

Definition at line 82 of file offload_host.cpp.
Referenced by __offload_console_trace(), __offload_init_library_once(), and OffloadDescriptor::setup_misc_data().

uint64_t cpu_frequency = 0

Definition at line 121 of file offload_host.cpp.
Referenced by __offload_init_library_once().

const char* htrace_envname = "H_TRACE" [static]

Definition at line 85 of file offload_host.cpp.
Referenced by __offload_init_library_once().

uint64_t mic_buffer_size = 0

Definition at line 127 of file offload_host.cpp.
Referenced by __offload_init_library_once(), and Engine::init_process().

Engine* mic_engines = 0

Definition at line 117 of file offload_host.cpp.
Referenced by __offload_fini_library(), __offload_myoFini(), __offload_myoInit_once(), __offload_myoIsAvailable(), OFFLOAD_TARGET_ACQUIRE(), OFFLOAD_TARGET_ACQUIRE1(), ORSL::release(), ORSL::reserve(), and ORSL::try_reserve().

uint32_t mic_engines_total = 0

Definition at line 118 of file offload_host.cpp.
Referenced by __offload_fini_library(), __offload_init_library(), __offload_init_library_once(), __offload_myoFini(), __offload_myoInit_once(), __offload_myoIsAvailable(), __offload_register_image(), _Offload_number_of_devices(), _Offload_signaled(), Engine::init_device(), OFFLOAD_TARGET_ACQUIRE(), OFFLOAD_TARGET_ACQUIRE1(), omp_get_num_devices(), server_init(), and Thread::~Thread().

MicEnvVar mic_env_vars

Definition at line 120 of file offload_host.cpp.
Referenced by Engine::init_process().

char* mic_library_path = 0

Definition at line 130 of file offload_host.cpp.
Referenced by __offload_fini_library(), __offload_init_library_once(), Engine::init_process(), and Engine::load_libraries().

char* mic_proxy_fs_root = 0

Definition at line 136 of file offload_host.cpp.
Referenced by __offload_fini_library(), __offload_init_library_once(), and Engine::init_process().

bool mic_proxy_io = true

Definition at line 133 of file offload_host.cpp.
Referenced by __offload_init_library_once(), and Engine::init_process().

uint32_t mic_stack_size = 12 * 1024 * 1024

Definition at line 124 of file offload_host.cpp.
Referenced by __offload_init_library_once(), and Engine::get_pipeline().

pthread_key_t mic_thread_key

Definition at line 119 of file offload_host.cpp.
Referenced by __offload_fini_library(), __offload_init_library_once(), Engine::get_auto_vars(), and Engine::get_pipeline().

const char* mic_use_2mb_buffers_envname [static]

Initial value:

```
=
    "MIC_USE_2MB_BUFFERS"
```

Definition at line 143 of file offload_host.cpp.
Referenced by __offload_init_library_once().

const char* mic_use_async_buffer_read_envname [static]

Initial value:

```
=
    "MIC_USE_ASYNC_BUFFER_READ"
```

Definition at line 151 of file offload_host.cpp.
Referenced by __offload_init_library_once().

const char* mic_use_async_buffer_write_envname [static]

Initial value:

```
=
    "MIC_USE_ASYNC_BUFFER_WRITE"
```

Definition at line 147 of file offload_host.cpp.
Referenced by __offload_init_library_once().

const char* offload_active_wait_envname = "OFFLOAD_ACTIVE_WAIT" [static]

Definition at line 160 of file offload_host.cpp.
Referenced by __offload_init_library_once().

const char* offload_init_envname = "OFFLOAD_INIT" [static]

Definition at line 156 of file offload_host.cpp.
Referenced by __offload_init_library_once().

int offload_number = 0

Definition at line 83 of file offload_host.cpp.

const char* offload_report_envname = "OFFLOAD_REPORT" [static]

Definition at line 86 of file offload_host.cpp.
Referenced by __offload_init_library_once().

const char* omp_device_num_envname = "OMP_DEFAULT_DEVICE" [static]

Definition at line 164 of file offload_host.cpp.
Referenced by __offload_init_library_once().

const char* prefix

Definition at line 81 of file offload_host.cpp.
Referenced by __offload_init_library_once().

mutex_t stack_alloc_lock [static]

Definition at line 170 of file offload_host.cpp.

char* timer_envname = "H_TIME" [static]

Definition at line 87 of file offload_host.cpp.
Referenced by __offload_init_library_once().

const char* vardesc_direction_as_string[] [static]

Initial value:

```
= {
    "NOCOPY",
    "IN",
    "OUT",
    "INOUT"
}
```

Definition at line 90 of file offload_host.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

const char* vardesc_type_as_string[] [static]

Initial value:

```
= {
    "unknown",
    "data",
    "data_ptr",
    "func_ptr",
    "void_ptr",
    "string_ptr",
    "dv",
    "dv_data",
    "dv_data_slice",
    "dv_ptr",
    "dv_ptr_data",
    "dv_ptr_data_slice",
    "cean_var",
    "cean_var_ptr",
    "c_data_ptr_array",
    "c_func_ptr_array",
    "c_void_ptr_array",
    "c_string_ptr_array"
}
```

Definition at line 96 of file offload_host.cpp.
Referenced by OffloadDescriptor::setup_descriptors().

7.26 offload_host.h File Reference

The parts of the runtime library used only on the host.

```
#include <unistd.h>
#include "offload_common.h"
#include "offload_util.h"
#include "offload_engine.h"
#include "offload_env.h"
#include "offload_orsl.h"
#include "coi/coi_client.h"
```

Classes

- struct Image

  The target image is packed as follows.
- class OffloadDescriptor
- struct OffloadDescriptor::VarExtra
- class OffloadDescriptor::ReadArrElements< T >

Macros

- #define MAX_TARGET_NAME 512

Enumerations

- enum OffloadInitType { c_init_on_start, c_init_on_offload, c_init_on_offload_all }

Functions

- void __offload_register_image (const void *image)
- void __offload_unregister_image (const void *image)
- int __offload_init_library (void)
- void __dbg_target_so_loaded ()
- void __dbg_target_so_unloaded ()

Variables

- Engine * mic_engines
- uint32_t mic_engines_total
- pthread_key_t mic_thread_key
- MicEnvVar mic_env_vars
- uint64_t cpu_frequency
- char * mic_library_path
- uint32_t mic_stack_size
- uint64_t mic_buffer_size
- bool mic_proxy_io
- char * mic_proxy_fs_root
- uint64_t __offload_use_2mb_buffers
- OffloadInitType __offload_init_type
- int __omp_device_num
- TargetImage * __target_exe
- char __dbg_target_exe_name [MAX_TARGET_NAME]
- pid_t __dbg_target_so_pid
- int __dbg_target_id
- int __dbg_is_attached
- const int __dbg_api_major_version
- const int __dbg_api_minor_version

7.26.1 Detailed Description

The parts of the runtime library used only on the host.

Definition in file offload_host.h.

7.26.2 Macro Definition Documentation

#define MAX_TARGET_NAME 512

Definition at line 324 of file offload_host.h.

Referenced by Engine::init_process().

7.26.3 Enumeration Type Documentation

enum OffloadInitType

Enumerator

c_init_on_start

c_init_on_offload

c_init_on_offload_all

Definition at line 265 of file offload_host.h.

7.26.4 Function Documentation

void __dbg_target_so_loaded ( )

Definition at line 4355 of file offload_host.cpp.
Referenced by Engine::init_process().

void __dbg_target_so_unloaded ( )

Definition at line 4358 of file offload_host.cpp.
Referenced by Engine::fini_process().

int __offload_init_library (void)

Definition at line 4171 of file offload_host.cpp.
Referenced by __offload_myoLoadLibrary_once(), __offload_register_image(), _Offload_number_of_devices(), _Offload_signaled(), OFFLOAD_TARGET_ACQUIRE(), OFFLOAD_TARGET_ACQUIRE1(), and omp_get_num_devices().

void __offload_register_image (const void * *image*)

Definition at line 4203 of file offload_host.cpp.
Referenced by offload_init().

void __offload_unregister_image (const void * *image*)

Definition at line 4261 of file offload_host.cpp.
Referenced by offload_fini().

7.26.5 Variable Documentation

const int __dbg_api_major_version

Definition at line 338 of file offload_host.h.

const int __dbg_api_minor_version

Definition at line 341 of file offload_host.h.

int __dbg_is_attached

Definition at line 335 of file offload_host.h.

char __dbg_target_exe_name[MAX_TARGET_NAME]

Definition at line 325 of file offload_host.h.

int __dbg_target_id

Definition at line 331 of file offload_host.h.

pid_t __dbg_target_so_pid

Definition at line 328 of file offload_host.h.

OffloadInitType __offload_init_type

Definition at line 155 of file offload_host.cpp.
Referenced by __offload_init_library_once(), __offload_register_image(), OFFLOAD_TARGET_ACQUIRE(), and OFFLOAD_TARGET_ACQUIRE1().

uint64_t __offload_use_2mb_buffers

Definition at line 142 of file offload_host.cpp.
Referenced by __offload_init_library_once(), OffloadDescriptor::alloc_ptr_data(), and OffloadDescriptor::offload_stack_memory_manager().

int __omp_device_num

Definition at line 163 of file offload_host.cpp.
Referenced by __offload_init_library_once(), OFFLOAD_TARGET_ACQUIRE1(), omp_get_default_device(), and omp_set_default_device().

TargetImage* __target_exe

Definition at line 173 of file offload_host.cpp.
Referenced by Engine::init_process().

uint64_t cpu_frequency

Definition at line 121 of file offload_host.cpp.
Referenced by __offload_init_library_once().

uint64_t mic_buffer_size

Definition at line 127 of file offload_host.cpp.
Referenced by __offload_init_library_once(), and Engine::init_process().

Engine* mic_engines

Definition at line 117 of file offload_host.cpp.
Referenced by __offload_fini_library(), __offload_myoFini(), __offload_myoInit_once(), __offload_myoIsAvailable(), OFFLOAD_TARGET_ACQUIRE(), OFFLOAD_TARGET_ACQUIRE1(), ORSL::release(), ORSL::reserve(), and ORSL::try_reserve().

uint32_t mic_engines_total

Definition at line 118 of file offload_host.cpp.

MicEnvVar mic_env_vars

Definition at line 120 of file offload_host.cpp.
Referenced by Engine::init_process().

char* mic_library_path

Definition at line 130 of file offload_host.cpp.
Referenced by __offload_fini_library(), __offload_init_library_once(), Engine::init_process(), and Engine::load_libraries().

char* mic_proxy_fs_root

Definition at line 136 of file offload_host.cpp.
Referenced by __offload_fini_library(), __offload_init_library_once(), and Engine::init_process().

bool mic_proxy_io

Definition at line 133 of file offload_host.cpp.
Referenced by __offload_init_library_once(), and Engine::init_process().

uint32_t mic_stack_size

Definition at line 124 of file offload_host.cpp.
Referenced by __offload_init_library_once(), and Engine::get_pipeline().

pthread_key_t mic_thread_key

Definition at line 119 of file offload_host.cpp.
Referenced by __offload_fini_library(), __offload_init_library_once(), Engine::get_auto_vars(), and Engine::get_pipeline().

