

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.

This document is Copyright ©2014, Intel Corporation. All rights reserved.

Contents

1	Namespace Index	3
1.1	Namespace List	3
2	Hierarchical Index	5
2.1	Class Hierarchy	5
3	Class Index	7
3.1	Class List	7
4	File Index	9
4.1	File List	9
5	Namespace Documentation	11
5.1	COI Namespace Reference	11
5.1.1	Function Documentation	12
	fini	12
	init	12
5.1.2	Variable Documentation	12
	BufferCopy	12
	BufferCreate	12
	BufferCreateFromMemory	12
	BufferDestroy	12
	BufferGetSinkAddress	12
	BufferMap	12
	BufferRead	13
	BufferSetState	13
	BufferUnmap	13
	BufferWrite	13
	EngineGetCount	13
	EngineGetHandle	13
	EventWait	13
	is.available	13
	lib_handle	13
	PerfGetCycleFrequency	13
	PipelineCreate	14
	PipelineDestroy	14
	PipelineRunFunction	14
	ProcessCreateFromMemory	14
	ProcessDestroy	14
	ProcessGetFunctionHandles	14
	ProcessLoadLibraryFromMemory	14
	ProcessRegisterLibraries	14
5.2	ORSL Namespace Reference	14
5.2.1	Function Documentation	15
	init	15
	release	15
	reserve	15

try_reserve	15
5.2.2 Variable Documentation	15
is_enabled	15
my_tag	15
6 Class Documentation	17
6.1 _Offload_status Struct Reference	17
6.1.1 Detailed Description	17
6.1.2 Member Data Documentation	17
data_received	17
data_sent	17
device_number	17
result	17
6.2 arr_desc Struct Reference	18
6.2.1 Detailed Description	18
6.2.2 Member Data Documentation	18
base	18
dim	18
rank	18
6.3 ArrDesc Struct Reference	18
6.3.1 Detailed Description	18
6.3.2 Member Data Documentation	19
Base	19
Dim	19
Flags	19
Len	19
Offset	19
Rank	19
Reserved	19
6.4 AutoData Class Reference	19
6.4.1 Detailed Description	20
6.4.2 Constructor & Destructor Documentation	20
AutoData	20
6.4.3 Member Function Documentation	20
add_reference	20
get_reference	20
operator<	20
remove_reference	20
6.4.4 Member Data Documentation	20
cpu_addr	20
ref_count	20
6.5 VarList::BufEntry Struct Reference	20
6.5.1 Detailed Description	20
6.5.2 Member Data Documentation	21
addr	21
name	21
6.6 MicEnvVar::CardEnvVars Struct Reference	21
6.6.1 Detailed Description	21
6.6.2 Constructor & Destructor Documentation	21
CardEnvVars	21
CardEnvVars	21
~CardEnvVars	21
6.6.3 Member Function Documentation	21
add_new_env_var	21
find_var	21
6.6.4 Member Data Documentation	22
card_number	22
env_vars	22

6.7	CeanReadDim Struct Reference	22
6.7.1	Detailed Description	22
6.7.2	Member Data Documentation	22
	count	22
	size	22
6.8	CeanReadRanges Struct Reference	22
6.8.1	Detailed Description	23
6.8.2	Member Data Documentation	23
	current_number	23
	Dim	23
	init_offset	23
	last_noncont_ind	23
	ptr	23
	range_max_number	23
	range_size	23
6.9	dim_desc Struct Reference	23
6.9.1	Detailed Description	24
6.9.2	Member Data Documentation	24
	index	24
	lower	24
	size	24
	stride	24
	upper	24
6.10	DimDesc Struct Reference	24
6.10.1	Detailed Description	24
6.10.2	Member Data Documentation	24
	Extent	24
	LowerBound	25
	Mult	25
6.11	Engine Struct Reference	25
6.11.1	Detailed Description	26
6.11.2	Member Typedef Documentation	26
	PtrSet	26
	SignalMap	26
6.11.3	Member Enumeration Documentation	26
	anonymous enum	26
6.11.4	Constructor & Destructor Documentation	27
	Engine	27
	~Engine	27
6.11.5	Member Function Documentation	27
	add_lib	27
	add_signal	27
	compute	27
	destroy_thread_data	27
	find_auto_data	27
	find_ptr_data	27
	find_signal	27
	fini_process	27
	get_auto_vars	27
	get_logical_index	28
	get_physical_index	28
	get_pipeline	28
	get_process	28
	init	28
	init_device	28
	init_process	28
	init_ptr_data	28
	insert_auto_data	28

insert_ptr_data	28
load_libraries	28
remove_auto_data	29
remove_ptr_data	29
set_indexes	29
6.11.6 Friends And Related Function Documentation	29
__offload_fini_library	29
__offload_init_library_once	29
6.11.7 Member Data Documentation	29
c_signal_max	29
c_signal_names	29
m_func_names	29
m_funcs	29
m_images	29
m_index	30
m_lock	30
m_persist_list	30
m_physical_index	30
m_proc_number	30
m_process	30
m_ptr_lock	30
m_ptr_set	30
m_ready	30
m_signal_lock	30
m_signal_map	30
6.12 FuncTable::Entry Struct Reference	31
6.12.1 Detailed Description	31
6.12.2 Member Data Documentation	31
func	31
name	31
6.13 VarTable::Entry Struct Reference	31
6.13.1 Detailed Description	32
6.13.2 Member Data Documentation	32
addr	32
name	32
6.14 FptrTableEntry Struct Reference	32
6.14.1 Detailed Description	32
6.14.2 Member Data Documentation	32
funcAddr	32
funcName	32
localThunkAddr	33
6.15 FuncList Class Reference	33
6.15.1 Detailed Description	33
6.15.2 Constructor & Destructor Documentation	33
FuncList	33
6.15.3 Member Function Documentation	33
add_table	33
dump	33
find_addr	34
find_name	34
max_name_length	34
6.15.4 Member Data Documentation	34
m_max_name_len	34
6.16 FuncTable Struct Reference	34
6.16.1 Detailed Description	34
6.16.2 Member Data Documentation	34
entries	34
max_name_len	34

6.17	FunctionDescriptor Struct Reference	35
6.17.1	Detailed Description	35
6.17.2	Member Data Documentation	35
	console_enabled	35
	data	35
	data_offset	35
	in_datalen	35
	offload_number	35
	offload_report_level	35
	out_datalen	35
	timer_enabled	36
	vars_num	36
6.18	Image Struct Reference	36
6.18.1	Detailed Description	36
6.18.2	Member Data Documentation	36
	data	36
	size	36
6.19	InitTableEntry Struct Reference	36
6.19.1	Detailed Description	37
6.19.2	Member Data Documentation	37
	func	37
6.20	VarList::Iterator Class Reference	37
6.20.1	Detailed Description	37
6.20.2	Constructor & Destructor Documentation	37
	Iterator	37
	Iterator	38
6.20.3	Member Function Documentation	38
	new_node	38
	operator!=	38
	operator*	38
	operator++	38
	operator==	38
6.20.4	Member Data Documentation	38
	m_entry	38
	m_node	38
6.21	kmp_affinity_mask_target_t Struct Reference	38
6.21.1	Detailed Description	38
6.21.2	Member Data Documentation	39
	mask	39
6.22	mic_lib::kmp_create_affinity_mask_target Interface Reference	39
6.22.1	Detailed Description	39
6.22.2	Constructor & Destructor Documentation	39
	kmp_create_affinity_mask_target	39
6.23	mic_lib::kmp_destroy_affinity_mask_target Interface Reference	39
6.23.1	Detailed Description	39
6.23.2	Constructor & Destructor Documentation	39
	kmp_destroy_affinity_mask_target	39
6.24	mic_lib::kmp_get_affinity_mask_proc_target Interface Reference	39
6.24.1	Detailed Description	40
6.24.2	Constructor & Destructor Documentation	40
	kmp_get_affinity_mask_proc_target	40
6.25	mic_lib::kmp_get_affinity_max_proc_target Interface Reference	40
6.25.1	Detailed Description	40
6.25.2	Constructor & Destructor Documentation	40
	kmp_get_affinity_max_proc_target	40
6.26	mic_lib::kmp_get_affinity_target Interface Reference	40
6.26.1	Detailed Description	40
6.26.2	Constructor & Destructor Documentation	40

	kmp_get_affinity_target	40
6.27	mic_lib::kmp_get_blocktime_target Interface Reference	40
6.27.1	Detailed Description	41
6.27.2	Constructor & Destructor Documentation	41
	kmp_get_blocktime_target	41
6.28	mic_lib::kmp_get_library_target Interface Reference	41
6.28.1	Detailed Description	41
6.28.2	Constructor & Destructor Documentation	41
	kmp_get_library_target	41
6.29	mic_lib::kmp_get_stacksize_s_target Interface Reference	41
6.29.1	Detailed Description	41
6.29.2	Constructor & Destructor Documentation	41
	kmp_get_stacksize_s_target	41
6.30	mic_lib::kmp_get_stacksize_target Interface Reference	41
6.30.1	Detailed Description	42
6.30.2	Constructor & Destructor Documentation	42
	kmp_get_stacksize_target	42
6.31	mic_lib::kmp_set_affinity_mask_proc_target Interface Reference	42
6.31.1	Detailed Description	42
6.31.2	Constructor & Destructor Documentation	42
	kmp_set_affinity_mask_proc_target	42
6.32	mic_lib::kmp_set_affinity_target Interface Reference	42
6.32.1	Detailed Description	42
6.32.2	Constructor & Destructor Documentation	42
	kmp_set_affinity_target	42
6.33	mic_lib::kmp_set_blocktime_target Interface Reference	42
6.33.1	Detailed Description	43
6.33.2	Constructor & Destructor Documentation	43
	kmp_set_blocktime_target	43
6.34	mic_lib::kmp_set_defaults_target Interface Reference	43
6.34.1	Detailed Description	43
6.34.2	Constructor & Destructor Documentation	43
	kmp_set_defaults_target	43
6.35	mic_lib::kmp_set_library_serial_target Interface Reference	43
6.35.1	Detailed Description	43
6.35.2	Constructor & Destructor Documentation	43
	kmp_set_library_serial_target	43
6.36	mic_lib::kmp_set_library_target Interface Reference	43
6.36.1	Detailed Description	44
6.36.2	Constructor & Destructor Documentation	44
	kmp_set_library_target	44
6.37	mic_lib::kmp_set_library_throughput_target Interface Reference	44
6.37.1	Detailed Description	44
6.37.2	Constructor & Destructor Documentation	44
	kmp_set_library_throughput_target	44
6.38	mic_lib::kmp_set_library_turnaround_target Interface Reference	44
6.38.1	Detailed Description	44
6.38.2	Constructor & Destructor Documentation	44
	kmp_set_library_turnaround_target	44
6.39	mic_lib::kmp_set_stacksize_s_target Interface Reference	44
6.39.1	Detailed Description	45
6.39.2	Constructor & Destructor Documentation	45
	kmp_set_stacksize_s_target	45
6.40	mic_lib::kmp_set_stacksize_target Interface Reference	45
6.40.1	Detailed Description	45
6.40.2	Constructor & Destructor Documentation	45
	kmp_set_stacksize_target	45
6.41	mic_lib::kmp_unset_affinity_mask_proc_target Interface Reference	45

6.41.1	Detailed Description	45
6.41.2	Constructor & Destructor Documentation	45
	kmp_unset_affinity_mask_proc_target	45
6.42	Marshaller Class Reference	45
6.42.1	Detailed Description	46
6.42.2	Constructor & Destructor Documentation	46
	Marshaller	46
6.42.3	Member Function Documentation	46
	get_buffer_size	46
	get_buffer_start	46
	get_tfr_size	46
	init_buffer	46
	receive_data	46
	receive_func_ptr	47
	send_data	47
	send_func_ptr	47
6.42.4	Member Data Documentation	47
	buffer_ptr	47
	buffer_size	47
	buffer_start	47
	tfr_size	47
6.43	MemRange Class Reference	47
6.43.1	Detailed Description	48
6.43.2	Constructor & Destructor Documentation	48
	MemRange	48
	MemRange	48
6.43.3	Member Function Documentation	48
	contains	48
	end	48
	length	48
	overlaps	48
	start	48
6.43.4	Member Data Documentation	48
	m.length	48
	m.start	48
6.44	mic.lib Module Reference	49
6.44.1	Detailed Description	50
6.44.2	Member Data Documentation	50
	default_target_number	50
	default_target_type	50
	target_mic	50
6.45	MicEnvVar Struct Reference	50
6.45.1	Detailed Description	50
6.45.2	Constructor & Destructor Documentation	51
	MicEnvVar	51
	~MicEnvVar	51
6.45.3	Member Function Documentation	51
	add_env_var	51
	analyze_env_var	51
	create_envron_for_card	51
	get_card	51
	get_env_var_kind	51
	mic_parse_env_var_list	51
	set_prefix	51
6.45.4	Member Data Documentation	51
	any_card	51
	card_spec_list	51
	common_vars	52

prefix	52
6.46 mutex_locker_t Struct Reference	52
6.46.1 Detailed Description	52
6.46.2 Constructor & Destructor Documentation	52
mutex_locker_t	52
~mutex_locker_t	52
6.46.3 Member Data Documentation	52
m_mutex	52
6.47 mutex_t Struct Reference	52
6.47.1 Detailed Description	53
6.47.2 Constructor & Destructor Documentation	53
mutex_t	53
~mutex_t	53
6.47.3 Member Function Documentation	53
lock	53
unlock	53
6.47.4 Member Data Documentation	53
m_lock	53
6.48 MyoTable Struct Reference	54
6.48.1 Detailed Description	54
6.48.2 Constructor & Destructor Documentation	54
MyoTable	54
6.48.3 Member Data Documentation	54
var_tab	54
var_tab_len	54
6.49 MyoWrapper Class Reference	54
6.49.1 Detailed Description	55
6.49.2 Constructor & Destructor Documentation	55
MyoWrapper	55
6.49.3 Member Function Documentation	55
Acquire	55
CheckResult	55
GetResult	55
HostFptrTableRegister	55
HostVarTablePropagate	55
is_available	55
LibFini	56
LibInit	56
LoadLibrary	56
Release	56
RemoteCall	56
RemoteThunkCall	56
SharedAlignedFree	56
SharedAlignedMalloc	56
SharedFree	56
SharedMalloc	56
UnloadLibrary	56
6.49.4 Member Data Documentation	57
m_acquire	57
m_get_result	57
m_host_fptr_table_register	57
m_host_var_table_propagate	57
m_is_available	57
m_lib_fini	57
m_lib_handle	57
m_lib_init	57
m_release	57
m_remote_call	57

m_remote_thunk_call	57
m_shared_aligned_free	57
m_shared_aligned_malloc	58
m_shared_free	58
m_shared_malloc	58
6.50 TableList< T >::Node Struct Reference	58
6.50.1 Detailed Description	58
6.50.2 Member Data Documentation	58
next	58
prev	58
table	58
6.51 mic_lib::offload_get_device_number Interface Reference	58
6.51.1 Detailed Description	59
6.51.2 Constructor & Destructor Documentation	59
offload_get_device_number	59
6.52 mic_lib::offload_get_physical_device_number Interface Reference	59
6.52.1 Detailed Description	59
6.52.2 Constructor & Destructor Documentation	59
offload_get_physical_device_number	59
6.53 mic_lib::offload_number_of_devices Interface Reference	59
6.53.1 Detailed Description	59
6.53.2 Constructor & Destructor Documentation	59
offload_number_of_devices	59
6.54 mic_lib::offload_report Interface Reference	59
6.54.1 Detailed Description	60
6.54.2 Constructor & Destructor Documentation	60
offload_report	60
6.55 mic_lib::offload_signaled Interface Reference	60
6.55.1 Detailed Description	60
6.55.2 Constructor & Destructor Documentation	60
offload_signaled	60
6.56 mic_lib::offload_status Type Reference	60
6.56.1 Detailed Description	60
6.56.2 Member Data Documentation	60
data_received	60
data_sent	60
device_number	60
result	61
6.57 OffloadDescriptor Class Reference	61
6.57.1 Detailed Description	62
6.57.2 Member Typedef Documentation	62
BufferList	62
BufferList	62
6.57.3 Constructor & Destructor Documentation	63
OffloadDescriptor	63
~OffloadDescriptor	63
~OffloadDescriptor	63
OffloadDescriptor	63
6.57.4 Member Function Documentation	63
alloc_ptr_data	63
cleanup	63
compute	63
find_ptr_data	63
gather_copyin_data	63
gather_copyout_data	63
gen_var_descs_for_pointer_array	63
get_offload_number	64
get_timer_data	64

init_mic_address	64
init_static_ptr_data	64
is_signaled	64
merge_var_descs	64
nullify_target_stack	64
offload	64
offload	64
offload_finish	64
offload_stack_memory_manager	64
receive_pointer_data	65
recieve_noncontiguous_pointer_data	65
report_coi_error	65
scatter_copyin_data	65
scatter_copyout_data	65
send_noncontiguous_pointer_data	65
send_pointer_data	65
set_offload_number	65
setup_descriptors	65
setup_misc_data	65
translate_coi_error	66
wait_dependencies	66
6.57.5 Member Data Documentation	66
m_buffers	66
m_compute_buffers	66
m_destroy_buffers	66
m_destroy_stack	66
m_device	66
m_func_desc	66
m_func_desc_size	66
m_in	66
m_in_datalen	67
m_in_deps	67
m_in_deps_total	67
m_inout_buf	67
m_is_mandatory	67
m_is_openmp	67
m_need_runfunction	67
m_offload_number	67
m_out	67
m_out_datalen	67
m_out_deps	67
m_out_deps_total	68
m_stack_ptr_data	68
m_status	68
m_timer_data	68
m_vars	68
m_vars_extra	68
m_vars_total	68
6.58 mic_lib::omp_destroy_lock_target Interface Reference	68
6.58.1 Detailed Description	68
6.58.2 Constructor & Destructor Documentation	69
omp_destroy_lock_target	69
6.59 mic_lib::omp_destroy_nest_lock_target Interface Reference	69
6.59.1 Detailed Description	69
6.59.2 Constructor & Destructor Documentation	69
omp_destroy_nest_lock_target	69
6.60 mic_lib::omp_get_dynamic_target Interface Reference	69
6.60.1 Detailed Description	69