7.27 offload_myo_host.cpp File Reference

```
#include "offload_myo_host.h"
#include <errno.h>
#include <malloc.h>
#include "offload_host.h"
```

Classes

- class MyoWrapper
- struct MyoTable

Macros

- #define MYO_VERSION1 "MYO_1.0"
- #define MYO_TABLE_END_MARKER() reinterpret_cast<const char*>(0)

Typedefs

- typedef std::list< MyoTable > MyoTableList

Functions

- void __cilkrts_cilk_for_32 (void *, void *, uint32_t, int32_t)
- void __cilkrts_cilk_for_64 (void *, void *, uint64_t, int32_t)
- static void __offload_myo_shared_table_register (SharedTableEntry *entry)
- static void __offload_myo_shared_init_table_register (InitTableEntry *entry)
- static void __offload_myo_fptr_table_register (FptrTableEntry *entry)
- static void __offload_myoLoadLibrary_once (void)
- static bool __offload_myoLoadLibrary (void)
- static void __offload_myoInit_once (void)
- static bool __offload_myoInit (void)
- static bool shared_table_entries (SharedTableEntry *entry)
- static bool fptr_table_entries (FptrTableEntry *entry)
- void __offload_myoRegisterTables (InitTableEntry *init_table, SharedTableEntry *shared_table, FptrTableEntry *fptr_table)

- void __offload_myoFini (void)
- int __offload_myoIsAvailable (int target_number)
- void __offload_myoiRemoteIThunkCall (void *thunk, void *arg, int target_number)
- void * _Offload_shared_malloc (size_t size)
- void _Offload_shared_free (void *ptr)
- void * _Offload_shared_aligned_malloc (size_t size, size_t align)
- void _Offload_shared_aligned_free (void *ptr)
- void __intel_cilk_for_32_offload (int size, void(*copy_constructor)(void *, void *), int target_number, void *raddr, void *closure_object, unsigned int iters, unsigned int grain_size)
- void __intel_cilk_for_64_offload (int size, void(*copy_constructor)(void *, void *), int target_number, void *raddr, void *closure_object, uint64_t iters, uint64_t grain_size)

Variables

- static bool myo_is_available
- static MyoWrapper myo_wrapper
- static MyoTableList __myo_table_list
- static mutex_t __myo_table_lock
- static bool __myo_tables = false

7.27.1 Macro Definition Documentation

#define MYO_TABLE_END_MARKER( ) reinterpret_cast<const char*>(0)

Definition at line 33 of file offload_myo_host.cpp.

Referenced by __offload_myo_fptr_table_register(), __offload_myo_shared_init_table_register(), __offload_myo_shared_table_register(), __offload_myoInit(), fptr_table_entries(), and shared_table_entries().

#define MYO_VERSION1 "MYO_1.0"

Definition at line 20 of file offload_myo_host.cpp.

Referenced by MyoWrapper::LoadLibrary().

7.27.2 Typedef Documentation

typedef std::list<MyoTable> MyoTableList

Definition at line 315 of file offload_myo_host.cpp.

7.27.3 Function Documentation

void __cilkrts_cilk_for_32 (void * , void * , uint32_t , int32_t)

Referenced by __intel_cilk_for_32_offload().

void __cilkrts_cilk_for_64 (void * , void * , uint64_t , int32_t)

Referenced by __intel_cilk_for_64_offload().

void __intel_cilk_for_32_offload (int *size*, void(*)(void *, void *) *copy_constructor*, int *target_number*, void * *raddr*, void * *closure_object*, unsigned int *iters*, unsigned int *grain_size*)

Definition at line 705 of file offload_myo_host.cpp.

void __intel_cilk_for_64_offload (int *size*, void(*)(void *, void *) *copy_constructor*, int *target_number*, void * *raddr*, void * *closure_object*, uint64_t *iters*, uint64_t *grain_size*)

Definition at line 756 of file offload_myo_host.cpp.

static void __offload_myo_fptr_table_register (FptrTableEntry * *entry*) [static]

Definition at line 555 of file offload_myo_host.cpp.
Referenced by __offload_myoRegisterTables().

static void __offload_myo_shared_init_table_register (InitTableEntry * *entry*) [static]

Definition at line 533 of file offload_myo_host.cpp.
Referenced by __offload_myoRegisterTables().

static void __offload_myo_shared_table_register (SharedTableEntry * *entry*) [static]

Definition at line 499 of file offload_myo_host.cpp.
Referenced by __offload_myoRegisterTables().

void __offload_myoFini (void)

Definition at line 474 of file offload_myo_host.cpp.
Referenced by __offload_unregister_image().

static bool __offload_myoInit (void) [static]

Definition at line 382 of file offload_myo_host.cpp.
Referenced by __offload_myoIsAvailable().

static void __offload_myoInit_once (void) [static]

Definition at line 339 of file offload_myo_host.cpp.
Referenced by __offload_myoInit().

void __offload_myoiRemotelThunkCall (void * *thunk,* void * *arg,* int *target_number*)

Definition at line 638 of file offload_myo_host.cpp.

int __offload_myoIsAvailable (int *target_number*)

Definition at line 595 of file offload_myo_host.cpp.
Referenced by __intel_cilk_for_32_offload(), and __intel_cilk_for_64_offload().

static bool __offload_myoLoadLibrary (void) [static]

Definition at line 331 of file offload_myo_host.cpp.
Referenced by __offload_myoInit_once(), __offload_myoRegisterTables(), _Offload_shared_aligned_free(), _Offload_shared_aligned_malloc(), _Offload_shared_free(), and _Offload_shared_malloc().

static void __offload_myoLoadLibrary_once (void) [static]

Definition at line 324 of file offload_myo_host.cpp.
Referenced by __offload_myoLoadLibrary().

void __offload_myoRegisterTables (InitTableEntry * *init_table,* SharedTableEntry * *shared_table,* FptrTableEntry * *fptr_table*)

Definition at line 455 of file offload_myo_host.cpp.
Referenced by offload_init().

void _Offload_shared_aligned_free (void * *ptr*)

Definition at line 693 of file offload_myo_host.cpp.

void* _Offload_shared_aligned_malloc (size_t *size,* size_t *align*)

Definition at line 678 of file offload_myo_host.cpp.

void _Offload_shared_free (void * *ptr*)

Definition at line 666 of file offload_myo_host.cpp.
Referenced by __intel_cilk_for_32_offload(), and __intel_cilk_for_64_offload().

void* _Offload_shared_malloc (size_t *size*)

Definition at line 654 of file offload_myo_host.cpp.
Referenced by __intel_cilk_for_32_offload(), and __intel_cilk_for_64_offload().

static bool fptr_table_entries (FptrTableEntry * *entry*) [static]

Definition at line 436 of file offload_myo_host.cpp.
Referenced by __offload_myoRegisterTables().

static bool shared_table_entries (SharedTableEntry * *entry*) [static]

Definition at line 417 of file offload_myo_host.cpp.
Referenced by __offload_myoRegisterTables().

7.27.4 Variable Documentation

MyoTableList __myo_table_list [static]

Definition at line 316 of file offload_myo_host.cpp.
Referenced by __offload_myo_shared_table_register(), and __offload_myoInit().

mutex_t __myo_table_lock [static]

Definition at line 317 of file offload_myo_host.cpp.

bool __myo_tables = false [static]

Definition at line 318 of file offload_myo_host.cpp.
Referenced by __offload_myo_shared_table_register(), and __offload_myoInit().

bool myo_is_available [static]

Definition at line 303 of file offload_myo_host.cpp.
Referenced by __offload_myoFini(), __offload_myoInit(), and __offload_myoInit_once().

MyoWrapper myo_wrapper [static]

Definition at line 304 of file offload_myo_host.cpp.

7.28 offload_myo_host.h File Reference

```
#include <myotypes.h>
#include <myoimpl.h>
#include <myo.h>
#include "offload.h"
```

Classes

- struct FptrTableEntry
- struct InitTableEntry

Macros

- #define OFFLOAD_MYO_SHARED_TABLE_SECTION_START ".MyoSharedTable."
- #define OFFLOAD_MYO_SHARED_TABLE_SECTION_END ".MyoSharedTable."
- #define OFFLOAD_MYO_SHARED_INIT_TABLE_SECTION_START ".MyoSharedInitTable."
- #define OFFLOAD_MYO_SHARED_INIT_TABLE_SECTION_END ".MyoSharedInitTable."
- #define OFFLOAD_MYO_FPTR_TABLE_SECTION_START ".MyoFptrTable."
- #define OFFLOAD_MYO_FPTR_TABLE_SECTION_END ".MyoFptrTable."

Typedefs

- typedef MyoiSharedVarEntry SharedTableEntry

Functions

- void __offload_myoRegisterTables (InitTableEntry *init_table, SharedTableEntry *shared_table, FptrTableEntry *fptr_table)
- void __offload_myoFini (void)

7.28.1 Macro Definition Documentation

#define OFFLOAD_MYO_FPTR_TABLE_SECTION_END ".MyoFptrTable."

Definition at line 60 of file offload_myo_host.h.

#define OFFLOAD_MYO_FPTR_TABLE_SECTION_START ".MyoFptrTable."

Definition at line 59 of file offload_myo_host.h.

#define OFFLOAD_MYO_SHARED_INIT_TABLE_SECTION_END ".MyoSharedInitTable."

Definition at line 57 of file offload_myo_host.h.

#define OFFLOAD_MYO_SHARED_INIT_TABLE_SECTION_START ".MyoSharedInitTable."

Definition at line 56 of file offload_myo_host.h.

#define OFFLOAD_MYO_SHARED_TABLE_SECTION_END ".MyoSharedTable."

Definition at line 54 of file offload_myo_host.h.

#define OFFLOAD_MYO_SHARED_TABLE_SECTION_START ".MyoSharedTable."

Definition at line 53 of file offload_myo_host.h.

7.28.2 Typedef Documentation

typedef MyoiSharedVarEntry SharedTableEntry

Definition at line 19 of file offload_myo_host.h.

7.28.3 Function Documentation

void __offload_myoFini (void)

Definition at line 474 of file offload_myo_host.cpp.
Referenced by __offload_unregister_image().

void __offload_myoRegisterTables (InitTableEntry * *init_table,* SharedTableEntry * *shared_table,* FptrTableEntry * *fptr_table*)

Definition at line 455 of file offload_myo_host.cpp.
Referenced by offload_init().

7.29 offload_myo_target.cpp File Reference

```
#include "offload_myo_target.h"
#include "offload_target.h"
```

Functions

- void __cilkrts_cilk_for_32 (void *, void *, uint32_t, int32_t)
- void __cilkrts_cilk_for_64 (void *, void *, uint64_t, int32_t)
- static void CheckResult (const char *func, MyoError error)
- static void __offload_myo_shared_table_register (SharedTableEntry *entry)
- static void __offload_myo_fptr_table_register (FptrTableEntry *entry)
- void __offload_myoAcquire (void)
- void __offload_myoRelease (void)
- void __intel_cilk_for_32_offload_wrapper (void *args_)
- void __intel_cilk_for_64_offload_wrapper (void *args_)
- static void __offload_myo_once_init (void)
- void __offload_myoRegisterTables (SharedTableEntry *shared_table, FptrTableEntry *fptr_table)
- void * _Offload_shared_malloc (size_t size)
- void _Offload_shared_free (void *ptr)
- void * _Offload_shared_aligned_malloc (size_t size, size_t align)
- void _Offload_shared_aligned_free (void *ptr)
- void __offload_myoLibInit ()
- void __offload_myoLibFini ()

7.29.1 Function Documentation

void __cilkrts_cilk_for_32 (void * , void * , uint32_t , int32_t)

Referenced by __intel_cilk_for_32_offload_wrapper().

void __cilkrts_cilk_for_64 (void * , void * , uint64_t , int32_t)

Referenced by __intel_cilk_for_64_offload_wrapper().

void __intel_cilk_for_32_offload_wrapper (void * *args_*)

Definition at line 89 of file offload_myo_target.cpp.
Referenced by __offload_myo_once_init().

void __intel_cilk_for_64_offload_wrapper (void * *args_*)

Definition at line 103 of file offload_myo_target.cpp.
Referenced by __offload_myo_once_init().

static void __offload_myo_fptr_table_register (FptrTableEntry * *entry*) [static]

Definition at line 51 of file offload_myo_target.cpp.
Referenced by __offload_myoRegisterTables().

static void __offload_myo_once_init (void) [static]

Definition at line 117 of file offload_myo_target.cpp.
Referenced by __offload_myoRegisterTables().

static void __offload_myo_shared_table_register (SharedTableEntry * *entry*) [static]

Definition at line 27 of file offload_myo_target.cpp.
Referenced by __offload_myoRegisterTables().

void __offload_myoAcquire (void)

Definition at line 77 of file offload_myo_target.cpp.

void __offload_myoLibFini ( )

Definition at line 180 of file offload_myo_target.cpp.

void __offload_myoLibInit ( )

Definition at line 174 of file offload_myo_target.cpp.

void __offload_myoRegisterTables (SharedTableEntry * *shared_table,* FptrTableEntry * *fptr_table*)

Definition at line 129 of file offload_myo_target.cpp.

void __offload_myoRelease (void)

Definition at line 83 of file offload_myo_target.cpp.

void _Offload_shared_aligned_free (void * *ptr*)

Definition at line 167 of file offload_myo_target.cpp.

void* _Offload_shared_aligned_malloc (size_t *size,* size_t *align*)

Definition at line 161 of file offload_myo_target.cpp.

void _Offload_shared_free (void * *ptr*)

Definition at line 155 of file offload_myo_target.cpp.

void* _Offload_shared_malloc (size_t *size*)

Definition at line 149 of file offload_myo_target.cpp.

static void CheckResult (const char * *func,* MyoError *error*) [static]

Definition at line 20 of file offload_myo_target.cpp.
Referenced by __offload_myo_fptr_table_register(), __offload_myo_once_init(), __offload_myo_shared_table_register(), __offload_myoAcquire(), __offload_myoLibInit(), and __offload_myoRelease().

7.30 offload_myo_target.h File Reference

```
#include <myotypes.h>
#include <myoimpl.h>
#include <myo.h>
#include "offload.h"
```

Macros

- #define OFFLOAD_MYO_SHARED_TABLE_SECTION_START ".MyoSharedTable."
- #define OFFLOAD_MYO_SHARED_TABLE_SECTION_END ".MyoSharedTable."
- #define OFFLOAD_MYO_FPTR_TABLE_SECTION_START ".MyoFptrTable."
- #define OFFLOAD_MYO_FPTR_TABLE_SECTION_END ".MyoFptrTable."

Typedefs

- typedef MyoiSharedVarEntry SharedTableEntry
- typedef MyoiTargetSharedFptrEntry FptrTableEntry

Functions

- void __offload_myoRegisterTables (SharedTableEntry *shared_table, FptrTableEntry *fptr_table)
- void __offload_myoAcquire (void)
- void __offload_myoRelease (void)
- void __offload_myoLibInit ()
- void __offload_myoLibFini ()

7.30.1 Macro Definition Documentation

#define OFFLOAD_MYO_FPTR_TABLE_SECTION_END ".MyoFptrTable."

Definition at line 33 of file offload_myo_target.h.

#define OFFLOAD_MYO_FPTR_TABLE_SECTION_START ".MyoFptrTable."

Definition at line 32 of file offload_myo_target.h.

#define OFFLOAD_MYO_SHARED_TABLE_SECTION_END ".MyoSharedTable."

Definition at line 30 of file offload_myo_target.h.

#define OFFLOAD_MYO_SHARED_TABLE_SECTION_START ".MyoSharedTable."

Definition at line 29 of file offload_myo_target.h.

7.30.2 Typedef Documentation

typedef MyoiTargetSharedFptrEntry FptrTableEntry

Definition at line 20 of file offload_myo_target.h.

typedef MyoiSharedVarEntry SharedTableEntry

Definition at line 19 of file offload_myo_target.h.

7.30.3 Function Documentation

void __offload_myoAcquire (void)

Definition at line 77 of file offload_myo_target.cpp.

void __offload_myoLibFini ( )

Definition at line 180 of file offload_myo_target.cpp.

void __offload_myoLibInit ( )

Definition at line 174 of file offload_myo_target.cpp.

void __offload_myoRegisterTables (SharedTableEntry * *shared_table,* FptrTableEntry * *fptr_table*)

Definition at line 129 of file offload_myo_target.cpp.

void __offload_myoRelease (void)

Definition at line 83 of file offload_myo_target.cpp.

7.31 offload_omp_host.cpp File Reference

```
#include <omp.h>
#include "offload.h"
#include "compiler_if_host.h"
```

Functions

- void omp_set_default_device (int num)
- int omp_get_default_device (void)
- int omp_get_num_devices ()
- static void omp_set_int_target (TARGET_TYPE target_type, int target_number, int setting, const char *f_name)
- static int omp_get_int_target (TARGET_TYPE target_type, int target_number, const char *f_name)
- void omp_set_num_threads_target (TARGET_TYPE target_type, int target_number, int num_threads)
- int omp_get_max_threads_target (TARGET_TYPE target_type, int target_number)
- int omp_get_num_procs_target (TARGET_TYPE target_type, int target_number)
- void omp_set_dynamic_target (TARGET_TYPE target_type, int target_number, int num_threads)
- int omp_get_dynamic_target (TARGET_TYPE target_type, int target_number)
- void omp_set_nested_target (TARGET_TYPE target_type, int target_number, int nested)
- int omp_get_nested_target (TARGET_TYPE target_type, int target_number)
- void omp_set_schedule_target (TARGET_TYPE target_type, int target_number, omp_sched_t kind, int modifier)
- void omp_get_schedule_target (TARGET_TYPE target_type, int target_number, omp_sched_t *kind, int *modifier)
- void omp_init_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- void omp_destroy_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- void omp_set_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- void omp_unset_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- int omp_test_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- void omp_init_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void omp_destroy_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void omp_set_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void omp_unset_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)

- int omp_test_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void kmp_set_stacksize_target (TARGET_TYPE target_type, int target_number, int size)
- int kmp_get_stacksize_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_stacksize_s_target (TARGET_TYPE target_type, int target_number, size_t size)
- size_t kmp_get_stacksize_s_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_blocktime_target (TARGET_TYPE target_type, int target_number, int time)
- int kmp_get_blocktime_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_serial_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_turnaround_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_throughput_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_target (TARGET_TYPE target_type, int target_number, int mode)
- int kmp_get_library_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_defaults_target (TARGET_TYPE target_type, int target_number, char const *defaults)
- void kmp_create_affinity_mask_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- void kmp_destroy_affinity_mask_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- int kmp_set_affinity_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- int kmp_get_affinity_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- int kmp_get_affinity_max_proc_target (TARGET_TYPE target_type, int target_number)
- int kmp_set_affinity_mask_proc_target (TARGET_TYPE target_type, int target_number, int proc, kmp_affinity_mask_target_t *mask)
- int kmp_unset_affinity_mask_proc_target (TARGET_TYPE target_type, int target_number, int proc, kmp_affinity_mask_target_t *mask)
- int kmp_get_affinity_mask_proc_target (TARGET_TYPE target_type, int target_number, int proc, kmp_affinity_mask_target_t *mask)

7.31.1 Function Documentation

void kmp_create_affinity_mask_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 609 of file offload_omp_host.cpp.

void kmp_destroy_affinity_mask_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 632 of file offload_omp_host.cpp.

int kmp_get_affinity_mask_proc_target (TARGET_TYPE *target_type,* int *target_number,* int *proc,* kmp_affinity_mask_target_t * *mask*)

Definition at line 812 of file offload_omp_host.cpp.

int kmp_get_affinity_max_proc_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 721 of file offload_omp_host.cpp.

int kmp_get_affinity_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 688 of file offload_omp_host.cpp.

int kmp_get_blocktime_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 517 of file offload_omp_host.cpp.

int kmp_get_library_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 575 of file offload_omp_host.cpp.

size_t kmp_get_stacksize_s_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 498 of file offload_omp_host.cpp.

int kmp_get_stacksize_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 479 of file offload_omp_host.cpp.

int kmp_set_affinity_mask_proc_target (TARGET_TYPE *target_type,* int *target_number,* int *proc,* kmp_affinity_mask_target_t * *mask*)

Definition at line 730 of file offload_omp_host.cpp.

int kmp_set_affinity_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 655 of file offload_omp_host.cpp.

void kmp_set_blocktime_target (TARGET_TYPE *target_type,* int *target_number,* int *time*)

Definition at line 507 of file offload_omp_host.cpp.

void kmp_set_defaults_target (TARGET_TYPE *target_type,* int *target_number,* char const * *defaults*)

Definition at line 584 of file offload_omp_host.cpp.

void kmp_set_library_serial_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 526 of file offload_omp_host.cpp.

void kmp_set_library_target (TARGET_TYPE *target_type,* int *target_number,* int *mode*)

Definition at line 565 of file offload_omp_host.cpp.

void kmp_set_library_throughput_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 552 of file offload_omp_host.cpp.

void kmp_set_library_turnaround_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 539 of file offload_omp_host.cpp.

void kmp_set_stacksize_s_target (TARGET_TYPE *target_type,* int *target_number,* size_t *size*)

Definition at line 488 of file offload_omp_host.cpp.

void kmp_set_stacksize_target (TARGET_TYPE *target_type,* int *target_number,* int *size*)

Definition at line 469 of file offload_omp_host.cpp.

int kmp_unset_affinity_mask_proc_target (TARGET_TYPE *target_type,* int *target_number,* int *proc,* kmp_affinity_mask_target_t * *mask*)

Definition at line 771 of file offload_omp_host.cpp.

void omp_destroy_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 238 of file offload_omp_host.cpp.

void omp_destroy_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 365 of file offload_omp_host.cpp.

int omp_get_default_device (void)

Definition at line 24 of file offload_omp_host.cpp.

int omp_get_dynamic_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 123 of file offload_omp_host.cpp.

static int omp_get_int_target (TARGET_TYPE *target_type,* int *target_number,* const char * *f_name*) [static]

Definition at line 60 of file offload_omp_host.cpp.

Referenced by kmp_get_affinity_max_proc_target(), kmp_get_blocktime_target(), kmp_get_library_target(), kmp_get_stacksize_s_target(), kmp_get_stacksize_target(), omp_get_dynamic_target(), omp_get_max_threads_target(), omp_get_nested_target(), and omp_get_num_procs_target().

int omp_get_max_threads_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 95 of file offload_omp_host.cpp.

int omp_get_nested_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 142 of file offload_omp_host.cpp.

int omp_get_num_devices (void)

Definition at line 29 of file offload_omp_host.cpp.

int omp_get_num_procs_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 104 of file offload_omp_host.cpp.

void omp_get_schedule_target (TARGET_TYPE *target_type,* int *target_number,* omp_sched_t * *kind,* int * *modifier*)

Definition at line 182 of file offload_omp_host.cpp.

void omp_init_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 215 of file offload_omp_host.cpp.

void omp_init_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 342 of file offload_omp_host.cpp.

void omp_set_default_device (int *num*)

Definition at line 17 of file offload_omp_host.cpp.

void omp_set_dynamic_target (TARGET_TYPE *target_type,* int *target_number,* int *num_threads*)

Definition at line 113 of file offload_omp_host.cpp.

static void omp_set_int_target (TARGET_TYPE *target_type,* int *target_number,* int *setting,* const char * *f_name*) [static]

Definition at line 37 of file offload_omp_host.cpp.

Referenced by kmp_set_blocktime_target(), kmp_set_library_target(), kmp_set_stacksize_s_target(), kmp_set_stacksize_target(), omp_set_dynamic_target(), omp_set_nested_target(), and omp_set_num_threads_target().

void omp_set_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 261 of file offload_omp_host.cpp.

void omp_set_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 388 of file offload_omp_host.cpp.

void omp_set_nested_target (TARGET_TYPE *target_type,* int *target_number,* int *nested*)

Definition at line 132 of file offload_omp_host.cpp.

void omp_set_num_threads_target (TARGET_TYPE *target_type,* int *target_number,* int *num_threads*)

Definition at line 85 of file offload_omp_host.cpp.

void omp_set_schedule_target (TARGET_TYPE *target_type,* int *target_number,* omp_sched_t *kind,* int *modifier*)

Definition at line 151 of file offload_omp_host.cpp.

int omp_test_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 307 of file offload_omp_host.cpp.

int omp_test_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 434 of file offload_omp_host.cpp.

void omp_unset_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 284 of file offload_omp_host.cpp.

void omp_unset_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 411 of file offload_omp_host.cpp.