6.60.2	Constructor & Destructor Documentation	69
	omp_get_dynamic_target	69
6.61	mic_lib::omp_get_max_threads_target Interface Reference	69
6.61.1	Detailed Description	69
6.61.2	Constructor & Destructor Documentation	70
	omp_get_max_threads_target	70
6.62	mic_lib::omp_get_nested_target Interface Reference	70
6.62.1	Detailed Description	70
6.62.2	Constructor & Destructor Documentation	70
	omp_get_nested_target	70
6.63	mic_lib::omp_get_num_procs_target Interface Reference	70
6.63.1	Detailed Description	70
6.63.2	Constructor & Destructor Documentation	70
	omp_get_num_procs_target	70
6.64	mic_lib::omp_get_schedule_target Interface Reference	70
6.64.1	Detailed Description	70
6.64.2	Constructor & Destructor Documentation	71
	omp_get_schedule_target	71
6.65	mic_lib::omp_init_lock_target Interface Reference	71
6.65.1	Detailed Description	71
6.65.2	Constructor & Destructor Documentation	71
	omp_init_lock_target	71
6.66	mic_lib::omp_init_nest_lock_target Interface Reference	71
6.66.1	Detailed Description	71
6.66.2	Constructor & Destructor Documentation	71
	omp_init_nest_lock_target	71
6.67	omp_lock_target.t Struct Reference	71
6.67.1	Detailed Description	72
6.67.2	Member Data Documentation	72
	lock	72
6.68	omp_nest_lock_target.t Struct Reference	72
6.68.1	Detailed Description	72
6.68.2	Member Data Documentation	72
	lock	72
6.69	mic_lib::omp_set_dynamic_target Interface Reference	72
6.69.1	Detailed Description	72
6.69.2	Constructor & Destructor Documentation	72
	omp_set_dynamic_target	72
6.70	mic_lib::omp_set_lock_target Interface Reference	73
6.70.1	Detailed Description	73
6.70.2	Constructor & Destructor Documentation	73
	omp_set_lock_target	73
6.71	mic_lib::omp_set_nest_lock_target Interface Reference	73
6.71.1	Detailed Description	73
6.71.2	Constructor & Destructor Documentation	73
	omp_set_nest_lock_target	73
6.72	mic_lib::omp_set_nested_target Interface Reference	73
6.72.1	Detailed Description	73
6.72.2	Constructor & Destructor Documentation	73
	omp_set_nested_target	73
6.73	mic_lib::omp_set_num_threads_target Interface Reference	74
6.73.1	Detailed Description	74
6.73.2	Constructor & Destructor Documentation	74
	omp_set_num_threads_target	74
6.74	mic_lib::omp_set_schedule_target Interface Reference	74
6.74.1	Detailed Description	74
6.74.2	Constructor & Destructor Documentation	74
	omp_set_schedule_target	74

6.75	mic.lib::omp_test_lock_target Interface Reference	74
6.75.1	Detailed Description	74
6.75.2	Constructor & Destructor Documentation	74
	omp_test_lock_target	74
6.76	mic.lib::omp_test_nest_lock_target Interface Reference	75
6.76.1	Detailed Description	75
6.76.2	Constructor & Destructor Documentation	75
	omp_test_nest_lock_target	75
6.77	mic.lib::omp_unset_lock_target Interface Reference	75
6.77.1	Detailed Description	75
6.77.2	Constructor & Destructor Documentation	75
	omp_unset_lock_target	75
6.78	mic.lib::omp_unset_nest_lock_target Interface Reference	75
6.78.1	Detailed Description	75
6.78.2	Constructor & Destructor Documentation	75
	omp_unset_nest_lock_target	75
6.79	ORSLBusySet Struct Reference	76
6.79.1	Detailed Description	76
6.79.2	Member Data Documentation	76
	type	76
6.80	PersistData Struct Reference	76
6.80.1	Detailed Description	76
6.80.2	Constructor & Destructor Documentation	76
	PersistData	76
6.80.3	Member Data Documentation	76
	cpu_stack_addr	76
	routine_id	76
	stack_cpu_addr	77
	stack_ptr_data	77
6.81	PtrData Class Reference	77
6.81.1	Detailed Description	77
6.81.2	Constructor & Destructor Documentation	77
	PtrData	77
	PtrData	77
6.81.3	Member Function Documentation	78
	add_reference	78
	get_reference	78
	operator<	78
	remove_reference	78
6.81.4	Member Data Documentation	78
	alloc_disp	78
	alloc_ptr_data_lock	78
	cpu_addr	78
	cpu_buf	78
	is_static	78
	mic_addr	78
	mic_buf	79
	mic_offset	79
	ref_count	79
6.82	OffloadDescriptor::ReadArrElements< T > Class Template Reference	79
6.82.1	Detailed Description	79
6.82.2	Constructor & Destructor Documentation	79
	ReadArrElements	79
6.82.3	Member Function Documentation	80
	read_next	80
6.82.4	Member Data Documentation	80
	base	80
	count	80

	el_size	80
	is_empty	80
	length_cur	80
	offset	80
	ranges	80
	size	80
	val	81
6.83	RefInfo Struct Reference	81
6.83.1	Detailed Description	81
6.83.2	Constructor & Destructor Documentation	81
	RefInfo	81
6.83.3	Member Data Documentation	81
	count	81
	is_added	81
6.84	TableList< T > Class Template Reference	81
6.84.1	Detailed Description	82
6.84.2	Member Typedef Documentation	82
	Table	82
6.84.3	Constructor & Destructor Documentation	82
	TableList	82
6.84.4	Member Function Documentation	82
	add_table	82
	remove_table	82
6.84.5	Member Data Documentation	82
	m_head	82
	m_lock	82
6.85	TargetImage Struct Reference	82
6.85.1	Detailed Description	83
6.85.2	Constructor & Destructor Documentation	83
	TargetImage	83
6.85.3	Member Data Documentation	83
	data	83
	name	83
	offset	83
	origin	83
	size	83
6.86	Thread Struct Reference	84
6.86.1	Detailed Description	84
6.86.2	Constructor & Destructor Documentation	84
	Thread	84
	~Thread	84
6.86.3	Member Function Documentation	84
	get_auto_vars	84
	get_pipeline	84
	set_pipeline	84
6.86.4	Member Data Documentation	84
	m_addr_coipipe_counter	84
	m_auto_vars	84
	m_pipelines	85
6.87	VarDesc Struct Reference	85
6.87.1	Detailed Description	86
6.87.2	Member Data Documentation	86
	"@5	86
	"@7	86
	align	86
	alloc	86
	alloc_disp	87
	alloc_if	87

bits	87
bits	87
count	87
direction	87
disp	88
dst	88
flags	88
free_if	88
has_length	89
in	89
into	89
is_noncont_dst	89
is_noncont_src	89
is_stack_buf	89
is_static	89
is_static_dstn	90
mic_offset	90
offset	90
out	90
ptr	90
ptr_arr_offset	91
sink_addr	91
size	91
src	91
type	91
6.88 VarDesc2 Struct Reference	92
6.88.1 Detailed Description	92
6.88.2 Member Data Documentation	92
dname	92
sname	92
6.89 VarDesc3 Struct Reference	92
6.89.1 Detailed Description	93
6.89.2 Member Data Documentation	93
align_array	93
alloc_elements	93
alloc_if_array	93
alloc_start	93
array_fields	94
extent_elements	94
extent_start	94
free_if_array	94
into_elements	94
into_start	94
ptr_array	94
6.90 OffloadDescriptor::VarExtra Struct Reference	95
6.90.1 Detailed Description	95
6.90.2 Member Data Documentation	95
auto_data	95
cpu_disp	95
cpu_offset	95
dst_data	95
is_arr_ptr_el	95
ptr_arr_offset	95
read_rng_dst	96
read_rng_src	96
src_data	96
6.91 VarList Class Reference	96
6.91.1 Detailed Description	96

6.91.2	Constructor & Destructor Documentation	97
	VarList	97
6.91.3	Member Function Documentation	97
	begin	97
	dump	97
	end	97
	table_copy	97
	table_patch_names	97
	table_size	97
6.92	VarTable Struct Reference	97
6.92.1	Detailed Description	97
6.92.2	Member Data Documentation	98
	entries	98
6.93	MicEnvVar::VarValue Struct Reference	98
6.93.1	Detailed Description	98
6.93.2	Constructor & Destructor Documentation	98
	VarValue	98
	~VarValue	98
6.93.3	Member Data Documentation	98
	env_var	98
	env_var_value	98
	length	98
7	File Documentation	99
7.1	cean_util.cpp File Reference	99
7.1.1	Typedef Documentation	99
	fpp	99
7.1.2	Function Documentation	99
	__arr_data_offset_and_length	99
	cean_get_transf_size	100
	cean_ranges_match	100
	generate_mem_ranges	100
	generate_mem_ranges_one_rank	100
	generate_one_range	100
	get_next_range	100
	init_read_ranges_arr_desc	100
	is_arr_desc_contiguous	100
7.1.3	Variable Documentation	100
	last_left	100
	last_right	100
7.2	cean_util.h File Reference	100
7.2.1	Macro Definition Documentation	101
	__arr_desc_dump	101
	__arr_desc_length	101
7.2.2	Function Documentation	101
	__arr_data_offset_and_length	101
	cean_get_transf_size	101
	cean_ranges_match	101
	get_next_range	101
	init_read_ranges_arr_desc	101
	is_arr_desc_contiguous	102
7.3	coi/coi_client.cpp File Reference	102
7.3.1	Macro Definition Documentation	103
	COI_VERSION1	103
	COI_VERSION2	103
7.4	coi/coi_client.h File Reference	103
7.4.1	Macro Definition Documentation	103
	MIC_ENGINES_MAX	103

7.5	coi/coi_server.cpp File Reference	104
7.5.1	Function Documentation	104
	server_compute	104
	server_init	104
	server_var_table_copy	104
	server_var_table_size	104
7.6	coi/coi_server.h File Reference	104
7.6.1	Macro Definition Documentation	105
	BufferAddRef	105
	BufferReleaseRef	105
	EngineGetIndex	105
	PipelineStartExecutingRunFunctions	105
	ProcessWaitForShutdown	106
7.7	compiler_if_host.cpp File Reference	106
7.7.1	Function Documentation	106
	OFFLOAD_CALL_COUNT	106
	OFFLOAD_OFFLOAD	106
	OFFLOAD_OFFLOAD1	106
	OFFLOAD_OFFLOAD2	107
	offload_offload_wrap	107
	OFFLOAD_TARGET_ACQUIRE	107
	OFFLOAD_TARGET_ACQUIRE1	107
7.7.2	Variable Documentation	107
	offload_call_count	107
7.8	compiler_if_host.h File Reference	107
7.8.1	Detailed Description	108
7.8.2	Macro Definition Documentation	108
	OFFLOAD_CALL_COUNT	108
	OFFLOAD_OFFLOAD	108
	OFFLOAD_OFFLOAD1	108
	OFFLOAD_OFFLOAD2	108
	OFFLOAD_TARGET_ACQUIRE	109
	OFFLOAD_TARGET_ACQUIRE1	109
7.8.3	Function Documentation	109
	OFFLOAD_CALL_COUNT	109
	OFFLOAD_OFFLOAD	109
	OFFLOAD_OFFLOAD1	109
	OFFLOAD_OFFLOAD2	110
	OFFLOAD_TARGET_ACQUIRE	110
	OFFLOAD_TARGET_ACQUIRE1	110
7.9	compiler_if_target.cpp File Reference	110
7.9.1	Function Documentation	110
	OFFLOAD_TARGET_ENTER	110
	OFFLOAD_TARGET_LEAVE	110
	OFFLOAD_TARGET_MAIN	110
7.10	compiler_if_target.h File Reference	110
7.10.1	Detailed Description	111
7.10.2	Macro Definition Documentation	111
	OFFLOAD_TARGET_ENTER	111
	OFFLOAD_TARGET_LEAVE	111
	OFFLOAD_TARGET_MAIN	111
7.10.3	Function Documentation	111
	OFFLOAD_TARGET_ENTER	111
	OFFLOAD_TARGET_LEAVE	112
	OFFLOAD_TARGET_MAIN	112
7.11	dv_util.cpp File Reference	112
7.11.1	Function Documentation	112
	_dv_data_length	112