7.32 offload_omp_target.cpp File Reference

```
#include <omp.h>
#include "offload.h"
#include "compiler_if_target.h"
```

Functions

- void omp_set_default_device (int num)
- int omp_get_default_device (void)
- int omp_get_num_devices ()
- static void omp_send_int_to_host (void *ofld_, int setting)
- static int omp_get_int_from_host (void *ofld_)
- void omp_set_num_threads_lrb (void *ofld)
- void omp_get_max_threads_lrb (void *ofld)
- void omp_get_num_procs_lrb (void *ofld)
- void omp_set_dynamic_lrb (void *ofld)
- void omp_get_dynamic_lrb (void *ofld)
- void omp_set_nested_lrb (void *ofld)
- void omp_get_nested_lrb (void *ofld)
- void omp_set_schedule_lrb (void *ofld_)
- void omp_get_schedule_lrb (void *ofld_)
- void omp_init_lock_lrb (void *ofld_)
- void omp_destroy_lock_lrb (void *ofld_)
- void omp_set_lock_lrb (void *ofld_)
- void omp_unset_lock_lrb (void *ofld_)
- void omp_test_lock_lrb (void *ofld_)
- void omp_init_nest_lock_lrb (void *ofld_)
- void omp_destroy_nest_lock_lrb (void *ofld_)
- void omp_set_nest_lock_lrb (void *ofld_)
- void omp_unset_nest_lock_lrb (void *ofld_)
- void omp_test_nest_lock_lrb (void *ofld_)
- void kmp_set_stacksize_lrb (void *ofld)
- void kmp_get_stacksize_lrb (void *ofld)
- void kmp_set_stacksize_s_lrb (void *ofld)
- void kmp_get_stacksize_s_lrb (void *ofld)
- void kmp_set_blocktime_lrb (void *ofld)
- void kmp_get_blocktime_lrb (void *ofld)
- void kmp_set_library_serial_lrb (void *ofld_)
- void kmp_set_library_turnaround_lrb (void *ofld_)
- void kmp_set_library_throughput_lrb (void *ofld_)
- void kmp_set_library_lrb (void *ofld)
- void kmp_get_library_lrb (void *ofld)
- void kmp_set_defaults_lrb (void *ofld_)
- void kmp_create_affinity_mask_lrb (void *ofld_)
- void kmp_destroy_affinity_mask_lrb (void *ofld_)
- void kmp_set_affinity_lrb (void *ofld_)
- void kmp_get_affinity_lrb (void *ofld_)
- void kmp_get_affinity_max_proc_lrb (void *ofld)
- void kmp_set_affinity_mask_proc_lrb (void *ofld_)
- void kmp_unset_affinity_mask_proc_lrb (void *ofld_)
- void kmp_get_affinity_mask_proc_lrb (void *ofld_)
- void omp_set_num_threads_target (TARGET_TYPE target_type, int target_number, int num_threads)
- int omp_get_max_threads_target (TARGET_TYPE target_type, int target_number)
- int omp_get_num_procs_target (TARGET_TYPE target_type, int target_number)
- void omp_set_dynamic_target (TARGET_TYPE target_type, int target_number, int num_threads)
- int omp_get_dynamic_target (TARGET_TYPE target_type, int target_number)
- void omp_set_nested_target (TARGET_TYPE target_type, int target_number, int num_threads)
- int omp_get_nested_target (TARGET_TYPE target_type, int target_number)
- void omp_set_schedule_target (TARGET_TYPE target_type, int target_number, omp_sched_t kind, int modifier)

- void omp_get_schedule_target (TARGET_TYPE target_type, int target_number, omp_sched_t *kind, int *modifier)
- void omp_init_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- void omp_destroy_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- void omp_set_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- void omp_unset_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- int omp_test_lock_target (TARGET_TYPE target_type, int target_number, omp_lock_target_t *lock)
- void omp_init_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void omp_destroy_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void omp_set_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void omp_unset_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- int omp_test_nest_lock_target (TARGET_TYPE target_type, int target_number, omp_nest_lock_target_t *lock)
- void kmp_set_stacksize_target (TARGET_TYPE target_type, int target_number, int size)
- int kmp_get_stacksize_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_stacksize_s_target (TARGET_TYPE target_type, int target_number, size_t size)
- size_t kmp_get_stacksize_s_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_blocktime_target (TARGET_TYPE target_type, int target_number, int time)
- int kmp_get_blocktime_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_serial_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_turnaround_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_throughput_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_library_target (TARGET_TYPE target_type, int target_number, int mode)
- int kmp_get_library_target (TARGET_TYPE target_type, int target_number)
- void kmp_set_defaults_target (TARGET_TYPE target_type, int target_number, char const *defaults)
- void kmp_create_affinity_mask_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- void kmp_destroy_affinity_mask_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- int kmp_set_affinity_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- int kmp_get_affinity_target (TARGET_TYPE target_type, int target_number, kmp_affinity_mask_target_t *mask)
- int kmp_get_affinity_max_proc_target (TARGET_TYPE target_type, int target_number)
- int kmp_set_affinity_mask_proc_target (TARGET_TYPE target_type, int target_number, int proc, kmp_affinity_mask_target_t *mask)
- int kmp_unset_affinity_mask_proc_target (TARGET_TYPE target_type, int target_number, int proc, kmp_affinity_mask_target_t *mask)
- int kmp_get_affinity_mask_proc_target (TARGET_TYPE target_type, int target_number, int proc, kmp_affinity_mask_target_t *mask)

7.32.1 Function Documentation

void kmp_create_affinity_mask_lrb (void * *ofld_*)

Definition at line 518 of file offload_omp_target.cpp.

void kmp_create_affinity_mask_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 951 of file offload_omp_target.cpp.

void kmp_destroy_affinity_mask_lrb (void * *ofld_*)

Definition at line 536 of file offload_omp_target.cpp.

void kmp_destroy_affinity_mask_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 959 of file offload_omp_target.cpp.

void kmp_get_affinity_lrb (void * *ofld_*)

Definition at line 578 of file offload_omp_target.cpp.

void kmp_get_affinity_mask_proc_lrb (void * *ofld_*)

Definition at line 670 of file offload_omp_target.cpp.

int kmp_get_affinity_mask_proc_target (TARGET_TYPE *target_type,* int *target_number,* int *proc,* kmp_affinity_mask_target_t * *mask*)

Definition at line 1013 of file offload_omp_target.cpp.

void kmp_get_affinity_max_proc_lrb (void * *ofld*)

Definition at line 602 of file offload_omp_target.cpp.

int kmp_get_affinity_max_proc_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 985 of file offload_omp_target.cpp.

int kmp_get_affinity_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 976 of file offload_omp_target.cpp.

void kmp_get_blocktime_lrb (void * *ofld*)

Definition at line 435 of file offload_omp_target.cpp.

int kmp_get_blocktime_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 898 of file offload_omp_target.cpp.

void kmp_get_library_lrb (void * *ofld*)

Definition at line 488 of file offload_omp_target.cpp.

int kmp_get_library_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 935 of file offload_omp_target.cpp.

void kmp_get_stacksize_lrb (void * *ofld*)

Definition at line 395 of file offload_omp_target.cpp.

void kmp_get_stacksize_s_lrb (void * *ofld*)

Definition at line 415 of file offload_omp_target.cpp.

size_t kmp_get_stacksize_s_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 882 of file offload_omp_target.cpp.

int kmp_get_stacksize_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 866 of file offload_omp_target.cpp.

void kmp_set_affinity_lrb (void * *ofld_*)

Definition at line 554 of file offload_omp_target.cpp.

void kmp_set_affinity_mask_proc_lrb (void * *ofld_*)

Definition at line 612 of file offload_omp_target.cpp.

int kmp_set_affinity_mask_proc_target (TARGET_TYPE *target_type,* int *target_number,* int *proc,* kmp_affinity_mask_target_t * *mask*)

Definition at line 993 of file offload_omp_target.cpp.

int kmp_set_affinity_target (TARGET_TYPE *target_type,* int *target_number,* kmp_affinity_mask_target_t * *mask*)

Definition at line 967 of file offload_omp_target.cpp.

void kmp_set_blocktime_lrb (void * *ofld*)

Definition at line 425 of file offload_omp_target.cpp.

void kmp_set_blocktime_target (TARGET_TYPE *target_type,* int *target_number,* int *time*)

Definition at line 890 of file offload_omp_target.cpp.

void kmp_set_defaults_lrb (void * *ofld_*)

Definition at line 498 of file offload_omp_target.cpp.

void kmp_set_defaults_target (TARGET_TYPE *target_type,* int *target_number,* char const * *defaults*)

Definition at line 943 of file offload_omp_target.cpp.

void kmp_set_library_lrb (void * *ofld*)

Definition at line 478 of file offload_omp_target.cpp.

void kmp_set_library_serial_lrb (void * *ofld_*)

Definition at line 445 of file offload_omp_target.cpp.

void kmp_set_library_serial_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 906 of file offload_omp_target.cpp.

void kmp_set_library_target (TARGET_TYPE *target_type,* int *target_number,* int *mode*)

Definition at line 927 of file offload_omp_target.cpp.

void kmp_set_library_throughput_lrb (void * *ofld_*)

Definition at line 467 of file offload_omp_target.cpp.

void kmp_set_library_throughput_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 920 of file offload_omp_target.cpp.

void kmp_set_library_turnaround_lrb (void * *ofld_*)

Definition at line 456 of file offload_omp_target.cpp.

void kmp_set_library_turnaround_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 913 of file offload_omp_target.cpp.

void kmp_set_stacksize_lrb (void * *ofld*)

Definition at line 385 of file offload_omp_target.cpp.

void kmp_set_stacksize_s_lrb (void * *ofld*)

Definition at line 405 of file offload_omp_target.cpp.

void kmp_set_stacksize_s_target (TARGET_TYPE *target_type,* int *target_number,* size_t *size*)

Definition at line 874 of file offload_omp_target.cpp.

void kmp_set_stacksize_target (TARGET_TYPE *target_type,* int *target_number,* int *size*)

Definition at line 858 of file offload_omp_target.cpp.

void kmp_unset_affinity_mask_proc_lrb (void * *ofld_*)

Definition at line 641 of file offload_omp_target.cpp.

int kmp_unset_affinity_mask_proc_target (TARGET_TYPE *target_type,* int *target_number,* int *proc,* kmp_affinity_mask_target_t * *mask*)

Definition at line 1003 of file offload_omp_target.cpp.

void omp_destroy_lock_lrb (void * *ofld_*)

Definition at line 207 of file offload_omp_target.cpp.

void omp_destroy_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 784 of file offload_omp_target.cpp.

void omp_destroy_nest_lock_lrb (void * *ofld_*)

Definition at line 305 of file offload_omp_target.cpp.

void omp_destroy_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 825 of file offload_omp_target.cpp.

int omp_get_default_device (void)

Definition at line 21 of file offload_omp_target.cpp.

void omp_get_dynamic_lrb (void * *ofld*)

Definition at line 109 of file offload_omp_target.cpp.

int omp_get_dynamic_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 734 of file offload_omp_target.cpp.

static int omp_get_int_from_host (void * *ofld_*) [static]

Definition at line 50 of file offload_omp_target.cpp.

Referenced by kmp_set_blocktime_lrb(), kmp_set_library_lrb(), kmp_set_stacksize_lrb(), kmp_set_stacksize_s_lrb(), omp_set_dynamic_lrb(), omp_set_nested_lrb(), and omp_set_num_threads_lrb().

void omp_get_max_threads_lrb (void * *ofld*)

Definition at line 79 of file offload_omp_target.cpp.

int omp_get_max_threads_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 710 of file offload_omp_target.cpp.

void omp_get_nested_lrb (void * *ofld*)

Definition at line 129 of file offload_omp_target.cpp.

int omp_get_nested_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 750 of file offload_omp_target.cpp.

int omp_get_num_devices (void)

Definition at line 26 of file offload_omp_target.cpp.

void omp_get_num_procs_lrb (void * *ofld*)

Definition at line 89 of file offload_omp_target.cpp.

int omp_get_num_procs_target (TARGET_TYPE *target_type,* int *target_number*)

Definition at line 718 of file offload_omp_target.cpp.

void omp_get_schedule_lrb (void * *ofld_*)

Definition at line 163 of file offload_omp_target.cpp.

void omp_get_schedule_target (TARGET_TYPE *target_type,* int *target_number,* omp_sched_t * *kind,* int * *modifier*)

Definition at line 767 of file offload_omp_target.cpp.

void omp_init_lock_lrb (void * *ofld_*)

Definition at line 189 of file offload_omp_target.cpp.

void omp_init_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 776 of file offload_omp_target.cpp.

void omp_init_nest_lock_lrb (void * *ofld_*)

Definition at line 287 of file offload_omp_target.cpp.

void omp_init_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 817 of file offload_omp_target.cpp.

static void omp_send_int_to_host (void * *ofld_,* int *setting*) [static]

Definition at line 33 of file offload_omp_target.cpp.