__dv_data_length	112
__dv_is_allocated	112
__dv_is_contiguous	112
init_read_ranges_dv	112
7.12 dv_util.h File Reference	112
7.12.1 Macro Definition Documentation	113
__dv_desc_dump	113
ArrDescFlagsContiguous	113
ArrDescFlagsDefined	113
ArrDescFlagsNodealloc	113
ArrDescMaxArrayRank	113
7.12.2 Typedef Documentation	113
ArrDesc	113
DimDesc	113
dv_size	113
pArrDesc	113
7.12.3 Function Documentation	114
__dv_data_length	114
__dv_data_length	114
__dv_is_allocated	114
__dv_is_contiguous	114
init_read_ranges_dv	114
7.13 liboffload_error.c File Reference	114
7.13.1 Macro Definition Documentation	114
va_copy	114
7.13.2 Function Documentation	114
__liboffload_error_support	114
report_get_host_stage_str	114
report_get_message_str	115
report_get_target_stage_str	115
7.14 liboffload_error_codes.h File Reference	115
7.14.1 Macro Definition Documentation	116
LIBOFFLOAD_ABORT	116
LIBOFFLOAD_ERROR	116
test_msg_cat	116
test_msg_cat1	117
7.14.2 Enumeration Type Documentation	117
error_types	117
OffloadHostPhase	120
OffloadTargetPhase	120
7.14.3 Function Documentation	120
__liboffload_error_support	120
__liboffload_report_support	121
offload_get_message_str	121
report_get_host_stage_str	121
report_get_message_str	121
report_get_target_stage_str	121
write_message	121
7.15 liboffload_msg.c File Reference	121
7.15.1 Macro Definition Documentation	121
DYNART_STDERR_PUTS	121
7.15.2 Function Documentation	121
offload_get_message_str	121
write_message	121
7.16 liboffload_msg.h File Reference	122
7.16.1 Macro Definition Documentation	123
MESSAGE_TABLE_NAME	123
7.16.2 Enumeration Type Documentation	123

anonymous enum	123
7.16.3 Variable Documentation	126
MESSAGE_TABLE_NAME	126
7.17 mic_lib.f90 File Reference	127
7.18 offload.h File Reference	127
7.18.1 Macro Definition Documentation	129
DEFAULT_TARGET_NUMBER	129
DEFAULT_TARGET_TYPE	129
OFFLOAD_STATUS_INIT	129
OFFLOAD_STATUS_INITIALIZER	129
TARGET_ATTRIBUTE	129
7.18.2 Typedef Documentation	130
TARGET_TYPE	130
7.18.3 Enumeration Type Documentation	130
_Offload_result	130
TARGET_TYPE	130
7.18.4 Function Documentation	130
_Offload_get_device_number	130
_Offload_get_physical_device_number	130
_Offload_number_of_devices	130
_Offload_report	130
_Offload_shared_aligned_free	130
_Offload_shared_aligned_malloc	130
_Offload_shared_free	130
_Offload_shared_malloc	131
_Offload_signaled	131
kmp_create_affinity_mask_target	131
kmp_destroy_affinity_mask_target	131
kmp_get_affinity_mask_proc_target	131
kmp_get_affinity_max_proc_target	131
kmp_get_affinity_target	131
kmp_get_blocktime_target	131
kmp_get_library_target	131
kmp_get_stacksize_s_target	131
kmp_get_stacksize_target	131
kmp_set_affinity_mask_proc_target	131
kmp_set_affinity_target	131
kmp_set_blocktime_target	132
kmp_set_defaults_target	132
kmp_set_library_serial_target	132
kmp_set_library_target	132
kmp_set_library_throughput_target	132
kmp_set_library_turnaround_target	132
kmp_set_stacksize_s_target	132
kmp_set_stacksize_target	132
kmp_unset_affinity_mask_proc_target	132
omp_destroy_lock_target	132
omp_destroy_nest_lock_target	132
omp_get_default_device	132
omp_get_dynamic_target	132
omp_get_max_threads_target	132
omp_get_nested_target	132
omp_get_num_devices	133
omp_get_num_procs_target	133
omp_get_schedule_target	133
omp_init_lock_target	133
omp_init_nest_lock_target	133
omp_set_default_device	133

omp_set_dynamic_target	133
omp_set_lock_target	133
omp_set_nest_lock_target	133
omp_set_nested_target	133
omp_set_num_threads_target	133
omp_set_schedule_target	133
omp_test_lock_target	133
omp_test_nest_lock_target	133
omp_unset_lock_target	134
omp_unset_nest_lock_target	134
7.19 offload.common.cpp File Reference	134
7.19.1 Function Documentation	134
OFFLOAD_MALLOC	134
7.20 offload.common.h File Reference	134
7.20.1 Detailed Description	136
7.20.2 Macro Definition Documentation	136
OFFLOAD_DEBUG_DUMP_BYTES	136
OFFLOAD_DEBUG_LOG	136
OFFLOAD_DEBUG_PRINT_PREFIX	136
OFFLOAD_DO_TRACE	136
OFFLOAD_FREE	136
OFFLOAD_MALLOC	136
OFFLOAD_PREFIX	136
OFFLOAD_TRACE	136
VAR_TYPE_IS_DV_DATA	136
VAR_TYPE_IS_DV_DATA_SLICE	137
VAR_TYPE_IS_PTR	137
VAR_TYPE_IS_SCALAR	137
7.20.3 Typedef Documentation	137
OFFLOAD	137
7.20.4 Enumeration Type Documentation	137
OffloadItemType	137
OffloadParameterType	138
7.20.5 Function Documentation	138
OFFLOAD_MALLOC	138
7.20.6 Variable Documentation	138
console.enabled	138
flag_align.is_array	138
flag_alloc.elements.is_array	138
flag_alloc.elements.is_scalar	138
flag_alloc.if.is_array	138
flag_alloc.start.is_array	138
flag_alloc.start.is_scalar	139
flag_extent.elements.is_array	139
flag_extent.elements.is_scalar	139
flag_extent.start.is_array	139
flag_extent.start.is_scalar	139
flag_free.if.is_array	139
flag_into.elements.is_array	139
flag_into.elements.is_scalar	139
flag_into.start.is_array	139
flag_into.start.is_scalar	139
mic_index	139
offload_number	139
offload_report_level	140
prefix	140
7.21 offload.engine.cpp File Reference	140
7.21.1 Function Documentation	140

host_entry_cmp	140
target_entry_cmp	140
7.22 offload.engine.h File Reference	140
7.22.1 Macro Definition Documentation	141
check_result	141
7.22.2 Typedef Documentation	141
AutoSet	141
PersistDataList	141
PtrDataList	141
TargetImageList	141
7.23 offload.env.cpp File Reference	141
7.24 offload.env.h File Reference	142
7.24.1 Enumeration Type Documentation	142
MicEnvVarKind	142
7.25 offload.host.cpp File Reference	142
7.25.1 Macro Definition Documentation	144
GET_OFFLOAD_NUMBER	144
PATH_SEPARATOR	144
7.25.2 Function Documentation	144
_dbg_target_so_loaded	144
_dbg_target_so_unloaded	144
_offload_console_trace	144
_offload_fini_library	144
_offload_init_library	144
_offload_init_library_once	144
_offload_register_image	144
_offload_unregister_image	144
_Offload_get_device_number	144
_Offload_get_physical_device_number	145
_Offload_number_of_devices	145
_Offload_report	145
_Offload_signaled	145
get_arr_desc_numbers	145
make_arr_desc	145
offload_get_src_base	145
7.25.3 Variable Documentation	145
_dbg_api_major_version	145
_dbg_api_minor_version	145
_dbg_is_attached	145
_dbg_target_exe_name	145
_dbg_target_id	145
_dbg_target_so_pid	146
_offload_active_wait	146
_offload_init_type	146
_offload_use_2mb_buffers	146
_offload_use_async_buffer_read	146
_offload_use_async_buffer_write	146
_omp_device_num	146
_target_exe	146
_target_libs	146
_target_libs_list	146
_target_libs_lock	146
console.enabled	146
cpu.frequency	147
htrace.envname	147
mic_buffer_size	147
mic_engines	147
mic_engines.total	147

mic_env_vars	147
mic_library_path	147
mic_proxy_fs_root	147
mic_proxy_io	147
mic_stack_size	147
mic_thread_key	147
mic_use_2mb_buffers_envname	148
mic_use_async_buffer_read_envname	148
mic_use_async_buffer_write_envname	148
offload_active_wait_envname	148
offload_init_envname	148
offload_number	148
offload_report_envname	148
omp_device_num_envname	148
prefix	148
stack_alloc_lock	148
timer_envname	149
vardesc_direction_as_string	149
vardesc_type_as_string	149
7.26 offload_host.h File Reference	149
7.26.1 Detailed Description	150
7.26.2 Macro Definition Documentation	150
MAX_TARGET_NAME	150
7.26.3 Enumeration Type Documentation	151
OffloadInitType	151
7.26.4 Function Documentation	151
_dbg_target_so_loaded	151
_dbg_target_so_unloaded	151
_offload_init_library	151
_offload_register_image	151
_offload_unregister_image	151
7.26.5 Variable Documentation	151
_dbg_api_major_version	151
_dbg_api_minor_version	151
_dbg_is_attached	151
_dbg_target_exe_name	151
_dbg_target_id	151
_dbg_target_so_pid	152
_offload_init_type	152
_offload_use_2mb_buffers	152
_omp_device_num	152
_target_exe	152
cpu_frequency	152
mic_buffer_size	152
mic_engines	152
mic_engines_total	152
mic_env_vars	152
mic_library_path	152
mic_proxy_fs_root	153
mic_proxy_io	153
mic_stack_size	153
mic_thread_key	153
7.27 offload_myo_host.cpp File Reference	153
7.27.1 Macro Definition Documentation	154
MYO_TABLE_END_MARKER	154
MYO_VERSION1	154
7.27.2 Typedef Documentation	154
MyoTableList	154

7.27.3	Function Documentation	154
	_cilkrts_cilk_for_32	154
	_cilkrts_cilk_for_64	154
	_intel_cilk_for_32_offload	154
	_intel_cilk_for_64_offload	154
	_offload_myofptr_table_register	155
	_offload_myoshared_init_table_register	155
	_offload_myoshared_table_register	155
	_offload_myofini	155
	_offload_myoinit	155
	_offload_myoinit_once	155
	_offload_myoiRemotelThunkCall	155
	_offload_myolsAvailable	155
	_offload_myoloadLibrary	155
	_offload_myoloadLibrary_once	155
	_offload_myoregisterTables	155
	Offload_shared_aligned_free	155
	Offload_shared_aligned_malloc	156
	Offload_shared_free	156
	Offload_shared_malloc	156
	fptr_table_entries	156
	shared_table_entries	156
7.27.4	Variable Documentation	156
	_myof_table_list	156
	_myof_table_lock	156
	_myof_tables	156
	myof_is_available	156
	myof_wrapper	156
7.28	offload_myof.h File Reference	156
7.28.1	Macro Definition Documentation	157
	OFFLOAD_MYOF_FPTR_TABLE_SECTION_END	157
	OFFLOAD_MYOF_FPTR_TABLE_SECTION_START	157
	OFFLOAD_MYOF_SHARED_INIT_TABLE_SECTION_END	157
	OFFLOAD_MYOF_SHARED_INIT_TABLE_SECTION_START	157
	OFFLOAD_MYOF_SHARED_TABLE_SECTION_END	157
	OFFLOAD_MYOF_SHARED_TABLE_SECTION_START	157
7.28.2	Typedef Documentation	157
	SharedTableEntry	157
7.28.3	Function Documentation	158
	_offload_myofini	158
	_offload_myoregisterTables	158
7.29	offload_myof_target.cpp File Reference	158
7.29.1	Function Documentation	158
	_cilkrts_cilk_for_32	158
	_cilkrts_cilk_for_64	158
	_intel_cilk_for_32_offload_wrapper	158
	_intel_cilk_for_64_offload_wrapper	158
	_offload_myof_fptr_table_register	159
	_offload_myof_once_init	159
	_offload_myof_shared_table_register	159
	_offload_myofAcquire	159
	_offload_myofLibFini	159
	_offload_myofLibInit	159
	_offload_myofRegisterTables	159
	_offload_myofRelease	159
	Offload_shared_aligned_free	159
	Offload_shared_aligned_malloc	159
	Offload_shared_free	159

_offload_shared_malloc	159
CheckResult	159
7.30 offload.myo_target.h File Reference	160
7.30.1 Macro Definition Documentation	160
OFFLOAD_MYO_FPTR_TABLE_SECTION_END	160
OFFLOAD_MYO_FPTR_TABLE_SECTION_START	160
OFFLOAD_MYO_SHARED_TABLE_SECTION_END	160
OFFLOAD_MYO_SHARED_TABLE_SECTION_START	160
7.30.2 Typedef Documentation	160
FptrTableEntry	160
SharedTableEntry	160
7.30.3 Function Documentation	161
__offload_myoAcquire	161
__offload_myoLibFini	161
__offload_myoLibInit	161
__offload_myoRegisterTables	161
__offload_myoRelease	161
7.31 offload.omp_host.cpp File Reference	161
7.31.1 Function Documentation	162
kmp_create_affinity_mask_target	162
kmp_destroy_affinity_mask_target	162
kmp_get_affinity_mask_proc_target	162
kmp_get_affinity_max_proc_target	162
kmp_get_affinity_target	162
kmp_get_blocktime_target	162
kmp_get_library_target	163
kmp_get_stacksize_s_target	163
kmp_get_stacksize_target	163
kmp_set_affinity_mask_proc_target	163
kmp_set_affinity_target	163
kmp_set_blocktime_target	163
kmp_set_defaults_target	163
kmp_set_library_serial_target	163
kmp_set_library_target	163
kmp_set_library_throughput_target	163
kmp_set_library_turnaround_target	163
kmp_set_stacksize_s_target	163
kmp_set_stacksize_target	163
kmp_unset_affinity_mask_proc_target	163
omp_destroy_lock_target	164
omp_destroy_nest_lock_target	164
omp_get_default_device	164
omp_get_dynamic_target	164
omp_get_int_target	164
omp_get_max_threads_target	164
omp_get_nested_target	164
omp_get_num_devices	164
omp_get_num_procs_target	164
omp_get_schedule_target	164
omp_init_lock_target	164
omp_init_nest_lock_target	164
omp_set_default_device	164
omp_set_dynamic_target	165
omp_set_int_target	165
omp_set_lock_target	165
omp_set_nest_lock_target	165
omp_set_nested_target	165
omp_set_num_threads_target	165

omp_set_schedule_target	165
omp_test_lock_target	165
omp_test_nest_lock_target	165
omp_unset_lock_target	165
omp_unset_nest_lock_target	165
7.32 offload_omp_target.cpp File Reference	165
7.32.1 Function Documentation	167
kmp_create_affinity_mask_lrb	167
kmp_create_affinity_mask_target	167
kmp_destroy_affinity_mask_lrb	167
kmp_destroy_affinity_mask_target	168
kmp_get_affinity_lrb	168
kmp_get_affinity_mask_proc_lrb	168
kmp_get_affinity_mask_proc_target	168
kmp_get_affinity_max_proc_lrb	168
kmp_get_affinity_max_proc_target	168
kmp_get_affinity_target	168
kmp_get_blocktime_lrb	168
kmp_get_blocktime_target	168
kmp_get_library_lrb	168
kmp_get_library_target	168
kmp_get_stacksize_lrb	168
kmp_get_stacksize_s_lrb	168
kmp_get_stacksize_s_target	168
kmp_get_stacksize_target	169
kmp_set_affinity_lrb	169
kmp_set_affinity_mask_proc_lrb	169
kmp_set_affinity_mask_proc_target	169
kmp_set_affinity_target	169
kmp_set_blocktime_lrb	169
kmp_set_blocktime_target	169
kmp_set_defaults_lrb	169
kmp_set_defaults_target	169
kmp_set_library_lrb	169
kmp_set_library_serial_lrb	169
kmp_set_library_serial_target	169
kmp_set_library_target	169
kmp_set_library_throughput_lrb	169
kmp_set_library_throughput_target	169
kmp_set_library_turnaround_lrb	170
kmp_set_library_turnaround_target	170
kmp_set_stacksize_lrb	170
kmp_set_stacksize_s_lrb	170
kmp_set_stacksize_s_target	170
kmp_set_stacksize_target	170
kmp_unset_affinity_mask_proc_lrb	170
kmp_unset_affinity_mask_proc_target	170
omp_destroy_lock_lrb	170
omp_destroy_lock_target	170
omp_destroy_nest_lock_lrb	170
omp_destroy_nest_lock_target	170
omp_get_default_device	170
omp_get_dynamic_lrb	170
omp_get_dynamic_target	170
omp_get_int_from_host	171
omp_get_max_threads_lrb	171
omp_get_max_threads_target	171
omp_get_nested_lrb	171