Referenced by kmp_get_affinity_max_proc_lrb(), kmp_get_blocktime_lrb(), kmp_get_library_lrb(), kmp_get_stacksize_lrb(), kmp_get_stacksize_s_lrb(), omp_get_dynamic_lrb(), omp_get_max_threads_lrb(), omp_get_nested_lrb(), and omp_get_num_procs_lrb().

void omp_set_default_device (int *num*)

Definition at line 17 of file offload_omp_target.cpp.

void omp_set_dynamic_lrb (void * *ofld*)

Definition at line 99 of file offload_omp_target.cpp.

void omp_set_dynamic_target (TARGET_TYPE *target_type,* int *target_number,* int *num_threads*)

Definition at line 726 of file offload_omp_target.cpp.

void omp_set_lock_lrb (void * *ofld_*)

Definition at line 225 of file offload_omp_target.cpp.

void omp_set_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 792 of file offload_omp_target.cpp.

void omp_set_nest_lock_lrb (void * *ofld_*)

Definition at line 323 of file offload_omp_target.cpp.

void omp_set_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 833 of file offload_omp_target.cpp.

void omp_set_nested_lrb (void * *ofld*)

Definition at line 119 of file offload_omp_target.cpp.

void omp_set_nested_target (TARGET_TYPE *target_type,* int *target_number,* int *num_threads*)

Definition at line 742 of file offload_omp_target.cpp.

void omp_set_num_threads_lrb (void * *ofld*)

Definition at line 69 of file offload_omp_target.cpp.

void omp_set_num_threads_target (TARGET_TYPE *target_type,* int *target_number,* int *num_threads*)

Definition at line 702 of file offload_omp_target.cpp.

void omp_set_schedule_lrb (void * *ofld_*)

Definition at line 139 of file offload_omp_target.cpp.

void omp_set_schedule_target (TARGET_TYPE *target_type,* int *target_number,* omp_sched_t *kind,* int *modifier*)

Definition at line 758 of file offload_omp_target.cpp.

void omp_test_lock_lrb (void * *ofld_*)

Definition at line 261 of file offload_omp_target.cpp.

int omp_test_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 808 of file offload_omp_target.cpp.

void omp_test_nest_lock_lrb (void * *ofld_*)

Definition at line 359 of file offload_omp_target.cpp.

int omp_test_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 849 of file offload_omp_target.cpp.

void omp_unset_lock_lrb (void * *ofld_*)

Definition at line 243 of file offload_omp_target.cpp.

void omp_unset_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_lock_target_t * *lock*)

Definition at line 800 of file offload_omp_target.cpp.

void omp_unset_nest_lock_lrb (void * *ofld_*)

Definition at line 341 of file offload_omp_target.cpp.

void omp_unset_nest_lock_target (TARGET_TYPE *target_type,* int *target_number,* omp_nest_lock_target_t * *lock*)

Definition at line 841 of file offload_omp_target.cpp.

7.33 offload_orsl.cpp File Reference

```
#include "offload_orsl.h"
#include <stdlib.h>
#include "offload_host.h"
#include "orsl-lite/include/orsl-lite.h"
```

Namespaces

- ORSL

Functions

- void ORSL::init ()
- bool ORSL::reserve (int device)
- bool ORSL::try_reserve (int device)
- void ORSL::release (int device)

Variables

- static bool ORSL::is_enabled = false
- static const ORSLTag ORSL::my_tag = ”Offload”

7.34 offload_orsl.h File Reference

Namespaces

- ORSL

Functions

- void ORSL::init ()
- bool ORSL::reserve (int device)
- bool ORSL::try_reserve (int device)
- void ORSL::release (int device)

7.35 offload_table.cpp File Reference

```
#include "offload_table.h"
#include "offload_common.h"
```

Functions

- void omp_set_num_threads_lrb (void *)
- void omp_get_max_threads_lrb (void *)
- void omp_get_num_procs_lrb (void *)
- void omp_set_dynamic_lrb (void *)
- void omp_get_dynamic_lrb (void *)
- void omp_set_nested_lrb (void *)
- void omp_get_nested_lrb (void *)
- void omp_set_schedule_lrb (void *)
- void omp_get_schedule_lrb (void *)
- void omp_init_lock_lrb (void *)
- void omp_destroy_lock_lrb (void *)
- void omp_set_lock_lrb (void *)
- void omp_unset_lock_lrb (void *)
- void omp_test_lock_lrb (void *)
- void omp_init_nest_lock_lrb (void *)
- void omp_destroy_nest_lock_lrb (void *)
- void omp_set_nest_lock_lrb (void *)
- void omp_unset_nest_lock_lrb (void *)
- void omp_test_nest_lock_lrb (void *)
- void kmp_set_stacksize_lrb (void *)
- void kmp_get_stacksize_lrb (void *)
- void kmp_set_stacksize_s_lrb (void *)
- void kmp_get_stacksize_s_lrb (void *)
- void kmp_set_blocktime_lrb (void *)
- void kmp_get_blocktime_lrb (void *)
- void kmp_set_library_serial_lrb (void *)
- void kmp_set_library_turnaround_lrb (void *)
- void kmp_set_library_throughput_lrb (void *)
- void kmp_set_library_lrb (void *)
- void kmp_get_library_lrb (void *)
- void kmp_set_defaults_lrb (void *)
- void kmp_create_affinity_mask_lrb (void *)
- void kmp_destroy_affinity_mask_lrb (void *)
- void kmp_set_affinity_lrb (void *)

- void kmp_get_affinity_lrb (void *)
- void kmp_get_affinity_max_proc_lrb (void *)
- void kmp_set_affinity_mask_proc_lrb (void *)
- void kmp_unset_affinity_mask_proc_lrb (void *)
- void kmp_get_affinity_mask_proc_lrb (void *)
- void __offload_register_tables (FuncList::Node *entry_table, FuncList::Node *func_table, VarList::Node *var_table)
- void __offload_unregister_tables (FuncList::Node *entry_table, FuncList::Node *func_table, VarList::Node *var_table)

Variables

- static FuncTable::Entry predefined_entries []
- static FuncList::Node predefined_table
- FuncList __offload_funcs
- VarList __offload_vars

7.35.1 Function Documentation

void __offload_register_tables (FuncList::Node * *entry_table,* FuncList::Node * *func_table,* VarList::Node * *var_table*)

Definition at line 344 of file offload_table.cpp.
Referenced by offload_init().

void __offload_unregister_tables (FuncList::Node * *entry_table,* FuncList::Node * *func_table,* VarList::Node * *var_table*)

Definition at line 362 of file offload_table.cpp.
Referenced by offload_fini().

void kmp_create_affinity_mask_lrb (void *)

Definition at line 518 of file offload_omp_target.cpp.

void kmp_destroy_affinity_mask_lrb (void *)

Definition at line 536 of file offload_omp_target.cpp.

void kmp_get_affinity_lrb (void *)

Definition at line 578 of file offload_omp_target.cpp.

void kmp_get_affinity_mask_proc_lrb (void *)

Definition at line 670 of file offload_omp_target.cpp.

void kmp_get_affinity_max_proc_lrb (void *)

Definition at line 602 of file offload_omp_target.cpp.

void kmp_get_blocktime_lrb (void *)

Definition at line 435 of file offload_omp_target.cpp.

void kmp_get_library_lrb (void *)

Definition at line 488 of file offload_omp_target.cpp.

void kmp_get_stacksize_lrb (void *)

Definition at line 395 of file offload_omp_target.cpp.

void kmp_get_stacksize_s_lrb (void *)

Definition at line 415 of file offload_omp_target.cpp.

void kmp_set_affinity_lrb (void *)

Definition at line 554 of file offload_omp_target.cpp.

void kmp_set_affinity_mask_proc_lrb (void *)

Definition at line 612 of file offload_omp_target.cpp.

void kmp_set_blocktime_lrb (void *)

Definition at line 425 of file offload_omp_target.cpp.

void kmp_set_defaults_lrb (void *)

Definition at line 498 of file offload_omp_target.cpp.

void kmp_set_library_lrb (void *)

Definition at line 478 of file offload_omp_target.cpp.

void kmp_set_library_serial_lrb (void *)

Definition at line 445 of file offload_omp_target.cpp.

void kmp_set_library_throughput_lrb (void *)

Definition at line 467 of file offload_omp_target.cpp.

void kmp_set_library_turnaround_lrb (void *)

Definition at line 456 of file offload_omp_target.cpp.

void kmp_set_stacksize_lrb (void *)

Definition at line 385 of file offload_omp_target.cpp.

void kmp_set_stacksize_s_lrb (void *)

Definition at line 405 of file offload_omp_target.cpp.

void kmp_unset_affinity_mask_proc_lrb (void *)

Definition at line 641 of file offload_omp_target.cpp.

void omp_destroy_lock_lrb (void *)

Definition at line 207 of file offload_omp_target.cpp.

void omp_destroy_nest_lock_lrb (void *)

Definition at line 305 of file offload_omp_target.cpp.

void omp_get_dynamic_lrb (void *)

Definition at line 109 of file offload_omp_target.cpp.

void omp_get_max_threads_lrb (void *)

Definition at line 79 of file offload_omp_target.cpp.

void omp_get_nested_lrb (void *)

Definition at line 129 of file offload_omp_target.cpp.

void omp_get_num_procs_lrb (void *)

Definition at line 89 of file offload_omp_target.cpp.

void omp_get_schedule_lrb (void *)

Definition at line 163 of file offload_omp_target.cpp.

void omp_init_lock_lrb (void *)

Definition at line 189 of file offload_omp_target.cpp.

void omp_init_nest_lock_lrb (void *)

Definition at line 287 of file offload_omp_target.cpp.

void omp_set_dynamic_lrb (void *)

Definition at line 99 of file offload_omp_target.cpp.

void omp_set_lock_lrb (void *)

Definition at line 225 of file offload_omp_target.cpp.

void omp_set_nest_lock_lrb (void *)

Definition at line 323 of file offload_omp_target.cpp.

void omp_set_nested_lrb (void *)

Definition at line 119 of file offload_omp_target.cpp.

void omp_set_num_threads_lrb (void *)

Definition at line 69 of file offload_omp_target.cpp.

void omp_set_schedule_lrb (void *)

Definition at line 139 of file offload_omp_target.cpp.

void omp_test_lock_lrb (void *)

Definition at line 261 of file offload_omp_target.cpp.

void omp_test_nest_lock_lrb (void *)

Definition at line 359 of file offload_omp_target.cpp.

void omp_unset_lock_lrb (void *)

Definition at line 243 of file offload_omp_target.cpp.

void omp_unset_nest_lock_lrb (void *)

Definition at line 341 of file offload_omp_target.cpp.

7.35.2 Variable Documentation

FuncList __offload_funcs

Definition at line 161 of file offload_table.cpp.
Referenced by Marshaller::receive_func_ptr(), Marshaller::send_func_ptr(), and OffloadDescriptor::setup_descriptors().

VarList __offload_vars

Definition at line 164 of file offload_table.cpp.
Referenced by Engine::init_ptr_data(), server_var_table_copy(), and server_var_table_size().

FuncTable::Entry predefined_entries[] [static]

Definition at line 61 of file offload_table.cpp.

FuncList __offload_entries & predefined_table [static]

Initial value:

```
= {
    { predefined_entries, -1 },
    0, 0
}
```

Definition at line 149 of file offload_table.cpp.

7.36 offload_table.h File Reference

Function and Variable tables used by the runtime library.

```
   #include <iterator>
#include "offload_util.h"
```

Classes

- class TableList< T >
- struct TableList< T >::Node
- struct FuncTable
- struct FuncTable::Entry

  Function table entry.
- class FuncList
- struct VarTable
- struct VarTable::Entry

  Variable table entry.
- class VarList
- class VarList::Iterator
- struct VarList::BufEntry

Macros

- #define OFFLOAD_ENTRY_TABLE_SECTION_START ".OffloadEntryTable."
- #define OFFLOAD_ENTRY_TABLE_SECTION_END ".OffloadEntryTable."
- #define OFFLOAD_FUNC_TABLE_SECTION_START ".OffloadFuncTable."
- #define OFFLOAD_FUNC_TABLE_SECTION_END ".OffloadFuncTable."
- #define OFFLOAD_VAR_TABLE_SECTION_START ".OffloadVarTable."
- #define OFFLOAD_VAR_TABLE_SECTION_END ".OffloadVarTable."