omp_get_nested_target	171
omp_get_num_devices	171
omp_get_num_procs_lrb	171
omp_get_num_procs_target	171
omp_get_schedule_lrb	171
omp_get_schedule_target	171
omp_init_lock_lrb	171
omp_init_lock_target	171
omp_init_nest_lock_lrb	171
omp_init_nest_lock_target	171
omp_send_int_to_host	172
omp_set_default_device	172
omp_set_dynamic_lrb	172
omp_set_dynamic_target	172
omp_set_lock_lrb	172
omp_set_lock_target	172
omp_set_nest_lock_lrb	172
omp_set_nest_lock_target	172
omp_set_nested_lrb	172
omp_set_nested_target	172
omp_set_num_threads_lrb	172
omp_set_num_threads_target	172
omp_set_schedule_lrb	172
omp_set_schedule_target	172
omp_test_lock_lrb	173
omp_test_lock_target	173
omp_test_nest_lock_lrb	173
omp_test_nest_lock_target	173
omp_unset_lock_lrb	173
omp_unset_lock_target	173
omp_unset_nest_lock_lrb	173
omp_unset_nest_lock_target	173
7.33 offload.orsl.cpp File Reference	173
7.34 offload.orsl.h File Reference	174
7.35 offload.table.cpp File Reference	174
7.35.1 Function Documentation	175
__offload_register_tables	175
__offload_unregister_tables	175
kmp_create_affinity_mask_lrb	175
kmp_destroy_affinity_mask_lrb	175
kmp_get_affinity_lrb	175
kmp_get_affinity_mask_proc_lrb	175
kmp_get_affinity_max_proc_lrb	175
kmp_get_blocktime_lrb	175
kmp_get_library_lrb	175
kmp_get_stacksize_lrb	176
kmp_get_stacksize_s_lrb	176
kmp_set_affinity_lrb	176
kmp_set_affinity_mask_proc_lrb	176
kmp_set_blocktime_lrb	176
kmp_set_defaults_lrb	176
kmp_set_library_lrb	176
kmp_set_library_serial_lrb	176
kmp_set_library_throughput_lrb	176
kmp_set_library_turnaround_lrb	176
kmp_set_stacksize_lrb	176
kmp_set_stacksize_s_lrb	176
kmp_unset_affinity_mask_proc_lrb	176

omp_destroy_lock_lrb	176
omp_destroy_nest_lock_lrb	176
omp_get_dynamic_lrb	177
omp_get_max_threads_lrb	177
omp_get_nested_lrb	177
omp_get_num_procs_lrb	177
omp_get_schedule_lrb	177
omp_init_lock_lrb	177
omp_init_nest_lock_lrb	177
omp_set_dynamic_lrb	177
omp_set_lock_lrb	177
omp_set_nest_lock_lrb	177
omp_set_nested_lrb	177
omp_set_num_threads_lrb	177
omp_set_schedule_lrb	177
omp_test_lock_lrb	177
omp_test_nest_lock_lrb	177
omp_unset_lock_lrb	178
omp_unset_nest_lock_lrb	178
7.35.2 Variable Documentation	178
__offload_funcs	178
__offload_vars	178
predefined_entries	178
predefined_table	178
7.36 offload.table.h File Reference	178
7.36.1 Detailed Description	179
7.36.2 Macro Definition Documentation	179
OFFLOAD_ENTRY_TABLE_SECTION_END	179
OFFLOAD_ENTRY_TABLE_SECTION_START	179
OFFLOAD_FUNC_TABLE_SECTION_END	179
OFFLOAD_FUNC_TABLE_SECTION_START	179
OFFLOAD_VAR_TABLE_SECTION_END	179
OFFLOAD_VAR_TABLE_SECTION_START	179
7.36.3 Function Documentation	179
__offload_register_tables	179
__offload_unregister_tables	180
7.36.4 Variable Documentation	180
__offload_entries	180
__offload_funcs	180
__offload_vars	180
7.37 offload.target.cpp File Reference	180
7.37.1 Typedef Documentation	181
offload_func_with_parms	181
7.37.2 Function Documentation	181
__offload_target_init	181
__Offload_get_device_number	181
__Offload_get_physical_device_number	181
__Offload_number_of_devices	181
7.37.3 Variable Documentation	181
add_ref_lock	181
console.enabled	181
mic_engines.total	181
mic_frequency	181
mic_index	181
offload_number	181
offload_report_level	181
prefix	182
ref_data	182

	vardesc.direction_as_string	182
	vardesc.type_as_string	182
7.38	offload.target.h File Reference	182
7.38.1	Function Documentation	183
	__offload_target_init	183
7.38.2	Variable Documentation	183
	mic_engines_total	183
	mic_frequency	183
	mic_index	183
7.39	offload.target_main.cpp File Reference	183
7.39.1	Function Documentation	183
	__offload_target_main	183
	main	183
7.40	offload.timer.h File Reference	183
7.40.1	Macro Definition Documentation	184
	OFFLOAD_TIMER_DATALEN	184
	OFFLOAD_TIMER_INIT	184
	OFFLOAD_TIMER_START	184
	OFFLOAD_TIMER_STOP	184
	OFFLOAD_TIMER_TARGET_DATA	184
7.40.2	Variable Documentation	184
	timer_enabled	184
7.41	offload.timer_host.cpp File Reference	185
7.41.1	Variable Documentation	185
	timer_enabled	185
7.42	offload.timer_target.cpp File Reference	185
7.42.1	Variable Documentation	185
	timer_enabled	185
7.43	offload.trace.cpp File Reference	185
7.43.1	Function Documentation	186
	offload_signal	186
	offload_stage	186
	offload_stage_print	186
7.43.2	Variable Documentation	186
	mic_index	186
	prefix	186
7.44	offload.trace.h File Reference	186
7.44.1	Enumeration Type Documentation	187
	OffloadTraceStage	187
7.44.2	Function Documentation	187
	offload_stage_print	187
7.45	offload.util.cpp File Reference	188
7.45.1	Function Documentation	188
	__offload_parse_int_string	188
	__offload_parse_size_string	188
	DL_sym	188
	get_el_value	188
7.46	offload.util.h File Reference	188
7.46.1	Macro Definition Documentation	189
	__offload_run_once	189
	DL_addr	189
	DL_close	189
	DL_open	189
	OFFLOAD_ONCE_CONTROL_INIT	189
	thread_getspecific	189
	thread_key_create	189
	thread_key_delete	189
	thread_setspecific	189

7.46.2	Typedef Documentation	190
	OffloadOnceControl	190
7.46.3	Function Documentation	190
	__offload_parse_int_string	190
	__offload_parse_size_string	190
	DL_sym	190
	get_el_value	190
7.47	ofldbegin.cpp File Reference	190
7.47.1	Macro Definition Documentation	191
	ALLOCATE	191
	DLL_LOCAL	191
7.47.2	Function Documentation	191
	main	191
	MAIN__	191
	offload_fini	191
	offload_init	191
7.47.3	Variable Documentation	191
	__offload_entry_node	191
	__offload_entry_table_start	191
	__offload_func_node	191
	__offload_func_table_start	191
	__offload_var_node	192
	__offload_var_table_start	192
7.48	ofldend.cpp File Reference	192
7.48.1	Macro Definition Documentation	192
	ALLOCATE	192
7.48.2	Variable Documentation	192
	__offload_entry_table_end	192
	__offload_func_table_end	192
	__offload_var_table_end	192
7.49	orisl-lite/include/orisl-lite.h File Reference	192
7.49.1	Macro Definition Documentation	193
	ORSL_MAX_CARDS	193
	ORSL_MAX_TAG_LEN	193
7.49.2	Typedef Documentation	193
	BusySetType	193
	ORSLBusySet	193
	ORSLPartialGranularity	193
	ORSLTag	193
7.49.3	Enumeration Type Documentation	194
	ORSLBusySetType	194
	ORSLPartialGranularity	194
7.49.4	Function Documentation	194
	ORSLRelease	194
	ORSLReserve	195
	ORSLReservePartial	195
	ORSLTryReserve	196
7.50	orisl-lite/lib/orisl-lite.c File Reference	196
7.50.1	Macro Definition Documentation	197
	DISABLE_SYMBOL_VERSIONING	197
	ORSLRelease0	197
	ORSLReserve0	197
	ORSLReservePartial0	197
	ORSLTryReserve0	197
7.50.2	Function Documentation	197
	can_release_card	197
	can_reserve_card	197
	check_args	198

check_bsets	198
ORSLRelease0	198
ORSLReserve0	198
ORSLReservePartial0	198
ORSLTryReserve0	198
release_card	198
reserve_card	198
state_lock	198
state_signal_release	198
state_unlock	198
state_wait_for_release	199
7.50.3 Variable Documentation	199
owner	199
rsrv_cnt	199
rsrv_data	199
Index	200

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

COI	11
ORSL	14

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

_Offload.status	17
arr_desc	18
ArrDesc	18
AutoData	19
VarList::BufEntry	20
MicEnvVar::CardEnvVars	21
CeanReadDim	22
CeanReadRanges	22
dim_desc	23
DimDesc	24
Engine	25
FuncTable::Entry	31
VarTable::Entry	31
FptrTableEntry	32
FuncTable	34
FunctionDescriptor	35
Image	36
InitTableEntry	36
iterator	
VarList::Iterator	37
kmp_affinity_mask_target_t	38
mic_lib::kmp_create_affinity_mask_target	39
mic_lib::kmp_destroy_affinity_mask_target	39
mic_lib::kmp_get_affinity_mask_proc_target	39
mic_lib::kmp_get_affinity_max_proc_target	40
mic_lib::kmp_get_affinity_target	40
mic_lib::kmp_get_blocktime_target	40
mic_lib::kmp_get_library_target	41
mic_lib::kmp_get_stacksize_s_target	41
mic_lib::kmp_get_stacksize_target	41
mic_lib::kmp_set_affinity_mask_proc_target	42
mic_lib::kmp_set_affinity_target	42
mic_lib::kmp_set_blocktime_target	42
mic_lib::kmp_set_defaults_target	43
mic_lib::kmp_set_library_serial_target	43
mic_lib::kmp_set_library_target	43
mic_lib::kmp_set_library_throughput_target	44
mic_lib::kmp_set_library_turnaround_target	44
mic_lib::kmp_set_stacksize_s_target	44
mic_lib::kmp_set_stacksize_target	45

mic_lib::kmp_unset_affinity_mask_proc_target	45
Marshaller	45
MemRange	47
mic_lib	49
MicEnvVar	50
mutex_locker_t	52
mutex_t	52
MyoTable	54
MyoWrapper	54
TableList< T >::Node	58
mic_lib::offload_get_device_number	58
mic_lib::offload_get_physical_device_number	59
mic_lib::offload_number_of_devices	59
mic_lib::offload_report	59
mic_lib::offload_signaled	60
mic_lib::offload_status	60
OffloadDescriptor	61
mic_lib::omp_destroy_lock_target	68
mic_lib::omp_destroy_nest_lock_target	69
mic_lib::omp_get_dynamic_target	69
mic_lib::omp_get_max_threads_target	69
mic_lib::omp_get_nested_target	70
mic_lib::omp_get_num_procs_target	70
mic_lib::omp_get_schedule_target	70
mic_lib::omp_init_lock_target	71
mic_lib::omp_init_nest_lock_target	71
omp_lock_target_t	71
omp_nest_lock_target_t	72
mic_lib::omp_set_dynamic_target	72
mic_lib::omp_set_lock_target	73
mic_lib::omp_set_nest_lock_target	73
mic_lib::omp_set_nested_target	73
mic_lib::omp_set_num_threads_target	74
mic_lib::omp_set_schedule_target	74
mic_lib::omp_test_lock_target	74
mic_lib::omp_test_nest_lock_target	75
mic_lib::omp_unset_lock_target	75
mic_lib::omp_unset_nest_lock_target	75
ORSLBusySet	76
PersistData	76
PtrData	77
OffloadDescriptor::ReadArrElements< T >	79
RefInfo	81
TableList< T >	81
TableList< FuncTable >	81
FuncList	33
TableList< VarTable >	81
VarList	96
TargetImage	82
Thread	84
VarDesc	85
VarDesc2	92
VarDesc3	92
OffloadDescriptor::VarExtra	95
VarTable	97
MicEnvVar::VarValue	98

Chapter 3

Class Index

3.1 Class List

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

_Offload_status	17
arr_desc	18
ArrDesc	18
AutoData	19
VarList::BufEntry	20
MicEnvVar::CardEnvVars	21
CeanReadDim	22
CeanReadRanges	22
dim_desc	23
DimDesc	24
Engine	25
FuncTable::Entry	
Function table entry	31
VarTable::Entry	
Variable table entry	31
FptrTableEntry	32
FuncList	33
FuncTable	34
FunctionDescriptor	35
Image	
The target image is packed as follows	36
InitTableEntry	36
VarList::Iterator	37
kmp_affinity_mask_target_t	38
mic_lib::kmp_create_affinity_mask_target	39
mic_lib::kmp_destroy_affinity_mask_target	39
mic_lib::kmp_get_affinity_mask_proc_target	39
mic_lib::kmp_get_affinity_max_proc_target	40
mic_lib::kmp_get_affinity_target	40
mic_lib::kmp_get_blocktime_target	40
mic_lib::kmp_get_library_target	41
mic_lib::kmp_get_stacksize_s_target	41
mic_lib::kmp_get_stacksize_target	41
mic_lib::kmp_set_affinity_mask_proc_target	42
mic_lib::kmp_set_affinity_target	42
mic_lib::kmp_set_blocktime_target	42
mic_lib::kmp_set_defaults_target	43
mic_lib::kmp_set_library_serial_target	43
mic_lib::kmp_set_library_target	43
mic_lib::kmp_set_library_throughput_target	44

<code>mic_lib::kmp_set_library_turnaround_target</code>	44
<code>mic_lib::kmp_set_stacksize_s_target</code>	44
<code>mic_lib::kmp_set_stacksize_target</code>	45
<code>mic_lib::kmp_unset_affinity_mask_proc_target</code>	45
<code>Marshaller</code>	45
<code>MemRange</code>	47
<code>mic_lib</code>	49
<code>MicEnvVar</code>	50
<code>mutex_locker_t</code>	52
<code>mutex_t</code>	52
<code>MyoTable</code>	54
<code>MyoWrapper</code>	54
<code>TableList< T >::Node</code>	58
<code>mic_lib::offload_get_device_number</code>	58
<code>mic_lib::offload_get_physical_device_number</code>	59
<code>mic_lib::offload_number_of_devices</code>	59
<code>mic_lib::offload_report</code>	59
<code>mic_lib::offload_signaled</code>	60
<code>mic_lib::offload_status</code>	60
<code>OffloadDescriptor</code>	61
<code>mic_lib::omp_destroy_lock_target</code>	68
<code>mic_lib::omp_destroy_nest_lock_target</code>	69
<code>mic_lib::omp_get_dynamic_target</code>	69
<code>mic_lib::omp_get_max_threads_target</code>	69
<code>mic_lib::omp_get_nested_target</code>	70
<code>mic_lib::omp_get_num_procs_target</code>	70
<code>mic_lib::omp_get_schedule_target</code>	70
<code>mic_lib::omp_init_lock_target</code>	71
<code>mic_lib::omp_init_nest_lock_target</code>	71
<code>omp_lock_target_t</code>	71
<code>omp_nest_lock_target_t</code>	72
<code>mic_lib::omp_set_dynamic_target</code>	72
<code>mic_lib::omp_set_lock_target</code>	73
<code>mic_lib::omp_set_nest_lock_target</code>	73
<code>mic_lib::omp_set_nested_target</code>	73
<code>mic_lib::omp_set_num_threads_target</code>	74
<code>mic_lib::omp_set_schedule_target</code>	74
<code>mic_lib::omp_test_lock_target</code>	74
<code>mic_lib::omp_test_nest_lock_target</code>	75
<code>mic_lib::omp_unset_lock_target</code>	75
<code>mic_lib::omp_unset_nest_lock_target</code>	75
<code>ORSLBusySet</code>	76
<code>PersistData</code>	76
<code>PtrData</code>	77
<code>OffloadDescriptor::ReadArrElements< T ></code>	79
<code>RefInfo</code>	81
<code>TableList< T ></code>	81
<code>TargetImage</code>	82
<code>Thread</code>	84
<code>VarDesc</code>	
An Offload Variable descriptor	85
<code>VarDesc2</code>	
Auxiliary struct used when -g is enabled that holds variable names	92
<code>VarDesc3</code>	92
<code>OffloadDescriptor::VarExtra</code>	95
<code>VarList</code>	96
<code>VarTable</code>	97
<code>MicEnvVar::VarValue</code>	98

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

cean_util.cpp	99
cean_util.h	100
compiler_if_host.cpp	106
compiler_if_host.h	
The interface between compiler-generated host code and runtime library	107
compiler_if_target.cpp	110
compiler_if_target.h	
The interface between compiler-generated target code and runtime library	110
dv_util.cpp	112
dv_util.h	112
liboffload_error.c	114
liboffload_error_codes.h	115
liboffload_msg.c	121
liboffload_msg.h	122
mic_lib.f90	127
offload.h	127
offload_common.cpp	134
offload_common.h	
The parts of the runtime library common to host and target	134
offload_engine.cpp	140
offload_engine.h	140
offload_env.cpp	141
offload_env.h	142
offload_host.cpp	142
offload_host.h	
The parts of the runtime library used only on the host	149
offload_myo_host.cpp	153
offload_myo_host.h	156
offload_myo_target.cpp	158
offload_myo_target.h	160
offload_omp_host.cpp	161
offload_omp_target.cpp	165
offload_orisl.cpp	173
offload_orisl.h	174
offload_table.cpp	174
offload_table.h	
Function and Variable tables used by the runtime library	178
offload_target.cpp	180
offload_target.h	182
offload_target_main.cpp	183

offload_timer.h	183
offload_timer_host.cpp	185
offload_timer_target.cpp	185
offload_trace.cpp	185
offload_trace.h	186
offload_util.cpp	188
offload_util.h	188
ofldbbegin.cpp	190
ofldend.cpp	192
coi/coi_client.cpp	102
coi/coi_client.h	103
coi/coi_server.cpp	104
coi/coi_server.h	104
orisl-lite/include/orisl-lite.h	192
orisl-lite/lib/orisl-lite.c	196

Chapter 5

Namespace Documentation

5.1 COI Namespace Reference

Functions

- bool [init](#) (void)
- void [fini](#) (void)

Variables

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

- `COIRESET(* BufferSetState)(COIBUFFER, COIPROCESS, COI_BUFFER_STATE, COI_BUFFER_MOVE-
_FLAG, uint32_t, const COIEVENT *, COIEVENT *)`
- `COIRESET(* 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

COIRESET(* 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()`.