Functions

- void __offload_register_tables (FuncList::Node *entry_table, FuncList::Node *func_table, VarList::Node *var_table)
- void __offload_unregister_tables (FuncList::Node *entry_table, FuncList::Node *func_table, VarList::Node *var_table)

Variables

- FuncList __offload_entries
- FuncList __offload_funcs
- VarList __offload_vars

7.36.1 Detailed Description

Function and Variable tables used by the runtime library.
Definition in file offload_table.h.

7.36.2 Macro Definition Documentation

#define OFFLOAD_ENTRY_TABLE_SECTION_END ".OffloadEntryTable."

Definition at line 270 of file offload_table.h.

#define OFFLOAD_ENTRY_TABLE_SECTION_START ".OffloadEntryTable."

Definition at line 269 of file offload_table.h.

#define OFFLOAD_FUNC_TABLE_SECTION_END ".OffloadFuncTable."

Definition at line 273 of file offload_table.h.

#define OFFLOAD_FUNC_TABLE_SECTION_START ".OffloadFuncTable."

Definition at line 272 of file offload_table.h.

#define OFFLOAD_VAR_TABLE_SECTION_END ".OffloadVarTable."

Definition at line 276 of file offload_table.h.

#define OFFLOAD_VAR_TABLE_SECTION_START ".OffloadVarTable."

Definition at line 275 of file offload_table.h.

7.36.3 Function Documentation

void __offload_register_tables (FuncList::Node * *entry_table,* FuncList::Node * *func_table,* VarList::Node * *var_table*)

Definition at line 344 of file offload_table.cpp.
Referenced by offload_init().

void __offload_unregister_tables (FuncList::Node * *entry_table,* FuncList::Node * *func_table,* VarList::Node * *var_table*)

Definition at line 362 of file offload_table.cpp.
Referenced by offload_fini().

7.36.4 Variable Documentation

FuncList __offload_entries

Referenced by OffloadDescriptor::offload().

FuncList __offload_funcs

Definition at line 161 of file offload_table.cpp.
Referenced by Marshaller::receive_func_ptr(), Marshaller::send_func_ptr(), and OffloadDescriptor::setup_descriptors().

VarList __offload_vars

Definition at line 164 of file offload_table.cpp.
Referenced by Engine::init_ptr_data(), server_var_table_copy(), and server_var_table_size().

7.37 offload_target.cpp File Reference

```
#include "offload_target.h"
#include <stdlib.h>
#include <unistd.h>
#include <omp.h>
#include <map>
```

Typedefs

- typedef void(* offload_func_with_parms)(void *)

Functions

- void __offload_target_init (void)
- int _Offload_number_of_devices (void)
- int _Offload_get_device_number (void)
- int _Offload_get_physical_device_number (void)

Variables

- const char * prefix
- int console_enabled = 0
- int offload_report_level = 0
- static const char * vardesc_direction_as_string []
- static const char * vardesc_type_as_string []
- int mic_index = -1
- int mic_engines_total = -1
- uint64_t mic_frequency = 0
- int offload_number = 0
- static std::map< void *, RefInfo * > ref_data
- static mutex_t add_ref_lock

7.37.1 Typedef Documentation

typedef void(∗ offload_func_with_parms)(void ∗)

Definition at line 24 of file offload_target.cpp.

7.37.2 Function Documentation

void __offload_target_init (void)

Definition at line 718 of file offload_target.cpp.
Referenced by OFFLOAD_TARGET_MAIN().

int _Offload_get_device_number (void)

Definition at line 744 of file offload_target.cpp.

int _Offload_get_physical_device_number (void)

Definition at line 749 of file offload_target.cpp.

int _Offload_number_of_devices (void)

Definition at line 739 of file offload_target.cpp.

7.37.3 Variable Documentation

mutex_t add_ref_lock `[static]`

Definition at line 61 of file offload_target.cpp.

int console_enabled = 0

Definition at line 28 of file offload_target.cpp.
Referenced by Engine::init_device(), OffloadDescriptor::offload(), Marshaller::receive_func_ptr(), Marshaller::send_func_ptr(), and server_init().

int mic_engines_total = -1

Definition at line 57 of file offload_target.cpp.
Referenced by _Offload_number_of_devices().

uint64_t mic_frequency = 0

Definition at line 58 of file offload_target.cpp.
Referenced by __offload_target_init().

int mic_index = -1

Definition at line 56 of file offload_target.cpp.
Referenced by _Offload_get_device_number(), offload_stage(), omp_get_default_device(), and server_init().

int offload_number = 0

Definition at line 59 of file offload_target.cpp.
Referenced by OffloadDescriptor::offload().

int offload_report_level = 0

Definition at line 29 of file offload_target.cpp.
Referenced by __offload_init_library_once(), Engine::init_device(), OffloadDescriptor::offload(), server_init(), and OffloadDescriptor::setup_misc_data().

const char* prefix

Definition at line 27 of file offload_target.cpp.
Referenced by __offload_target_init(), offload_signal(), and offload_stage().

std::map<void*, RefInfo*> ref_data [static]

Definition at line 60 of file offload_target.cpp.
Referenced by OffloadDescriptor::scatter_copyin_data().

const char* vardesc_direction_as_string[] [static]

Initial value:

```
= {
    "NOCOPY",
    "IN",
    "OUT",
    "INOUT"
}
```

Definition at line 32 of file offload_target.cpp.
Referenced by OffloadDescriptor::merge_var_descs().

const char* vardesc_type_as_string[] [static]

Initial value:

```
= {
    "unknown",
    "data",
    "data_ptr",
    "func_ptr",
    "void_ptr",
    "string_ptr",
    "dv",
    "dv_data",
    "dv_data_slice",
    "dv_ptr",
    "dv_ptr_data",
    "dv_ptr_data_slice",
    "cean_var",
    "cean_var_ptr",
    "c_data_ptr_array"
}
```

Definition at line 38 of file offload_target.cpp.
Referenced by OffloadDescriptor::merge_var_descs().

7.38 offload_target.h File Reference

```
#include "offload_common.h"
#include "coi/coi_server.h"
```

Classes

- class OffloadDescriptor
- struct RefInfo

Functions

- void __offload_target_init (void)

Variables

- int mic_index
- int mic_engines_total
- uint64_t mic_frequency

7.38.1 Function Documentation

void __offload_target_init (void)

Definition at line 718 of file offload_target.cpp.
Referenced by OFFLOAD_TARGET_MAIN().

7.38.2 Variable Documentation

int mic_engines_total

Definition at line 118 of file offload_host.cpp.
Referenced by __offload_fini_library(), __offload_init_library(), __offload_init_library_once(), __offload_myoFini(), __offload_myoInit_once(), __offload_myoIsAvailable(), __offload_register_image(), _Offload_number_of_devices(), _Offload_signaled(), Engine::init_device(), OFFLOAD_TARGET_ACQUIRE(), OFFLOAD_TARGET_ACQUIRE1(), omp_get_num_devices(), server_init(), and Thread::~Thread().

uint64_t mic_frequency

Definition at line 58 of file offload_target.cpp.
Referenced by __offload_target_init().

int mic_index

Definition at line 56 of file offload_target.cpp.

7.39 offload_target_main.cpp File Reference

Functions

- void __offload_target_main (void)
- int main (int argc, char **argv)

7.39.1 Function Documentation

void __offload_target_main (void)

Referenced by main().

int main (int *argc,* char ** *argv*)

Definition at line 13 of file offload_target_main.cpp.

7.40 offload_timer.h File Reference

```
#include <stdio.h>
#include <stdarg.h>
#include <stdint.h>
#include "liboffload_error_codes.h"
```

Macros

- #define OFFLOAD_TIMER_START(...)
- #define OFFLOAD_TIMER_STOP(...)
- #define OFFLOAD_TIMER_INIT(...)
- #define OFFLOAD_TIMER_TARGET_DATA(...)
- #define OFFLOAD_TIMER_DATALEN(...) (0)

Variables

- int timer_enabled

7.40.1 Macro Definition Documentation

#define OFFLOAD_TIMER_DATALEN(...) (0)

Definition at line 168 of file offload_timer.h.

Referenced by OffloadDescriptor::offload(), OffloadDescriptor::scatter_copyout_data(), and OffloadDescriptor::setup_misc_data().

#define OFFLOAD_TIMER_INIT(...)

Definition at line 166 of file offload_timer.h.

Referenced by OffloadDescriptor::offload(), OFFLOAD_TARGET_ACQUIRE(), and OFFLOAD_TARGET_ACQUIRE1().

#define OFFLOAD_TIMER_START(...)

Definition at line 164 of file offload_timer.h.

Referenced by OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::offload(), OFFLOAD_TARGET_ACQUIRE(), OFFLOAD_TARGET_ACQUIRE1(), and OffloadDescriptor::scatter_copyin_data().

#define OFFLOAD_TIMER_STOP(...)

Definition at line 165 of file offload_timer.h.

Referenced by OffloadDescriptor::cleanup(), OffloadDescriptor::gather_copyout_data(), OffloadDescriptor::offload(), OFFLOAD_TARGET_ACQUIRE(), OFFLOAD_TARGET_ACQUIRE1(), and OffloadDescriptor::scatter_copyin_data().

#define OFFLOAD_TIMER_TARGET_DATA(...)

Definition at line 167 of file offload_timer.h.

Referenced by OffloadDescriptor::offload(), and OffloadDescriptor::scatter_copyout_data().

7.40.2 Variable Documentation

int timer_enabled

Definition at line 24 of file offload_timer_host.cpp.

Referenced by __offload_init_library_once(), __offload_unregister_image(), OffloadDescriptor::offload(), and OffloadDescriptor::setup_misc_data().

7.41 offload_timer_host.cpp File Reference

```
#include "offload_timer.h"
#include <x86intrin.h>
#include "offload_host.h"
#include <sstream>
#include <iostream>
#include <iomanip>
```

Variables

- int timer_enabled = 0

7.41.1 Variable Documentation

int timer_enabled = 0

Definition at line 24 of file offload_timer_host.cpp.

7.42 offload_timer_target.cpp File Reference

```
#include "offload_timer.h"
#include "offload_target.h"
#include <x86intrin.h>
```

Variables

- int timer_enabled = 0

7.42.1 Variable Documentation

int timer_enabled = 0

Definition at line 22 of file offload_timer_target.cpp.

Referenced by __offload_init_library_once(), __offload_unregister_image(), OffloadDescriptor::offload(), and OffloadDescriptor::setup_misc_data().

7.43 offload_trace.cpp File Reference

```
#include "offload_trace.h"
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <sstream>
#include "liboffload_error_codes.h"
```

Functions

- static const char * offload_stage (std::stringstream &ss, int offload_number, const char *tag, const char *text, bool print_tag)
- static const char * offload_signal (std::stringstream &ss, int offload_number, const char *tag, const char *text)
- void offload_stage_print (int stage, int offload_number,...)

Variables

- const char * prefix
- int mic_index

7.43.1 Function Documentation

static const char* offload_signal (std::stringstream & *ss,* int *offload_number,* const char * *tag,* const char * *text*) [static]

Definition at line 56 of file offload_trace.cpp.
Referenced by offload_stage_print().

static const char* offload_stage (std::stringstream & *ss,* int *offload_number,* const char * *tag,* const char * *text,* bool *print_tag*) [static]

Definition at line 26 of file offload_trace.cpp.
Referenced by offload_stage_print().

void offload_stage_print (int *stage,* int *offload_number,* ...)

Definition at line 70 of file offload_trace.cpp.

7.43.2 Variable Documentation

int mic_index

Definition at line 56 of file offload_target.cpp.
Referenced by _Offload_get_device_number(), and offload_stage().

const char* prefix

Definition at line 81 of file offload_host.cpp.
Referenced by __offload_init_library_once(), __offload_target_init(), offload_signal(), and offload_stage().

7.44 offload_trace.h File Reference

Enumerations

- enum OffloadTraceStage {
  c_offload_start = 0, c_offload_init, c_offload_register, c_offload_init_func,
  c_offload_create_buf_host, c_offload_create_buf_mic, c_offload_send_pointer_data, c_offload_sent_pointer_data,
  c_offload_gather_copyin_data, c_offload_copyin_data, c_offload_compute, c_offload_receive_pointer_data,
  c_offload_received_pointer_data, c_offload_start_target_func, c_offload_var, c_offload_scatter_copyin_data,
  c_offload_gather_copyout_data, c_offload_scatter_copyout_data, c_offload_copyout_data, c_offload_signal,
  c_offload_wait, c_offload_unregister, c_offload_destroy, c_offload_finish,
  c_offload_myoinit, c_offload_myoregister, c_offload_mic_myo_shared, c_offload_mic_myo_fptr,
  c_offload_myosharedmalloc, c_offload_myosharedfree, c_offload_myosharedalignedmalloc, c_offload_myosharedalignedfree,
  c_offload_myoacquire, c_offload_myorelease, c_offload_myofini }

Functions

- void offload_stage_print (int stage, int offload_number,...)