COIRESET(* 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()`.

COIRESET(* 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()`.

COIRESET(* COI::BufferDestroy)(COIBUFFER)

Definition at line 55 of file `coi_client.cpp`.

Referenced by `init()`, `Engine::init_ptr_data()`, and `OffloadDescriptor::offload_finish()`.

COIRESET(* COI::BufferGetSinkAddress)(COIBUFFER, uint64_t *)

Definition at line 65 of file `coi_client.cpp`.

Referenced by `init()`, and `OffloadDescriptor::init_mic_address()`.

COIRESET(* 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_myonit_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_myoiRemoteThunkCall()`, and `OffloadDescriptor::cleanup()`.

bool ORSL::reserve (int *device*)

Definition at line 43 of file offload_orsl.cpp.

Referenced by `__offload_myolsAvailable()`, `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_envron_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_envron_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::index

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_myonit_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_myonit_once(), __offload_myolsAvailable(), __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()`.

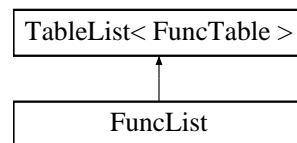
The documentation for this struct was generated from the following file:

- [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()`.

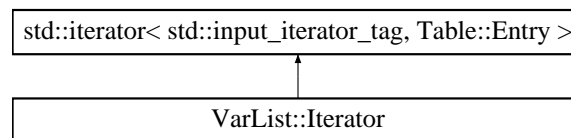
The documentation for this struct was generated from the following file:

- [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_envron_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_envron_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_envron_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_envron_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_envron_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_myoIRemoteThunkCall()`.

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_myoInitOnce()`.

bool MyoWrapper::LoadLibrary (void)

Definition at line 162 of file offload_myo_host.cpp.

Referenced by `_offload_myoLoadLibraryOnce()`.

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_myoRemoteThunkCall()`.

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_myoRemoteThunkCall()`.

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_signed Interface Reference

Public Member Functions

- integer(kind=c_int) function [offload_signed](#) (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_signed::offload_signed (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_omp, 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, COIRERESULT res)
- [_Offload_result](#) [translate_coi_error](#) (COIRERESULT 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*, COIRERESULT *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 (COIRERESULT 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_dataalen [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_omp [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_dataalen [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 <orl-lite/include/orl-lite.h>
```

Public Attributes

- [BusySetType](#) type

6.79.1 Detailed Description

[ORSLBusySet](#) encapsulation

Definition at line 32 of file orl-lite.h.

6.79.2 Member Data Documentation

BusySetType ORSLBusySet::type

Set type

Definition at line 33 of file orl-lite.h.

Referenced by [ORSL::release\(\)](#), [ORSL::reserve\(\)](#), and [ORSL::try_reserve\(\)](#).

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

- orl-lite/include/[orl-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

- [CleanReadRanges](#) * [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**
};
- void * **into**
This field not used by OpenMP 4.0.
- void * **ptr**
This field not used by OpenMP 4.0.
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.clean_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_celan_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](#)
- [CleanReadRanges](#) * [read_rng_src](#)
- [CleanReadRanges](#) * [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().

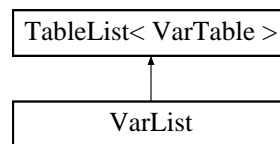
The documentation for this struct was generated from the following file:

- [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_envron_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_envron_for_card()`, `VarValue()`, and `~VarValue()`.

int MicEnvVar::VarValue::length

Definition at line 54 of file offload_env.h.

Referenced by `MicEnvVar::create_envron_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 "offloadhost.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, Var-`
`Desc2 *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, Var-`
`Desc2 *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

<i>o</i>	Offload descriptor created by OFFLOAD_TARGET_ACQUIRE .
<i>name</i>	Name of offload entry point.
<i>is_empty</i>	If no code to execute (e.g. offload.transfer)
<i>num_vars</i>	Number of variable descriptors.
<i>vars</i>	Pointer to VarDesc array.
<i>vars2</i>	Pointer to VarDesc2 array.
<i>num_waits</i>	Number of "wait" values.
<i>waits</i>	Pointer to array of wait values.
<i>signal</i>	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

<i>o</i>	Offload descriptor created by OFFLOAD_TARGET_ACQUIRE .
<i>name</i>	Name of offload entry point.
<i>is_empty</i>	If no code to execute (e.g. offload.transfer)
<i>num_vars</i>	Number of variable descriptors.
<i>vars</i>	Pointer to VarDesc array.

<i>vars2</i>	Pointer to VarDesc2 array.
<i>num_waits</i>	Number of "wait" values.
<i>waits</i>	Pointer to array of wait values.
<i>signal</i>	Pointer to signal value or NULL.
<i>entry_id</i>	A signature for the function doing the offload.
<i>stack_addr</i>	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

<i>target.type</i>	The type of target.
<i>target.number</i>	The device number.
<i>is_optional</i>	Whether CPU fall-back is allowed.
<i>status</i>	Address of variable to hold offload status.
<i>file</i>	Filename in which this offload occurred.
<i>line</i>	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

<i>device.number</i>	Device number or null if not specified.
<i>file</i>	Filename in which this offload occurred
<i>line</i>	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

<i>ofld</i>	Offload descriptor created by runtime.
<i>var_desc_num</i>	Number of variable descriptors.
<i>var_desc</i>	Pointer to VarDesc array.
<i>var_desc2</i>	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

<i>ofld</i>	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)
- [CleanReadRanges](#) * [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\(\)](#).

[CleanReadRanges](#)* [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)`
- `CleanReadRanges * 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_myofinit_once()`, `_offload_myofis-Available()`, `_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 == COIL_PROCESS_DIED) { \
 fini_process(true); \
 exit(1); \
 } \
 if (res != COIL_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_myLoadLibrary_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 = 0xffffffffffffULL**

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\_myolsAvailable(), 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\_myolsAvailable(), \_\_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\_myLoadLibrary\_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\_myolsAvailable(), 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.myoRemoteThunkCall` (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_myo_init()`, `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\_myoFinis ( void )**

Definition at line 474 of file offload\_myo\_host.cpp.  
Referenced by \_\_offload\_unregister\_image().

**static bool \_\_offload\_myolnit ( void ) [static]**

Definition at line 382 of file offload\_myo\_host.cpp.  
Referenced by \_\_offload\_myolsAvailable().

**static void \_\_offload\_myolnit\_once ( void ) [static]**

Definition at line 339 of file offload\_myo\_host.cpp.  
Referenced by \_\_offload\_myolnit().

**void \_\_offload\_myoRemoteThunkCall ( void \* *thunk*, void \* *arg*, int *target\_number* )**

Definition at line 638 of file offload\_myo\_host.cpp.

**int \_\_offload\_myolsAvailable ( 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\_myolnit\_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\_my\_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\\_my\\_register\\_tables](#) ([SharedTableEntry](#) \*shared\_table, [FptrTableEntry](#) \*fptr\_table)
- void [\\_\\_offload\\_my\\_acquire](#) (void)
- void [\\_\\_offload\\_my\\_release](#) (void)
- void [\\_\\_offload\\_my\\_libinit](#) ()
- void [\\_\\_offload\\_my\\_libfini](#) ()

#### 7.30.1 Macro Definition Documentation

**#define OFFLOAD\_MYO\_FPTR\_TABLE\_SECTION\_END ".MyoFptrTable."**

Definition at line 33 of file offload\_my\_target.h.

**#define OFFLOAD\_MYO\_FPTR\_TABLE\_SECTION\_START ".MyoFptrTable."**

Definition at line 32 of file offload\_my\_target.h.

**#define OFFLOAD\_MYO\_SHARED\_TABLE\_SECTION\_END ".MyoSharedTable."**

Definition at line 30 of file offload\_my\_target.h.

**#define OFFLOAD\_MYO\_SHARED\_TABLE\_SECTION\_START ".MyoSharedTable."**

Definition at line 29 of file offload\_my\_target.h.

#### 7.30.2 Typedef Documentation

**typedef MyoiTargetSharedFptrEntry FptrTableEntry**

Definition at line 20 of file offload\_my\_target.h.

**typedef MyoiSharedVarEntry SharedTableEntry**

Definition at line 19 of file offload\_my\_target.h.

### 7.30.3 Function Documentation

**void \_\_offload\_myAcquire ( void )**

Definition at line 77 of file offload\_my\_target.cpp.

**void \_\_offload\_myLibFini ( )**

Definition at line 180 of file offload\_my\_target.cpp.

**void \_\_offload\_myLibInit ( )**

Definition at line 174 of file offload\_my\_target.cpp.

**void \_\_offload\_myRegisterTables ( SharedTableEntry \* *shared\_table*, FptrTableEntry \* *fptr\_table* )**

Definition at line 129 of file offload\_my\_target.cpp.

**void \_\_offload\_myRelease ( void )**

Definition at line 83 of file offload\_my\_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 "orl-lite/include/orl-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\\_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_myolsAvailable()`, `__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 `COL::fini()`.

**#define `DL_open( path ) dlopen((path), RTLD_NOW)`**

Definition at line 121 of file `offload_util.h`.

Referenced by `COL::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 COL::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 ofldbbegin.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 ofldbbegin.cpp.

**#define DLL\_LOCAL \_\_attribute\_\_((visibility("hidden")))**

Definition at line 25 of file ofldbbegin.cpp.

### 7.47.2 Function Documentation

**int main ( void )**

Definition at line 37 of file ofldbbegin.cpp.

**int MAIN\_\_ ( void )**

Definition at line 44 of file ofldbbegin.cpp.

**static void offload\_fini ( ) [static]**

Definition at line 154 of file ofldbbegin.cpp.

Referenced by offload\_init().

**static void offload\_init ( ) [static]**

Definition at line 131 of file ofldbbegin.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 ofldbbegin.cpp.

**FuncTable::Entry \_\_offload\_entry\_table\_start = { 0 } [static]**

Definition at line 56 of file ofldbbegin.cpp.

**FuncList::Node \_\_offload\_func\_node [static]**

**Initial value:**