7.44.1 Enumeration Type Documentation

enum OffloadTraceStage

Enumerator

c_offload_start

c_offload_init

c_offload_register

c_offload_init_func

c_offload_create_buf_host

c_offload_create_buf_mic

c_offload_send_pointer_data

c_offload_sent_pointer_data

c_offload_gather_copyin_data

c_offload_copyin_data

c_offload_compute

c_offload_receive_pointer_data

c_offload_received_pointer_data

c_offload_start_target_func

c_offload_var

c_offload_scatter_copyin_data

c_offload_gather_copyout_data

c_offload_scatter_copyout_data

c_offload_copyout_data

c_offload_signal

c_offload_wait

c_offload_unregister

c_offload_destroy

c_offload_finish

c_offload_myoinit

c_offload_myoregister

c_offload_mic_myo_shared

c_offload_mic_myo_fptr

c_offload_myosharedmalloc

c_offload_myosharedfree

c_offload_myosharedalignedmalloc

c_offload_myosharedalignedfree

c_offload_myoacquire

c_offload_myorelease

c_offload_myofini

Definition at line 15 of file offload_trace.h.

7.44.2 Function Documentation

void offload_stage_print (int *stage,* int *offload_number,* ...)

Definition at line 70 of file offload_trace.cpp.

7.45 offload_util.cpp File Reference

```
#include "offload_util.h"
#include <errno.h>
#include "liboffload_error_codes.h"
```

Functions

- bool __offload_parse_size_string (const char *str, uint64_t &new_size)
- bool __offload_parse_int_string (const char *str, int64_t &value)
- void * DL_sym (void *handle, const char *name, const char *version)
- int64_t get_el_value (char *base, int64_t offset, int64_t size)

7.45.1 Function Documentation

bool __offload_parse_int_string (const char * *str,* int64_t & *value*)

Definition at line 90 of file offload_util.cpp.
Referenced by __offload_init_library_once(), and ORSL::init().

bool __offload_parse_size_string (const char * *str,* uint64_t & *new_size*)

Definition at line 32 of file offload_util.cpp.
Referenced by __offload_init_library_once().

void* DL_sym (void * *handle,* const char * *name,* const char * *version*)

Definition at line 171 of file offload_util.cpp.
Referenced by COI::init(), and MyoWrapper::LoadLibrary().

int64_t get_el_value (char * *base,* int64_t *offset,* int64_t *size*)

Definition at line 185 of file offload_util.cpp.
Referenced by OffloadDescriptor::ReadArrElements< T >::read_next().

7.46 offload_util.h File Reference

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <dlfcn.h>
#include <pthread.h>
```

Classes

- struct mutex_t
- struct mutex_locker_t

Macros

- #define thread_key_create(key, destructor) pthread_key_create((key), (destructor))
- #define thread_key_delete(key) pthread_key_delete(key)
- #define thread_getspecific(key) pthread_getspecific(key)
- #define thread_setspecific(key, value) pthread_setspecific(key, value)
- #define DL_open(path) dlopen((path), RTLD_NOW)
- #define DL_close(handle) dlclose(handle)

- #define DL_addr(addr, info) dladdr((addr), (info))
- #define OFFLOAD_ONCE_CONTROL_INIT PTHREAD_ONCE_INIT
- #define __offload_run_once(ctrl, func) pthread_once(ctrl, func)

Typedefs

- typedef pthread_once_t OffloadOnceControl

Functions

- void * DL_sym (void *handle, const char *name, const char *version)
- bool __offload_parse_size_string (const char *str, uint64_t &new_size)
- bool __offload_parse_int_string (const char *str, int64_t &value)
- int64_t get_el_value (char *base, int64_t offset, int64_t size)

7.46.1 Macro Definition Documentation

#define __offload_run_once(*ctrl, func*) pthread_once(ctrl, func)

Definition at line 138 of file offload_util.h.
Referenced by __offload_init_library(), __offload_myoInit(), and __offload_myoLoadLibrary().

#define DL_addr(*addr, info*) dladdr((addr), (info))

Definition at line 123 of file offload_util.h.

#define DL_close(*handle*) dlclose(handle)

Definition at line 122 of file offload_util.h.
Referenced by COI::fini().

#define DL_open(*path*) dlopen((path), RTLD_NOW)

Definition at line 121 of file offload_util.h.
Referenced by COI::init(), and MyoWrapper::LoadLibrary().

#define OFFLOAD_ONCE_CONTROL_INIT PTHREAD_ONCE_INIT

Definition at line 136 of file offload_util.h.
Referenced by __offload_init_library(), __offload_myoInit(), and __offload_myoLoadLibrary().

#define thread_getspecific(*key*) pthread_getspecific(key)

Definition at line 48 of file offload_util.h.
Referenced by Engine::get_auto_vars(), and Engine::get_pipeline().

#define thread_key_create(*key, destructor*) pthread_key_create((key), (destructor))

Definition at line 45 of file offload_util.h.
Referenced by __offload_init_library_once().

#define thread_key_delete(*key*) pthread_key_delete(key)

Definition at line 47 of file offload_util.h.
Referenced by __offload_fini_library().

#define thread_setspecific(*key, value*) pthread_setspecific(key, value)

Definition at line 49 of file offload_util.h.
Referenced by Engine::get_auto_vars(), and Engine::get_pipeline().

7.46.2 Typedef Documentation

typedef pthread_once_t OffloadOnceControl

Definition at line 135 of file offload_util.h.

7.46.3 Function Documentation

bool __offload_parse_int_string (const char * *str,* int64_t & *value*)

Definition at line 90 of file offload_util.cpp.
Referenced by __offload_init_library_once(), and ORSL::init().

bool __offload_parse_size_string (const char * *str,* uint64_t & *new_size*)

Definition at line 32 of file offload_util.cpp.
Referenced by __offload_init_library_once().

void* DL_sym (void * *handle,* const char * *name,* const char * *version*)

Definition at line 171 of file offload_util.cpp.
Referenced by COI::init(), and MyoWrapper::LoadLibrary().

int64_t get_el_value (char * *base,* int64_t *offset,* int64_t *size*)

Definition at line 185 of file offload_util.cpp.
Referenced by OffloadDescriptor::ReadArrElements< T >::read_next().

7.47 ofldbegin.cpp File Reference

```
#include "compiler_if_target.h"
#include "offload_target.h"
#include "offload_myo_target.h"
```

Macros

- #define ALLOCATE(name) __attribute__((section(name)))
- #define DLL_LOCAL __attribute__((visibility("hidden")))

Functions

- int main (void)
- int MAIN__ (void)
- static void offload_fini ()
- static void offload_init () __attribute__((constructor(101)))

Variables

- static FuncTable::Entry __offload_entry_table_start = { 0 }
- static FuncList::Node __offload_entry_node
- static FuncTable::Entry __offload_func_table_start = { 0 }
- static FuncList::Node __offload_func_node
- static VarTable::Entry __offload_var_table_start = { 0 }
- static VarList::Node __offload_var_node

7.47.1 Macro Definition Documentation

#define ALLOCATE(*name*) __attribute__((section(name)))

Definition at line 24 of file ofldbegin.cpp.

#define DLL_LOCAL __attribute__((visibility("hidden")))

Definition at line 25 of file ofldbegin.cpp.

7.47.2 Function Documentation

int main (void)

Definition at line 37 of file ofldbegin.cpp.

int MAIN__ (void)

Definition at line 44 of file ofldbegin.cpp.

static void offload_fini ( ) [static]

Definition at line 154 of file ofldbegin.cpp.
Referenced by offload_init().

static void offload_init ( ) [static]

Definition at line 131 of file ofldbegin.cpp.

7.47.3 Variable Documentation

FuncList::Node __offload_entry_node [static]

Initial value:

```
= {
    { &__offload_entry_table_start + 1, -1 },
    0, 0
}
```

Definition at line 59 of file ofldbegin.cpp.

FuncTable::Entry __offload_entry_table_start = { 0 } [static]

Definition at line 56 of file ofldbegin.cpp.

FuncList::Node __offload_func_node [static]

Initial value:

```
= {
    { &__offload_func_table_start + 1, -1 },
    0, 0
}
```

Definition at line 72 of file ofldbegin.cpp.

FuncTable::Entry __offload_func_table_start = { 0 } [static]

Definition at line 69 of file ofldbegin.cpp.

VarList::Node __offload_var_node [static]

Initial value:

```
= {
    { &__offload_var_table_start + 1 },
    0, 0
}
```

Definition at line 85 of file ofldbegin.cpp.

VarTable::Entry __offload_var_table_start = { 0 } [static]

Definition at line 82 of file ofldbegin.cpp.

7.48 ofldend.cpp File Reference

```
#include "offload_target.h"
#include "offload_myo_target.h"
```

Macros

- #define ALLOCATE(name) __attribute__((section(name)))

Variables

- static FuncTable::Entry __offload_entry_table_end = { (const char*)-1 }
- static FuncTable::Entry __offload_func_table_end = { (const char*)-1 }
- static VarTable::Entry __offload_var_table_end = { (const char*)-1 }

7.48.1 Macro Definition Documentation

#define ALLOCATE(*name*) __attribute__((section(name)))

Definition at line 22 of file ofldend.cpp.

7.48.2 Variable Documentation

FuncTable::Entry __offload_entry_table_end = { (const char*)-1 } [static]

Definition at line 30 of file ofldend.cpp.

FuncTable::Entry __offload_func_table_end = { (const char*)-1 } [static]

Definition at line 37 of file ofldend.cpp.

VarTable::Entry __offload_var_table_end = { (const char*)-1 } [static]

Definition at line 44 of file ofldend.cpp.

7.49 orsl-lite/include/orsl-lite.h File Reference