```
= {
 { &__offload_func_table_start + 1, -1 },
 0, 0
}
```

Definition at line 72 of file ofldbbegin.cpp.

**FuncTable::Entry \_\_offload\_func\_table\_start = { 0 } [static]**

Definition at line 69 of file ofldbbegin.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 "offloadmyo_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, const 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 `orl-lite.h`.  
 Referenced by `check_args()`.

#### `#define ORSL_MAX_TAG_LEN 128`

Maximal length of tag in characters  
 Definition at line 48 of file `orl-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 `orl-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

Granularity 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 | <i>n</i>     | Number of cards to reserve resources on. Cannot be < 0 or > ORSL_MAX_CARDS.                                                                                            |
| in | <i>inds</i>  | 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 | <i>bsets</i> | Requested resources on each of the card. Cannot be NULL if n > 0.                                                                                                      |
| in | <i>tag</i>   | 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 | <i>n</i>     | Number of cards to reserve resources on. Cannot be < 0 or > ORSL_MAX_CARDS.                                                                                                          |
| in | <i>inds</i>  | Indices of the cards: an integer array with <i>n</i> elements. Cannot be NULL if <i>n</i> > 0. Valid card indices are from 0 to ORSL_MAX_CARDS-1. Cannot contain duplicate elements. |
| in | <i>bsets</i> | Requested resources on each of the card. Cannot be NULL if <i>n</i> > 0.                                                                                                             |
| in | <i>tag</i>   | ORSLTag of the calling client. Cannot be NULL. Length must not exceed 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 | <i>gran</i>  | Reservation granularity                                                                                                                                                              |
| in | <i>n</i>     | Number of cards to reserve resources on. Cannot be < 0 or > ORSL_MAX_CARDS.                                                                                                          |
| in | <i>inds</i>  | Indices of the cards: an integer array with <i>n</i> elements. Cannot be NULL if <i>n</i> > 0. Valid card indices are from 0 to ORSL_MAX_CARDS-1. Cannot contain duplicate elements. |
| in | <i>bsets</i> | Requested resources on each of the card. Cannot be NULL if <i>n</i> > 0.                                                                                                             |
| in | <i>tag</i>   | 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      | <i>n</i>     | Number of cards to reserve resources on. Cannot be < 0 or > ORSL_MAX_CARDS.                                                                                                          |
| in      | <i>inds</i>  | Indices of the cards: an integer array with <i>n</i> elements. Cannot be NULL if <i>n</i> > 0. Valid card indices are from 0 to ORSL_MAX_CARDS-1. Cannot contain duplicate elements. |
| in, out | <i>bsets</i> | Requested resources on each of the card. Cannot be NULL if <i>n</i> > 0.                                                                                                             |
| in      | <i>tag</i>   | 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

- ~CardEnvVars
  - MicEnvVar::CardEnvVars, [21](#)
- ~Engine
  - Engine, [27](#)
- ~MicEnvVar
  - MicEnvVar, [51](#)
- ~OffloadDescriptor
  - OffloadDescriptor, [63](#)
- ~Thread
  - Thread, [84](#)
- ~VarValue
  - MicEnvVar::VarValue, [98](#)
- ~mutex\_locker\_t
  - mutex\_locker\_t, [52](#)
- ~mutex\_t
  - mutex\_t, [53](#)
- \_\_dummy\_\_
  - liboffload\_msg.h, [123](#)
- .\_Offload\_get\_device\_number
  - offload.h, [130](#)
  - offload\_host.cpp, [144](#)
  - offload\_target.cpp, [181](#)
- .\_Offload\_get\_physical\_device\_number
  - offload.h, [130](#)
  - offload\_host.cpp, [144](#)
  - offload\_target.cpp, [181](#)
- .\_Offload\_number\_of\_devices
  - offload.h, [130](#)
  - offload\_host.cpp, [145](#)
  - offload\_target.cpp, [181](#)
- .\_Offload\_report
  - offload.h, [130](#)
  - offload\_host.cpp, [145](#)
- .\_Offload\_result
  - offload.h, [130](#)
- .\_Offload\_shared\_aligned\_free
  - offload.h, [130](#)
  - offload\_myo\_host.cpp, [155](#)
  - offload\_myo\_target.cpp, [159](#)
- .\_Offload\_shared\_aligned\_malloc
  - offload.h, [130](#)
  - offload\_myo\_host.cpp, [155](#)
  - offload\_myo\_target.cpp, [159](#)
- .\_Offload\_shared\_free
  - offload.h, [130](#)
  - offload\_myo\_host.cpp, [156](#)
  - offload\_myo\_target.cpp, [159](#)
- .\_Offload\_shared\_malloc
  - offload.h, [130](#)
- offload\_myo\_host.cpp, [156](#)
- offload\_myo\_target.cpp, [159](#)
- .\_Offload\_signaled
  - offload.h, [131](#)
  - offload\_host.cpp, [145](#)
- .\_Offload\_status, [17](#)
  - data\_received, [17](#)
  - data\_sent, [17](#)
  - device\_number, [17](#)
  - result, [17](#)
- \_\_arr\_data\_offset\_and\_length
  - cean\_util.cpp, [99](#)
  - cean\_util.h, [101](#)
- \_\_arr\_desc\_dump
  - cean\_util.h, [101](#)
- \_\_arr\_desc\_length
  - cean\_util.h, [101](#)
- \_\_cilkrts\_cilk\_for\_32
  - offload\_myo\_host.cpp, [154](#)
  - offload\_myo\_target.cpp, [158](#)
- \_\_cilkrts\_cilk\_for\_64
  - offload\_myo\_host.cpp, [154](#)
  - offload\_myo\_target.cpp, [158](#)
- \_\_dbg\_api\_major\_version
  - offload\_host.cpp, [145](#)
  - offload\_host.h, [151](#)
- \_\_dbg\_api\_minor\_version
  - offload\_host.cpp, [145](#)
  - offload\_host.h, [151](#)
- \_\_dbg\_is\_attached
  - offload\_host.cpp, [145](#)
  - offload\_host.h, [151](#)
- \_\_dbg\_target\_exe\_name
  - offload\_host.cpp, [145](#)
  - offload\_host.h, [151](#)
- \_\_dbg\_target\_id
  - offload\_host.cpp, [145](#)
  - offload\_host.h, [151](#)
- \_\_dbg\_target\_so\_loaded
  - offload\_host.cpp, [144](#)
  - offload\_host.h, [151](#)
- \_\_dbg\_target\_so\_pid
  - offload\_host.cpp, [145](#)
  - offload\_host.h, [151](#)
- \_\_dbg\_target\_so\_unloaded
  - offload\_host.cpp, [144](#)
  - offload\_host.h, [151](#)
- \_\_dv\_data\_length
  - dv\_util.cpp, [112](#)

- dv\_util.h, 114
- \_\_dv\_desc\_dump
  - dv\_util.h, 113
- \_\_dv\_is\_allocated
  - dv\_util.cpp, 112
  - dv\_util.h, 114
- \_\_dv\_is\_contiguous
  - dv\_util.cpp, 112
  - dv\_util.h, 114
- \_\_intel\_cilk\_for\_32\_offload
  - offload\_myo\_host.cpp, 154
- \_\_intel\_cilk\_for\_32\_offload\_wrapper
  - offload\_myo\_target.cpp, 158
- \_\_intel\_cilk\_for\_64\_offload
  - offload\_myo\_host.cpp, 154
- \_\_intel\_cilk\_for\_64\_offload\_wrapper
  - offload\_myo\_target.cpp, 158
- \_\_liboffload\_error\_support
  - liboffload\_error.c, 114
  - liboffload\_error\_codes.h, 120
- \_\_liboffload\_report\_support
  - liboffload\_error\_codes.h, 120
- \_\_myo\_table\_list
  - offload\_myo\_host.cpp, 156
- \_\_myo\_table\_lock
  - offload\_myo\_host.cpp, 156
- \_\_myo\_tables
  - offload\_myo\_host.cpp, 156
- \_\_offload\_active\_wait
  - offload\_host.cpp, 146
- \_\_offload\_console\_trace
  - offload\_host.cpp, 144
- \_\_offload\_entries
  - offload\_table.h, 180
- \_\_offload\_entry\_node
  - ofldbegin.cpp, 191
- \_\_offload\_entry\_table\_end
  - ofldend.cpp, 192
- \_\_offload\_entry\_table\_start
  - ofldbegin.cpp, 191
- \_\_offload\_fini\_library
  - Engine, 29
  - offload\_host.cpp, 144
- \_\_offload\_func\_node
  - ofldbegin.cpp, 191
- \_\_offload\_func\_table\_end
  - ofldend.cpp, 192
- \_\_offload\_func\_table\_start
  - ofldbegin.cpp, 191
- \_\_offload\_funcs
  - offload\_table.cpp, 178
  - offload\_table.h, 180
- \_\_offload\_init\_library
  - offload\_host.cpp, 144
  - offload\_host.h, 151
- \_\_offload\_init\_library\_once
  - Engine, 29
  - offload\_host.cpp, 144
- \_\_offload\_init\_type
  - offload\_host.cpp, 146
  - offload\_host.h, 152
- \_\_offload\_myoAcquire
  - offload\_myo\_target.cpp, 159
  - offload\_myo\_target.h, 161
- \_\_offload\_myoFini
  - offload\_myo\_host.cpp, 155
  - offload\_myo\_host.h, 158
- \_\_offload\_myoInit
  - offload\_myo\_host.cpp, 155
- \_\_offload\_myoInit\_once
  - offload\_myo\_host.cpp, 155
- \_\_offload\_myolsAvailable
  - offload\_myo\_host.cpp, 155
- \_\_offload\_myoLibFini
  - offload\_myo\_target.cpp, 159
  - offload\_myo\_target.h, 161
- \_\_offload\_myoLibInit
  - offload\_myo\_target.cpp, 159
  - offload\_myo\_target.h, 161
- \_\_offload\_myoLoadLibrary
  - offload\_myo\_host.cpp, 155
- \_\_offload\_myoLoadLibrary\_once
  - offload\_myo\_host.cpp, 155
- \_\_offload\_myoRegisterTables
  - offload\_myo\_host.cpp, 155
  - offload\_myo\_host.h, 158
  - offload\_myo\_target.cpp, 159
  - offload\_myo\_target.h, 161
- \_\_offload\_myoRelease
  - offload\_myo\_target.cpp, 159
  - offload\_myo\_target.h, 161
- \_\_offload\_myo\_fptr\_table\_register
  - offload\_myo\_host.cpp, 154
  - offload\_myo\_target.cpp, 158
- \_\_offload\_myo\_once\_init
  - offload\_myo\_target.cpp, 159
- \_\_offload\_myo\_shared\_init\_table\_register
  - offload\_myo\_host.cpp, 155
- \_\_offload\_myo\_shared\_table\_register
  - offload\_myo\_host.cpp, 155
  - offload\_myo\_target.cpp, 159
- \_\_offload\_myoiRemoteThunkCall
  - offload\_myo\_host.cpp, 155
- \_\_offload\_parse\_int\_string
  - offload\_util.cpp, 188
  - offload\_util.h, 190
- \_\_offload\_parse\_size\_string
  - offload\_util.cpp, 188
  - offload\_util.h, 190
- \_\_offload\_register\_image
  - offload\_host.cpp, 144
  - offload\_host.h, 151
- \_\_offload\_register\_tables
  - offload\_table.cpp, 175
  - offload\_table.h, 179
- \_\_offload\_run\_once



- offload\_util.h, 189
- \_\_offload\_target\_init
  - offload\_target.cpp, 181
  - offload\_target.h, 183
- \_\_offload\_target\_main
  - offload\_target\_main.cpp, 183
- \_\_offload\_unregister\_image
  - offload\_host.cpp, 144
  - offload\_host.h, 151
- \_\_offload\_unregister\_tables
  - offload\_table.cpp, 175
  - offload\_table.h, 179
- \_\_offload\_use\_2mb\_buffers
  - offload\_host.cpp, 146
  - offload\_host.h, 152
- \_\_offload\_use\_async\_buffer\_read
  - offload\_host.cpp, 146
- \_\_offload\_use\_async\_buffer\_write
  - offload\_host.cpp, 146
- \_\_offload\_var\_node
  - ofldbegin.cpp, 191
- \_\_offload\_var\_table\_end
  - ofldend.cpp, 192
- \_\_offload\_var\_table\_start
  - ofldbegin.cpp, 192
- \_\_offload\_vars
  - offload\_table.cpp, 178
  - offload\_table.h, 180
- \_\_omp\_device\_num
  - offload\_host.cpp, 146
  - offload\_host.h, 152
- \_\_target\_exe
  - offload\_host.cpp, 146
  - offload\_host.h, 152
- \_\_target\_libs
  - offload\_host.cpp, 146
- \_\_target\_libs\_list
  - offload\_host.cpp, 146
- \_\_target\_libs\_lock
  - offload\_host.cpp, 146
- ALLOCATE
  - ofldbegin.cpp, 191
  - ofldend.cpp, 192
- Acquire
  - MyoWrapper, 55
- add\_env\_var
  - MicEnvVar, 51
- add\_lib
  - Engine, 27
- add\_new\_env\_var
  - MicEnvVar::CardEnvVars, 21
- add\_ref\_lock
  - offload\_target.cpp, 181
- add\_reference
  - AutoData, 20
  - PtrData, 78
- add\_signal
  - Engine, 27
- add\_table
  - FuncList, 33
  - TableList, 82
- addr
  - VarList::BufEntry, 21
  - VarTable::Entry, 32
- align
  - VarDesc, 86
- align\_array
  - VarDesc3, 93
- alloc
  - VarDesc, 86
- alloc\_disp
  - PtrData, 78
  - VarDesc, 86
- alloc\_elements
  - VarDesc3, 93
- alloc\_if
  - VarDesc, 87
- alloc\_if\_array
  - VarDesc3, 93
- alloc\_ptr\_data
  - OffloadDescriptor, 63
- alloc\_ptr\_data\_lock
  - PtrData, 78
- alloc\_start
  - VarDesc3, 93
- analyze\_env\_var
  - MicEnvVar, 51
- any\_card
  - MicEnvVar, 51
- arr\_desc, 18
  - base, 18
  - dim, 18
  - rank, 18
- ArrDesc, 18
  - Base, 19
  - Dim, 19
  - dv\_util.h, 113
  - Flags, 19
  - Len, 19
  - Offset, 19
  - Rank, 19
  - Reserved, 19
- ArrDescFlagsContiguous
  - dv\_util.h, 113
- ArrDescFlagsDefined
  - dv\_util.h, 113
- ArrDescFlagsNodealloc
  - dv\_util.h, 113
- ArrDescMaxArrayRank
  - dv\_util.h, 113
- array\_fields
  - VarDesc3, 93
- auto\_data
  - OffloadDescriptor::VarExtra, 95
- AutoData, 19
  - add\_reference, 20

- AutoData, 20
- AutoData, 20
- cpu\_addr, 20
- get\_reference, 20
- operator<, 20
- ref\_count, 20
- remove\_reference, 20
- AutoSet
  - offload\_engine.h, 141
- BUSY\_SET\_EMPTY
  - orisl-lite.h, 194
- BUSY\_SET\_FULL
  - orisl-lite.h, 194
- BUSY\_SET\_PARTIAL
  - orisl-lite.h, 194
- Base
  - ArrDesc, 19
- base
  - arr\_desc, 18
  - OffloadDescriptor::ReadArrElements, 80
- begin
  - VarList, 97
- bits
  - VarDesc, 87
- buffer\_ptr
  - Marshaller, 47
- buffer\_size
  - Marshaller, 47
- buffer\_start
  - Marshaller, 47
- BufferAddRef
  - coi\_server.h, 105
- BufferCopy
  - COI, 12
- BufferCreate
  - COI, 12
- BufferCreateFromMemory
  - COI, 12
- BufferDestroy
  - COI, 12
- BufferGetSinkAddress
  - COI, 12
- BufferList
  - OffloadDescriptor, 62
- BufferMap
  - COI, 12
- BufferRead
  - COI, 12
- BufferReleaseRef
  - coi\_server.h, 105
- BufferSetState
  - COI, 13
- BufferUnmap
  - COI, 13
- BufferWrite
  - COI, 13
- BusySetType
  - orisl-lite.h, 193
- c\_bad\_ptr\_mem\_range
  - liboffload\_error\_codes.h, 118
- c\_buf\_add\_ref
  - liboffload\_error\_codes.h, 118
- c\_buf\_copy
  - liboffload\_error\_codes.h, 118
- c\_buf\_create