```
#include <sched.h>
```

Classes

- struct ORSLBusySet

Macros

- #define ORSL_MAX_TAG_LEN 128
- #define ORSL_MAX_CARDS 32

Typedefs

- typedef enum ORSLBusySetType BusySetType
- typedef struct ORSLBusySet ORSLBusySet
- typedef char * ORSLTag
- typedef enum ORSLPartialGranularity ORSLPartialGranularity

Enumerations

- enum ORSLBusySetType { BUSY_SET_EMPTY = 0, BUSY_SET_PARTIAL = 1, BUSY_SET_FULL = 2 }
- enum ORSLPartialGranularity { GRAN_CARD = 0, GRAN_THREAD = 1 }

Functions

- int ORSLReserve (const int n, const int *__restrict inds, const ORSLBusySet *__restrict bsets, const ORSLTag __restrict tag)
- int ORSLTryReserve (const int n, const int *__restrict inds, const ORSLBusySet *__restrict bsets, const ORSLTag __restrict tag)
- int ORSLReservePartial (const ORSLPartialGranularity gran, const int n, const int *__restrict inds, ORSLBusySet *__restrict bsets, const ORSLTag __restrict tag)
- int ORSLRelease (const int n, const int *__restrict inds, const ORSLBusySet *__restrict bsets, const ORSLTag __restrict tag)

7.49.1 Macro Definition Documentation

#define ORSL_MAX_CARDS 32

Maximal number of cards that can be managed by ORSL
Definition at line 51 of file orsl-lite.h.
Referenced by check_args().

#define ORSL_MAX_TAG_LEN 128

Maximal length of tag in characters
Definition at line 48 of file orsl-lite.h.
Referenced by can_release_card(), can_reserve_card(), check_args(), release_card(), and reserve_card().

7.49.2 Typedef Documentation

typedef enum ORSLBusySetType BusySetType

Type of a ORSLBusySet

typedef struct ORSLBusySet ORSLBusySet

ORSLBusySet encapsulation

typedef enum ORSLPartialGranularity ORSLPartialGranularity

Granularify of partial reservation

typedef char* ORSLTag

Client tag
Definition at line 45 of file orsl-lite.h.

7.49.3 Enumeration Type Documentation

enum ORSLBusySetType

Type of a ORSLBusySet

Enumerator

BUSY_SET_EMPTY Empty set

BUSY_SET_PARTIAL Non-empty set that omits some threads

BUSY_SET_FULL A set that includes all threads on the card

Definition at line 25 of file orsl-lite.h.

enum ORSLPartialGranularity

Granularify of partial reservation

Enumerator

GRAN_CARD Card granularity

GRAN_THREAD Thread granularity

Definition at line 128 of file orsl-lite.h.

7.49.4 Function Documentation

int ORSLRelease (const int *n*, const int *__restrict *inds*, const ORSLBusySet *__restrict *bsets*, const ORSLTag __restrict *tag*)

Releases previously reserved computational resources on a set of cards.
This function will fail if any of the resources to be released were not reserved by the calling client.

See Also

ORSLReserve
ORSLTryReserve
ORSLReservePartial

Parameters

in	*n*	Number of cards to reserve resources on. Cannot be <0 or $>$ ORSL_MAX_CARDS.
in	*inds*	Indices of the cards: an integer array with n elements. Cannot be NULL if n >0. Valid card indices are from 0 to ORSL_MAX_CARDS-1. Cannot contain duplicate elements.
in	*bsets*	Requested resources on each of the card. Cannot be NULL if n >0.
in	*tag*	ORSLTag of the calling client. Cannot be NULL. Length must not exceed ORSL_MAX_TAG_LEN.

Returns

0 if the resources were successfully released
EINVAL if any of the arguments is invalid
EPERM the calling client did not reserve some of the resources it is trying to release.
ENOSYS (in ORSL Lite) if type of any of the busy sets is equal to BUSY_SET_PARTIAL

Referenced by ORSL::release().

int ORSLReserve (const int *n,* const int *__restrict *inds,* const ORSLBusySet *__restrict *bsets,* const ORSLTag __restrict *tag*)

Reserves computational resources on a set of cards. Blocks.

If any of the resources cannot be reserved, this function will block until they become available. Reservation can be recursive if performed by the same tag. A recursively reserved resource must be released the same number of times it was reserved.

See Also

ORSLTryReserve

Parameters

in	*n*	Number of cards to reserve resources on. Cannot be $<$ 0 or $>$ ORSL_MAX_CARDS.
in	*inds*	Indices of the cards: an integer array with n elements. Cannot be NULL if n $>$ 0. Valid card indices are from 0 to ORSL_MAX_CARDS-1. Cannot contain duplicate elements.
in	*bsets*	Requested resources on each of the card. Cannot be NULL if n $>$ 0.
in	*tag*	ORSLTag of the calling client. Cannot be NULL. Length must not exeed ORSL_MAX_TAG_LEN.

Returns

0 if the resources were successfully reserved
EINVAL if any of the arguments is invalid
EAGAIN limit of recursive reservations reached (not in ORSL Lite)
ENOSYS (in ORSL Lite) if type of any of the busy sets is equal to BUSY_SET_PARTIAL

Referenced by ORSL::reserve().

int ORSLReservePartial (const ORSLPartialGranularity *gran,* const int *n,* const int *__restrict *inds,* ORSLBusySet *__restrict *bsets,* const ORSLTag __restrict *tag*)

Requests reservation of some of computational resources on a set of cards. Does not block. Updates user-provided bsets to indicate which resources were reserved.

If any of the resources cannot be reserved, this function will update busy sets provided by the caller to reflect what resources were actually reserved. This function supports two granularity modes: 'card' and 'thread'. When granularity is set to 'card', a failure to reserve a thread on the card will imply that reservation has failed for the whole card. When granularity is set to 'thread', reservation on a card will be considered successful as long as at least one thread on the card was successfully reserved. Reservation can be recursive if performed by the same tag. A recursively reserved resource must be released the same number of times it was reserved.

Parameters

in	*gran*	Reservation granularity
in	*n*	Number of cards to reserve resources on. Cannot be $<$ 0 or $>$ ORSL_MAX_CARDS.
in	*inds*	Indices of the cards: an integer array with n elements. Cannot be NULL if n $>$ 0. Valid card indices are from 0 to ORSL_MAX_CARDS-1. Cannot contain duplicate elements.
in	*bsets*	Requested resources on each of the card. Cannot be NULL if n $>$ 0.
in	*tag*	ORSLTag of the calling client. Cannot be NULL. Length must not exceed ORSL_MAX_TAG_LEN.

Returns

0 if at least some of the resources were successfully reserved
EBUSY if all of the requested resources are busy
EINVAL if any of the arguments is invalid
EAGAIN limit of recursive reservations reached (not in ORSL Lite)
ENOSYS (in ORSL Lite) if type of any of the busy sets is equal to BUSY_SET_PARTIAL

int ORSLTryReserve (const int *n,* const int *__restrict inds,* const ORSLBusySet *__restrict bsets,* const ORSLTag __restrict *tag*)

Reserves computational resources on a set of cards. Does not block.

If any of the resources cannot be reserved, this function will return immediately. Reservation can be recursive if performed by the same tag. A recursively reserved resource must be released the same number of times it was reserved.

See Also

ORSLReserve

Parameters

in	*n*	Number of cards to reserve resources on. Cannot be $<$ 0 or $>$ ORSL_MAX_CARDS.
in	*inds*	Indices of the cards: an integer array with n elements. Cannot be NULL if n $>$ 0. Valid card indices are from 0 to ORSL_MAX_CARDS-1. Cannot contain duplicate elements.
in,out	*bsets*	Requested resources on each of the card. Cannot be NULL if n $>$ 0.
in	*tag*	ORSLTag of the calling client. Cannot be NULL. Length must not exceed ORSL_MAX_TAG_LEN.

Returns

0 if the resources were successfully reserved
EBUSY if some of the requested resources are busy
EINVAL if any of the arguments is invalid
EAGAIN limit of recursive reservations reached (not in ORSL Lite)
ENOSYS (in ORSL Lite) if type of any of the busy sets is equal to BUSY_SET_PARTIAL

Referenced by ORSL::try_reserve().

7.50 orsl-lite/lib/orsl-lite.c File Reference

```
#include <errno.h>
#include <string.h>
#include <limits.h>
#include <assert.h>
#include "orsl-lite/include/orsl-lite.h"
```

Macros

- #define DISABLE_SYMBOL_VERSIONING
- #define ORSLReserve0 ORSLReserve
- #define ORSLTryReserve0 ORSLTryReserve
- #define ORSLReservePartial0 ORSLReservePartial
- #define ORSLRelease0 ORSLRelease

Functions

- static int state_lock ()
- static int state_unlock ()
- static int state_wait_for_release ()
- static int state_signal_release ()
- static int check_args (const int n, const int *__restrict inds, const ORSLBusySet *__restrict bsets, const ORSLTag __restrict tag)
- static int check_bsets (const int n, const ORSLBusySet *bsets)
- static int can_reserve_card (int card, const ORSLBusySet *__restrict bset, const ORSLTag __restrict tag)

- static void reserve_card (int card, const ORSLBusySet *__restrict bset, const ORSLTag __restrict tag)
- static int can_release_card (int card, const ORSLBusySet *__restrict bset, const ORSLTag __restrict tag)
- static void release_card (int card, const ORSLBusySet *__restrict bset, const ORSLTag __restrict tag)
- int ORSLReserve0 (const int n, const int *__restrict inds, const ORSLBusySet *__restrict bsets, const ORSLTag __restrict tag)
- int ORSLTryReserve0 (const int n, const int *__restrict inds, const ORSLBusySet *__restrict bsets, const ORSLTag __restrict tag)
- int ORSLReservePartial0 (const ORSLPartialGranularity gran, const int n, const int *__restrict inds, ORSLBusySet *__restrict bsets, const ORSLTag __restrict tag)
- int ORSLRelease0 (const int n, const int *__restrict inds, const ORSLBusySet *__restrict bsets, const ORSLTag __restrict tag)

Variables

- struct {
  char owner [ORSL_MAX_TAG_LEN+1]
  unsigned long rsrv_cnt
  } rsrv_data [ORSL_MAX_CARDS]

7.50.1 Macro Definition Documentation

#define DISABLE_SYMBOL_VERSIONING

Definition at line 18 of file orsl-lite.c.

#define ORSLRelease0 ORSLRelease

Definition at line 30 of file orsl-lite.c.

#define ORSLReserve0 ORSLReserve

Definition at line 27 of file orsl-lite.c.

#define ORSLReservePartial0 ORSLReservePartial

Definition at line 29 of file orsl-lite.c.

#define ORSLTryReserve0 ORSLTryReserve

Definition at line 28 of file orsl-lite.c.

7.50.2 Function Documentation

static int can_release_card (int *card,* const ORSLBusySet *__restrict *bset,* const ORSLTag __restrict *tag*) [static]

Definition at line 179 of file orsl-lite.c.
Referenced by ORSLRelease0().

static int can_reserve_card (int *card,* const ORSLBusySet *__restrict *bset,* const ORSLTag __restrict *tag*) [static]

Definition at line 142 of file orsl-lite.c.
Referenced by ORSLReserve0(), ORSLReservePartial0(), and ORSLTryReserve0().

static int check_args (const int *n,* const int *__restrict *inds,* const ORSLBusySet *__restrict *bsets,* const ORSLTag __restrict *tag*) [static]

Definition at line 113 of file orsl-lite.c.
Referenced by ORSLRelease0(), ORSLReserve0(), ORSLReservePartial0(), and ORSLTryReserve0().

static int check_bsets (const int *n,* const ORSLBusySet * *bsets*) [static]

Definition at line 134 of file orsl-lite.c.
Referenced by ORSLRelease0(), ORSLReserve0(), ORSLReservePartial0(), and ORSLTryReserve0().

int ORSLRelease0 (const int *n,* const int *__restrict *inds,* const ORSLBusySet *__restrict *bsets,* const ORSLTag __restrict *tag*)

Definition at line 307 of file orsl-lite.c.

int ORSLReserve0 (const int *n,* const int *__restrict *inds,* const ORSLBusySet *__restrict *bsets,* const ORSLTag __restrict *tag*)

Definition at line 210 of file orsl-lite.c.

int ORSLReservePartial0 (const ORSLPartialGranularity *gran,* const int *n,* const int *__restrict *inds,* ORSLBusySet *__restrict *bsets,* const ORSLTag __restrict *tag*)

Definition at line 270 of file orsl-lite.c.

int ORSLTryReserve0 (const int *n,* const int *__restrict *inds,* const ORSLBusySet *__restrict *bsets,* const ORSLTag __restrict *tag*)

Definition at line 242 of file orsl-lite.c.

static void release_card (int *card,* const ORSLBusySet *__restrict *bset,* const ORSLTag __restrict *tag*) [static]

Definition at line 192 of file orsl-lite.c.
Referenced by ORSLRelease0().

static void reserve_card (int *card,* const ORSLBusySet *__restrict *bset,* const ORSLTag __restrict *tag*) [static]

Definition at line 157 of file orsl-lite.c.
Referenced by ORSLReserve0(), ORSLReservePartial0(), and ORSLTryReserve0().

static int state_lock ( ) [static]

Definition at line 60 of file orsl-lite.c.
Referenced by ORSLRelease0(), ORSLReserve0(), ORSLReservePartial0(), and ORSLTryReserve0().

static int state_signal_release ( ) [static]

Definition at line 96 of file orsl-lite.c.
Referenced by ORSLRelease0().

static int state_unlock ( ) [static]

Definition at line 72 of file orsl-lite.c.
Referenced by ORSLRelease0(), ORSLReserve0(), ORSLReservePartial0(), and ORSLTryReserve0().

static int state_wait_for_release ( ) [static]

Definition at line 84 of file orsl-lite.c.
Referenced by ORSLReserve0().

7.50.3 Variable Documentation

char owner[ORSL_MAX_TAG_LEN+1]

Definition at line 109 of file orsl-lite.c.
Referenced by can_release_card(), can_reserve_card(), release_card(), and reserve_card().

unsigned long rsrv_cnt

Definition at line 110 of file orsl-lite.c.
Referenced by can_reserve_card(), release_card(), and reserve_card().

struct { ... } rsrv_data[ORSL_MAX_CARDS] [static]

Referenced by can_release_card(), can_reserve_card(), release_card(), and reserve_card().

Index