  - liboffload\_error\_codes.h, 118
- c\_buf\_create\_from\_mem
  - liboffload\_error\_codes.h, 118
- c\_buf\_create\_out\_of\_mem
  - liboffload\_error\_codes.h, 118
- c\_buf\_destroy
  - liboffload\_error\_codes.h, 118
- c\_buf\_get\_address
  - liboffload\_error\_codes.h, 118
- c\_buf\_map
  - liboffload\_error\_codes.h, 118
- c\_buf\_read
  - liboffload\_error\_codes.h, 118
- c\_buf\_release\_ref
  - liboffload\_error\_codes.h, 118
- c\_buf\_set\_state
  - liboffload\_error\_codes.h, 118
- c\_buf\_unmap
  - liboffload\_error\_codes.h, 118
- c\_buf\_write
  - liboffload\_error\_codes.h, 118
- c\_cean\_var
  - offload\_common.h, 137
- c\_cean\_var\_ptr
  - offload\_common.h, 137
- c\_coipipe\_max\_number
  - liboffload\_error\_codes.h, 119
- c\_data
  - offload\_common.h, 137
- c\_data\_ptr
  - offload\_common.h, 137
- c\_data\_ptr\_array
  - offload\_common.h, 137
- c\_destination\_is\_over
  - liboffload\_error\_codes.h, 118
- c\_device\_is\_not\_available
  - liboffload\_error\_codes.h, 117
- c\_different\_src\_and\_dstn\_sizes
  - liboffload\_error\_codes.h, 118
- c\_dv
  - offload\_common.h, 137
- c\_dv\_data
  - offload\_common.h, 137
- c\_dv\_data\_slice
  - offload\_common.h, 137
- c\_dv\_ptr
  - offload\_common.h, 137
- c\_dv\_ptr\_data
  - offload\_common.h, 137
- c\_dv\_ptr\_data\_slice
  - offload\_common.h, 137

- c\_event\_wait
  - liboffload\_error\_codes.h, 118
- c\_func\_compute
  - Engine, 26
- c\_func\_init
  - Engine, 26
- c\_func\_ptr
  - offload\_common.h, 137
- c\_func\_ptr\_array
  - offload\_common.h, 138
- c\_func\_var\_table\_copy
  - Engine, 26
- c\_func\_var\_table\_size
  - Engine, 26
- c\_funcs\_total
  - Engine, 26
- c\_get\_engine\_handle
  - liboffload\_error\_codes.h, 117
- c\_get\_engine\_index
  - liboffload\_error\_codes.h, 117
- c\_init\_on\_offload
  - offload\_host.h, 151
- c\_init\_on\_offload\_all
  - offload\_host.h, 151
- c\_init\_on\_start
  - offload\_host.h, 151
- c\_invalid\_device\_number
  - liboffload\_error\_codes.h, 117
- c\_invalid\_env\_report\_value
  - liboffload\_error\_codes.h, 117
- c\_invalid\_env\_var\_int\_value
  - liboffload\_error\_codes.h, 117
- c\_invalid\_env\_var\_value
  - liboffload\_error\_codes.h, 117
- c\_load\_library
  - liboffload\_error\_codes.h, 117
- c\_merge\_var\_descs1
  - liboffload\_error\_codes.h, 117
- c\_merge\_var\_descs2
  - liboffload\_error\_codes.h, 117
- c\_mic\_card\_env
  - offload\_env.h, 142
- c\_mic\_card\_var
  - offload\_env.h, 142
- c\_mic\_init3
  - liboffload\_error\_codes.h, 117
- c\_mic\_init4
  - liboffload\_error\_codes.h, 117
- c\_mic\_init5
  - liboffload\_error\_codes.h, 117
- c\_mic\_init6
  - liboffload\_error\_codes.h, 117
- c\_mic\_parse\_env\_var\_list1
  - liboffload\_error\_codes.h, 117
- c\_mic\_parse\_env\_var\_list2
  - liboffload\_error\_codes.h, 117
- c\_mic\_process\_exit
  - liboffload\_error\_codes.h, 117
- c\_mic\_process\_exit\_ret
  - liboffload\_error\_codes.h, 117
- c\_mic\_process\_exit\_sig
  - liboffload\_error\_codes.h, 117
- c\_mic\_var
  - offload\_env.h, 142
- c\_multiple\_target\_exes
  - liboffload\_error\_codes.h, 118
- c\_myotarget\_checkresult
  - liboffload\_error\_codes.h, 117
- c\_myowrapper\_checkresult
  - liboffload\_error\_codes.h, 117
- c\_no\_mic
  - offload\_env.h, 142
- c\_no\_ptr\_data
  - liboffload\_error\_codes.h, 117
- c\_no\_static\_var\_data
  - liboffload\_error\_codes.h, 117
- c\_no\_target\_exe
  - liboffload\_error\_codes.h, 118
- c\_non\_contiguous\_dope\_vector
  - liboffload\_error\_codes.h, 118
- c\_offload1
  - liboffload\_error\_codes.h, 117
- c\_offload\_compute
  - offload\_trace.h, 187
- c\_offload\_copyin\_data
  - offload\_trace.h, 187
- c\_offload\_copyout\_data
  - offload\_trace.h, 187
- c\_offload\_create\_buf\_host
  - offload\_trace.h, 187
- c\_offload\_create\_buf\_mic
  - offload\_trace.h, 187
- c\_offload\_descriptor\_offload
  - liboffload\_error\_codes.h, 117
- c\_offload\_destroy
  - offload\_trace.h, 187
- c\_offload\_finish
  - offload\_trace.h, 187
- c\_offload\_gather\_copyin\_data
  - offload\_trace.h, 187
- c\_offload\_gather\_copyout\_data
  - offload\_trace.h, 187
- c\_offload\_host\_alloc\_buffers
  - liboffload\_error\_codes.h, 120
- c\_offload\_host\_alloc\_data\_buffer
  - liboffload\_error\_codes.h, 120
- c\_offload\_host\_destroy\_buffers
  - liboffload\_error\_codes.h, 120
- c\_offload\_host\_gather\_inputs
  - liboffload\_error\_codes.h, 120
- c\_offload\_host\_initialize
  - liboffload\_error\_codes.h, 120
- c\_offload\_host\_map\_in\_data\_buffer
  - liboffload\_error\_codes.h, 120
- c\_offload\_host\_map\_out\_data\_buffer
  - liboffload\_error\_codes.h, 120

- c\_offload\_host\_max\_phase  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_scatter\_outputs  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_send\_pointers  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_setup\_buffers  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_setup\_misc\_data  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_start\_buffers\_reads  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_start\_compute  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_target\_acquire  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_total\_offload  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_unmap\_in\_data\_buffer  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_unmap\_out\_data\_buffer  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_wait\_buffers\_reads  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_wait\_compute  
liboffload\_error\_codes.h, 120
- c\_offload\_host\_wait\_deps  
liboffload\_error\_codes.h, 120
- c\_offload\_init  
offload\_trace.h, 187
- c\_offload\_init\_func  
offload\_trace.h, 187
- c\_offload\_malloc  
liboffload\_error\_codes.h, 117
- c\_offload\_mic\_myofptr  
offload\_trace.h, 187
- c\_offload\_mic\_myoshared  
offload\_trace.h, 187
- c\_offload\_myoacquire  
offload\_trace.h, 187
- c\_offload\_myofini  
offload\_trace.h, 187
- c\_offload\_myoinit  
offload\_trace.h, 187
- c\_offload\_myoregister  
offload\_trace.h, 187
- c\_offload\_myorelease  
offload\_trace.h, 187
- c\_offload\_myosharedalignedfree  
offload\_trace.h, 187
- c\_offload\_myosharedalignedmalloc  
offload\_trace.h, 187
- c\_offload\_myosharedfree  
offload\_trace.h, 187
- c\_offload\_myosharedmalloc  
offload\_trace.h, 187
- c\_offload\_receive\_pointer\_data  
offload\_trace.h, 187
- c\_offload\_received\_pointer\_data  
offload\_trace.h, 187
- c\_offload\_register  
offload\_trace.h, 187
- c\_offload\_scatter\_copyin\_data  
offload\_trace.h, 187
- c\_offload\_scatter\_copyout\_data  
offload\_trace.h, 187
- c\_offload\_send\_pointer\_data  
offload\_trace.h, 187
- c\_offload\_sent\_pointer\_data  
offload\_trace.h, 187
- c\_offload\_signal  
offload\_trace.h, 187
- c\_offload\_signaled1  
liboffload\_error\_codes.h, 117
- c\_offload\_signaled2  
liboffload\_error\_codes.h, 117
- c\_offload\_start  
offload\_trace.h, 187
- c\_offload\_start\_target\_func  
offload\_trace.h, 187
- c\_offload\_target\_add\_buffer\_refs  
liboffload\_error\_codes.h, 120
- c\_offload\_target\_compute  
liboffload\_error\_codes.h, 120
- c\_offload\_target\_descriptor\_setup  
liboffload\_error\_codes.h, 120
- c\_offload\_target\_func\_lookup  
liboffload\_error\_codes.h, 120
- c\_offload\_target\_func\_time  
liboffload\_error\_codes.h, 120
- c\_offload\_target\_gather\_outputs  
liboffload\_error\_codes.h, 120
- c\_offload\_target\_max\_phase  
liboffload\_error\_codes.h, 120
- c\_offload\_target\_release\_buffer\_refs  
liboffload\_error\_codes.h, 120
- c\_offload\_target\_scatter\_inputs  
liboffload\_error\_codes.h, 120
- c\_offload\_target\_total\_time  
liboffload\_error\_codes.h, 120
- c\_offload\_unregister  
offload\_trace.h, 187
- c\_offload\_var  
offload\_trace.h, 187
- c\_offload\_wait  
offload\_trace.h, 187
- c\_omp\_invalid\_device\_num  
liboffload\_error\_codes.h, 118
- c\_omp\_invalid\_device\_num\_env  
liboffload\_error\_codes.h, 118
- c\_parameter\_in  
offload\_common.h, 138
- c\_parameter\_inout  
offload\_common.h, 138
- c\_parameter\_nocopy  
offload\_common.h, 138

- c.parameter\_out
  - offload.common.h, 138
- c.parameter\_unknown
  - offload.common.h, 138
- c.pipeline\_create
  - liboffload\_error\_codes.h, 117
- c.pipeline\_run\_func
  - liboffload\_error\_codes.h, 118
- c.pipeline\_start\_run\_funcs
  - liboffload\_error\_codes.h, 118
- c.pointer\_array\_mismatch
  - liboffload\_error\_codes.h, 118
- c.process\_create
  - liboffload\_error\_codes.h, 117
- c.process\_get\_func\_handles
  - liboffload\_error\_codes.h, 117
- c.process\_proxy\_flush
  - liboffload\_error\_codes.h, 117
- c.process\_wait\_shutdown
  - liboffload\_error\_codes.h, 117
- c.ranges\_dont\_match
  - liboffload\_error\_codes.h, 118
- c.receive\_func\_ptr
  - liboffload\_error\_codes.h, 117
- c.report\_bytes
  - liboffload\_error\_codes.h, 118
- c.report\_compute
  - liboffload\_error\_codes.h, 119
- c.report\_copyin\_data
  - liboffload\_error\_codes.h, 119
- c.report\_copyout\_data
  - liboffload\_error\_codes.h, 119
- c.report\_cpu\_time
  - liboffload\_error\_codes.h, 118
- c.report\_cpu\_to\_mic\_data
  - liboffload\_error\_codes.h, 118
- c.report\_create\_buf\_host
  - liboffload\_error\_codes.h, 119
- c.report\_create\_buf\_mic
  - liboffload\_error\_codes.h, 119
- c.report\_destroy
  - liboffload\_error\_codes.h, 119
- c.report\_file
  - liboffload\_error\_codes.h, 118
- c.report\_from\_file
  - liboffload\_error\_codes.h, 118
- c.report\_gather\_copyin\_data
  - liboffload\_error\_codes.h, 119
- c.report\_gather\_copyout\_data
  - liboffload\_error\_codes.h, 119
- c.report\_host
  - liboffload\_error\_codes.h, 118
- c.report\_init
  - liboffload\_error\_codes.h, 119
- c.report\_init\_func
  - liboffload\_error\_codes.h, 119
- c.report\_line
  - liboffload\_error\_codes.h, 118
- c.report\_logical\_card
  - liboffload\_error\_codes.h, 119
- c.report\_mic
  - liboffload\_error\_codes.h, 118
- c.report\_mic\_myo\_fptr
  - liboffload\_error\_codes.h, 119
- c.report\_mic\_myo\_shared
  - liboffload\_error\_codes.h, 119
- c.report\_mic\_time
  - liboffload\_error\_codes.h, 118
- c.report\_mic\_to\_cpu\_data
  - liboffload\_error\_codes.h, 118
- c.report\_myoacquire
  - liboffload\_error\_codes.h, 119
- c.report\_myofini
  - liboffload\_error\_codes.h, 119
- c.report\_myoinit
  - liboffload\_error\_codes.h, 119
- c.report\_myoregister
  - liboffload\_error\_codes.h, 119
- c.report\_myorelease
  - liboffload\_error\_codes.h, 119
- c.report\_myosharedalignedfree
  - liboffload\_error\_codes.h, 119
- c.report\_myosharedalignedmalloc
  - liboffload\_error\_codes.h, 119
- c.report\_myosharedfree
  - liboffload\_error\_codes.h, 119
- c.report\_myosharedmalloc
  - liboffload\_error\_codes.h, 119
- c.report\_offload
  - liboffload\_error\_codes.h, 119
- c.report\_physical\_card
  - liboffload\_error\_codes.h, 119
- c.report\_receive\_pointer\_data
  - liboffload\_error\_codes.h, 119
- c.report\_received\_pointer\_data
  - liboffload\_error\_codes.h, 119
- c.report\_register
  - liboffload\_error\_codes.h, 119
- c.report\_scatter\_copyin\_data
  - liboffload\_error\_codes.h, 119
- c.report\_scatter\_copyout\_data
  - liboffload\_error\_codes.h, 119
- c.report\_seconds
  - liboffload\_error\_codes.h, 118
- c.report\_send\_pointer\_data
  - liboffload\_error\_codes.h, 119
- c.report\_sent\_pointer\_data
  - liboffload\_error\_codes.h, 119
- c.report\_signal
  - liboffload\_error\_codes.h, 119
- c.report\_start
  - liboffload\_error\_codes.h, 119
- c.report\_start\_target\_func
  - liboffload\_error\_codes.h, 119
- c.report\_state
  - liboffload\_error\_codes.h, 119

- c\_report\_state\_signal
  - liboffload\_error\_codes.h, 119
- c\_report\_tag
  - liboffload\_error\_codes.h, 118
- c\_report\_target
  - liboffload\_error\_codes.h, 118
- c\_report\_title
  - liboffload\_error\_codes.h, 118
- c\_report\_unknown\_timer\_node
  - liboffload\_error\_codes.h, 118
- c\_report\_unknown\_trace\_node
  - liboffload\_error\_codes.h, 119
- c\_report\_unregister
  - liboffload\_error\_codes.h, 119
- c\_report\_var
  - liboffload\_error\_codes.h, 119
- c\_report\_w\_tag
  - liboffload\_error\_codes.h, 119
- c\_report\_wait
  - liboffload\_error\_codes.h, 119
- c\_send\_func\_ptr
  - liboffload\_error\_codes.h, 117
- c\_slice\_of\_noncont\_array
  - liboffload\_error\_codes.h, 118
- c\_string\_ptr
  - offload\_common.h, 137
- c\_string\_ptr\_array
  - offload\_common.h, 138
- c\_unknown\_binary\_type
  - liboffload\_error\_codes.h, 118
- c\_unknown\_var\_type
  - liboffload\_error\_codes.h, 117
- c\_void\_ptr
  - offload\_common.h, 137
- c\_void\_ptr\_array
  - offload\_common.h, 138
- c\_zero\_or\_neg\_ptr\_len
  - liboffload\_error\_codes.h, 118
- c\_zero\_or\_neg\_transfer\_size
  - liboffload\_error\_codes.h, 118
- c\_signal\_max
  - Engine, 29
- c\_signal\_names
  - Engine, 29
- COI, 11
  - BufferCopy, 12
  - BufferCreate, 12
  - BufferCreateFromMemory, 12
  - BufferDestroy, 12
  - BufferGetSinkAddress, 12
  - BufferMap, 12
  - BufferRead, 12
  - BufferSetState, 13
  - BufferUnmap, 13
  - BufferWrite, 13
  - EngineGetCount, 13
  - EngineGetHandle, 13
  - EventWait, 13
  - fini, 12
  - init, 12
  - is\_available, 13
  - lib\_handle, 13
  - PerfGetCycleFrequency, 13
  - PipelineCreate, 13
  - PipelineDestroy, 14
  - PipelineRunFunction, 14
  - ProcessCreateFromMemory, 14
  - ProcessDestroy, 14
  - ProcessGetFunctionHandles, 14
  - ProcessLoadLibraryFromMemory, 14
  - ProcessRegisterLibraries, 14
- COI\_VERSION1
  - coi\_client.cpp, 103
- COI\_VERSION2
  - coi\_client.cpp, 103
- can\_release\_card
  - orisl-lite.c, 197
- can\_reserve\_card
  - orisl-lite.c, 197
- card\_number
  - MicEnvVar::CardEnvVars, 22
- card\_spec\_list
  - MicEnvVar, 51
- CardEnvVars
  - MicEnvVar::CardEnvVars, 21
- cean\_get\_transf\_size
  - cean\_util.cpp, 99
  - cean\_util.h, 101
- cean\_ranges\_match
  - cean\_util.cpp, 100
  - cean\_util.h, 101
- cean\_util.cpp, 99
  - \_\_arr\_data\_offset\_and\_length, 99
  - cean\_get\_transf\_size, 99
  - cean\_ranges\_match, 100
  - fpp, 99
  - generate\_mem\_ranges, 100
  - generate\_mem\_ranges\_one\_rank, 100
  - generate\_one\_range, 100
  - get\_next\_range, 100
  - init\_read\_ranges\_arr\_desc, 100
  - is\_arr\_desc\_contiguous, 100
  - last\_left, 100
  - last\_right, 100
- cean\_util.h, 100
  - \_\_arr\_data\_offset\_and\_length, 101
  - \_\_arr\_desc\_dump, 101
  - \_\_arr\_desc\_length, 101
  - cean\_get\_transf\_size, 101
  - cean\_ranges\_match, 101
  - get\_next\_range, 101
  - init\_read\_ranges\_arr\_desc, 101
  - is\_arr\_desc\_contiguous, 101
- CeanReadDim, 22
  - count, 22
  - size, 22

- CeanReadRanges, 22
  - current\_number, 23
  - Dim, 23
  - init\_offset, 23
  - last\_noncont\_ind, 23
  - ptr, 23
  - range\_max\_number, 23
  - range\_size, 23
- check\_args
  - orisl-lite.c, 197
- check\_bsets
  - orisl-lite.c, 198
- check\_result
  - offload\_engine.h, 141
- CheckResult
  - MyoWrapper, 55
  - offload\_myio\_target.cpp, 159
- cleanup
  - OffloadDescriptor, 63
- coi/coi\_client.cpp, 102
- coi/coi\_client.h, 103
- coi/coi\_server.cpp, 104
- coi/coi\_server.h, 104
- coi\_client.cpp
  - COI\_VERSION1, 103
  - COI\_VERSION2, 103
- coi\_client.h
  - MIC\_ENGINES\_MAX, 103
- coi\_server.cpp
  - server\_compute, 104
  - server\_init, 104
  - server\_var\_table\_copy, 104
  - server\_var\_table\_size, 104
- coi\_server.h
  - BufferAddRef, 105
  - BufferReleaseRef, 105
  - EngineGetIndex, 105
  - PipelineStartExecutingRunFunctions, 105
  - ProcessWaitForShutdown, 105
- common\_vars
  - MicEnvVar, 51
- compiler\_if\_host.cpp, 106
  - OFFLOAD\_OFFLOAD, 106
  - OFFLOAD\_OFFLOAD1, 106
  - OFFLOAD\_OFFLOAD2, 106
  - offload\_call\_count, 107
  - offload\_offload\_wrap, 107
- compiler\_if\_host.h, 107
  - OFFLOAD\_OFFLOAD, 108, 109
  - OFFLOAD\_OFFLOAD1, 108, 109
  - OFFLOAD\_OFFLOAD2, 108, 109
- compiler\_if\_target.cpp, 110
- compiler\_if\_target.h, 110
- compute
  - Engine, 27
  - OffloadDescriptor, 63
- console\_enabled
  - FunctionDescriptor, 35
- offload\_common.h, 138
- offload\_host.cpp, 146
- offload\_target.cpp, 181
- contains
  - MemRange, 48
- count
  - CeanReadDim, 22
  - OffloadDescriptor::ReadArrElements, 80
  - RefInfo, 81
  - VarDesc, 87
- cpu\_addr
  - AutoData, 20
  - PtrData, 78
- cpu\_buf
  - PtrData, 78
- cpu\_disp
  - OffloadDescriptor::VarExtra, 95
- cpu\_frequency
  - offload\_host.cpp, 146
  - offload\_host.h, 152
- cpu\_offset
  - OffloadDescriptor::VarExtra, 95
- cpu\_stack\_addr
  - PersistData, 76
- create\_envron\_for\_card
  - MicEnvVar, 51
- current\_number
  - CeanReadRanges, 23
- DEFAULT\_TARGET\_TYPE
  - offload.h, 129
- DL\_addr
  - offload\_util.h, 189
- DL\_close
  - offload\_util.h, 189
- DL\_open
  - offload\_util.h, 189
- DL\_sym
  - offload\_util.cpp, 188
  - offload\_util.h, 190
- DLL\_LOCAL
  - ofldbegin.cpp, 191
- DYNART\_STDERR\_PUTS
  - liboffload\_msg.c, 121
- data
  - FunctionDescriptor, 35
  - Image, 36
  - TargetImage, 83
- data\_offset
  - FunctionDescriptor, 35
- data\_received
  - \_Offload\_status, 17
  - mic\_lib::offload\_status, 60
- data\_sent
  - \_Offload\_status, 17
  - mic\_lib::offload\_status, 60
- default\_target\_number
  - mic\_lib, 50
- default\_target\_type

- mic\_lib, 50
- destroy\_thread\_data
  - Engine, 27
- device\_number
  - \_Offload\_status, 17
  - mic\_lib::offload\_status, 60
- Dim
  - ArrDesc, 19
  - CeanReadRanges, 23
- dim
  - arr\_desc, 18
- dim\_desc, 23
  - lindex, 24
  - lower, 24
  - size, 24
  - stride, 24
  - upper, 24
- DimDesc, 24
  - dv\_util.h, 113
  - Extent, 24
  - LowerBound, 24
  - Mult, 25
- direction
  - VarDesc, 87
- disp
  - VarDesc, 88
- dname
  - VarDesc2, 92
- dst
  - VarDesc, 88
- dst\_data
  - OffloadDescriptor::VarExtra, 95
- dump
  - FuncList, 33
  - VarList, 97
- dv\_size
  - dv\_util.h, 113
- dv\_util.cpp, 112
  - \_\_dv\_data\_length, 112
  - \_\_dv\_is\_allocated, 112
  - \_\_dv\_is\_contiguous, 112
  - init\_read\_ranges\_dv, 112
- dv\_util.h, 112
  - \_\_dv\_data\_length, 114
  - \_\_dv\_desc\_dump, 113
  - \_\_dv\_is\_allocated, 114
  - \_\_dv\_is\_contiguous, 114
  - ArrDesc, 113
  - ArrDescFlagsContiguous, 113
  - ArrDescFlagsDefined, 113
  - ArrDescFlagsNodealloc, 113
  - ArrDescMaxArrayRank, 113
  - DimDesc, 113
  - dv\_size, 113
  - init\_read\_ranges\_dv, 114
  - pArrDesc, 113
- el\_size
  - OffloadDescriptor::ReadArrElements, 80
- end
  - MemRange, 48
  - VarList, 97
- Engine, 25
  - ~Engine, 27
  - \_\_offload\_fini\_library, 29
  - \_\_offload\_init\_library\_once, 29
  - add\_lib, 27
  - add\_signal, 27
  - c\_func\_compute, 26
  - c\_func\_init, 26
  - c\_func\_var\_table\_copy, 26
  - c\_func\_var\_table\_size, 26
  - c\_funcs\_total, 26
  - c\_signal\_max, 29
  - c\_signal\_names, 29
  - compute, 27
  - destroy\_thread\_data, 27
  - Engine, 27
  - find\_auto\_data, 27
  - find\_ptr\_data, 27
  - find\_signal, 27
  - fini\_process, 27
  - get\_auto\_vars, 27
  - get\_logical\_index, 27
  - get\_physical\_index, 28
  - get\_pipeline, 28
  - get\_process, 28
  - init, 28
  - init\_device, 28
  - init\_process, 28
  - init\_ptr\_data, 28
  - insert\_auto\_data, 28
  - insert\_ptr\_data, 28
  - load\_libraries, 28
  - m\_func\_names, 29
  - m\_funcs, 29
  - m\_images, 29
  - m\_index, 29
  - m\_lock, 30
  - m\_persist\_list, 30
  - m\_physical\_index, 30
  - m\_proc\_number, 30
  - m\_process, 30
  - m\_ptr\_lock, 30
  - m\_ptr\_set, 30
  - m\_ready, 30
  - m\_signal\_lock, 30
  - m\_signal\_map, 30
  - PtrSet, 26
  - remove\_auto\_data, 28
  - remove\_ptr\_data, 29
  - set\_indexes, 29
  - SignalMap, 26
- EngineGetCount
  - COI, 13
- EngineGetHandle
  - COI, 13



- EngineGetIndex
  - coi\_server.h, 105
- entries
  - FuncTable, 34
  - VarTable, 98
- env\_var
  - MicEnvVar::VarValue, 98
- env\_var\_value
  - MicEnvVar::VarValue, 98
- env\_vars
  - MicEnvVar::CardEnvVars, 22
- error\_types
  - liboffload\_error\_codes.h, 117
- EventWait
  - COI, 13
- Extent
  - DimDesc, 24
- extent\_elements
  - VarDesc3, 94
- extent\_start
  - VarDesc3, 94
- find\_addr
  - FuncList, 33
- find\_auto\_data
  - Engine, 27
- find\_name
  - FuncList, 34
- find\_ptr\_data
  - Engine, 27
  - OffloadDescriptor, 63
- find\_signal
  - Engine, 27
- find\_var
  - MicEnvVar::CardEnvVars, 21
- fini
  - COI, 12
- fini\_process
  - Engine, 27
- firstMsg
  - liboffload\_msg.h, 126
- flag\_align\_is\_array
  - offload\_common.h, 138
- flag\_alloc\_elements\_is\_array
  - offload\_common.h, 138
- flag\_alloc\_elements\_is\_scalar
  - offload\_common.h, 138
- flag\_alloc\_if\_is\_array
  - offload\_common.h, 138
- flag\_alloc\_start\_is\_array
  - offload\_common.h, 138
- flag\_alloc\_start\_is\_scalar
  - offload\_common.h, 138
- flag\_extent\_elements\_is\_array
  - offload\_common.h, 139
- flag\_extent\_elements\_is\_scalar
  - offload\_common.h, 139
- flag\_extent\_start\_is\_array
  - offload\_common.h, 139
- flag\_extent\_start\_is\_scalar
  - offload\_common.h, 139
- flag\_free\_if\_is\_array
  - offload\_common.h, 139
- flag\_into\_elements\_is\_array
  - offload\_common.h, 139
- flag\_into\_elements\_is\_scalar
  - offload\_common.h, 139
- flag\_into\_start\_is\_array
  - offload\_common.h, 139
- flag\_into\_start\_is\_scalar
  - offload\_common.h, 139
- Flags
  - ArrDesc, 19
- flags
  - VarDesc, 88
- fpp
  - cean\_util.cpp, 99
- fptr\_table\_entries
  - offload\_myo\_host.cpp, 156
- FptrTableEntry, 32
  - funcAddr, 32
  - funcName, 32
  - localThunkAddr, 32
  - offload\_myo\_target.h, 160
- free\_if
  - VarDesc, 88
- free\_if\_array
  - VarDesc3, 94
- func
  - FuncTable::Entry, 31
  - InitTableEntry, 37
- funcAddr
  - FptrTableEntry, 32
- FuncList, 33
  - add\_table, 33
  - dump, 33
  - find\_addr, 33
  - find\_name, 34
  - FuncList, 33
  - FuncList, 33
  - m\_max\_name\_len, 34
  - max\_name\_length, 34
- funcName
  - FptrTableEntry, 32
- FuncTable, 34
  - entries, 34
  - max\_name\_len, 34
- FuncTable::Entry, 31
  - func, 31
  - name, 31
- FunctionDescriptor, 35
  - console\_enabled, 35
  - data, 35
  - data\_offset, 35
  - in\_data\_len, 35
  - offload\_number, 35
  - offload\_report\_level, 35

- out\_datalen, [35](#)
- timer\_enabled, [35](#)
- vars\_num, [36](#)
- GRAN\_CARD
  - orisl-lite.h, [194](#)
- GRAN\_THREAD
  - orisl-lite.h, [194](#)
- GET\_OFFLOAD\_NUMBER
  - offload\_host.cpp, [144](#)
- gather\_copyin\_data
  - OffloadDescriptor, [63](#)
- gather\_copyout\_data
  - OffloadDescriptor, [63](#)
- gen\_var\_descs\_for\_pointer\_array
  - OffloadDescriptor, [63](#)
- generate\_mem\_ranges
  - cean\_util.cpp, [100](#)
- generate\_mem\_ranges\_one\_rank
  - cean\_util.cpp, [100](#)
- generate\_one\_range
  - cean\_util.cpp, [100](#)
- get\_arr\_desc\_numbers
  - offload\_host.cpp, [145](#)
- get\_auto\_vars
  - Engine, [27](#)
  - Thread, [84](#)
- get\_buffer\_size
  - Marshaller, [46](#)
- get\_buffer\_start
  - Marshaller, [46](#)
- get\_card
  - MicEnvVar, [51](#)
- get\_el\_value
  - offload\_util.cpp, [188](#)
  - offload\_util.h, [190](#)
- get\_env\_var\_kind
  - MicEnvVar, [51](#)
- get\_logical\_index
  - Engine, [27](#)
- get\_next\_range
  - cean\_util.cpp, [100](#)
  - cean\_util.h, [101](#)
- get\_offload\_number
  - OffloadDescriptor, [63](#)
- get\_physical\_index
  - Engine, [28](#)
- get\_pipeline
  - Engine, [28](#)
  - Thread, [84](#)
- get\_process
  - Engine, [28](#)
- get\_reference
  - AutoData, [20](#)
  - PtrData, [78](#)
- get\_tfr\_size
  - Marshaller, [46](#)
- get\_timer\_data
  - OffloadDescriptor, [64](#)
- GetResult
  - MyoWrapper, [55](#)
- has\_length
  - VarDesc, [88](#)
- host\_entry\_cmp
  - offload\_engine.cpp, [140](#)
- HostFptrTableRegister
  - MyoWrapper, [55](#)
- HostVarTablePropagate
  - MyoWrapper, [55](#)
- htrace\_envname
  - offload\_host.cpp, [147](#)
- Image, [36](#)
  - data, [36](#)
  - size, [36](#)
- in
  - VarDesc, [89](#)
- in\_datalen
  - FunctionDescriptor, [35](#)
- init
  - COI, [12](#)
  - Engine, [28](#)
  - ORSL, [15](#)
- init\_buffer
  - Marshaller, [46](#)
- init\_device
  - Engine, [28](#)
- init\_mic\_address
  - OffloadDescriptor, [64](#)
- init\_offset
  - CeanReadRanges, [23](#)
- init\_process
  - Engine, [28](#)
- init\_ptr\_data
  - Engine, [28](#)
- init\_read\_ranges\_arr\_desc
  - cean\_util.cpp, [100](#)
  - cean\_util.h, [101](#)
- init\_read\_ranges\_dv
  - dv\_util.cpp, [112](#)
  - dv\_util.h, [114](#)
- init\_static\_ptr\_data
  - OffloadDescriptor, [64](#)
- InitTableEntry, [36](#)
  - func, [37](#)
- insert\_auto\_data
  - Engine, [28](#)
- insert\_ptr\_data
  - Engine, [28](#)
- into
  - VarDesc, [89](#)
- into\_elements
  - VarDesc3, [94](#)
- into\_start
  - VarDesc3, [94](#)
- is\_added
  - RefInfo, [81](#)

- is\_arr\_desc\_contiguous
  - cean\_util.cpp, 100
  - cean\_util.h, 101
- is\_arr\_ptr\_el
  - OffloadDescriptor::VarExtra, 95
- is\_available
  - COI, 13
  - MyoWrapper, 55
- is\_empty
  - OffloadDescriptor::ReadArrElements, 80
- is\_enabled
  - ORSL, 15
- is\_noncont\_dst
  - VarDesc, 89
- is\_noncont\_src
  - VarDesc, 89
- is\_signaled
  - OffloadDescriptor, 64
- is\_stack\_buf
  - VarDesc, 89
- is\_static
  - PtrData, 78
  - VarDesc, 89
- is\_static\_dstn
  - VarDesc, 89
- Iterator
  - VarList::Iterator, 37
- kmp\_affinity\_mask\_target\_t, 38
  - mask, 39
- kmp\_create\_affinity\_mask\_lrb
  - offload\_omp\_target.cpp, 167
  - offload\_table.cpp, 175
- kmp\_create\_affinity\_mask\_target
  - mic\_lib::kmp\_create\_affinity\_mask\_target, 39
  - offload.h, 131
  - offload\_omp\_host.cpp, 162
  - offload\_omp\_target.cpp, 167
- kmp\_destroy\_affinity\_mask\_lrb
  - offload\_omp\_target.cpp, 167
  - offload\_table.cpp, 175
- kmp\_destroy\_affinity\_mask\_target
  - mic\_lib::kmp\_destroy\_affinity\_mask\_target, 39
  - offload.h, 131
  - offload\_omp\_host.cpp, 162
  - offload\_omp\_target.cpp, 167
- kmp\_get\_affinity\_lrb
  - offload\_omp\_target.cpp, 168
  - offload\_table.cpp, 175
- kmp\_get\_affinity\_mask\_proc\_lrb
  - offload\_omp\_target.cpp, 168
  - offload\_table.cpp, 175
- kmp\_get\_affinity\_mask\_proc\_target
  - mic\_lib::kmp\_get\_affinity\_mask\_proc\_target, 40
  - offload.h, 131
  - offload\_omp\_host.cpp, 162
  - offload\_omp\_target.cpp, 168
- kmp\_get\_affinity\_max\_proc\_lrb
  - offload\_omp\_target.cpp, 168
- offload\_table.cpp, 175
  - kmp\_get\_affinity\_max\_proc\_target
    - mic\_lib::kmp\_get\_affinity\_max\_proc\_target, 40
    - offload.h, 131
    - offload\_omp\_host.cpp, 162
    - offload\_omp\_target.cpp, 168
  - kmp\_get\_affinity\_target
    - mic\_lib::kmp\_get\_affinity\_target, 40
    - offload.h, 131
    - offload\_omp\_host.cpp, 162
    - offload\_omp\_target.cpp, 168
  - kmp\_get\_blocktime\_lrb
    - offload\_omp\_target.cpp, 168
    - offload\_table.cpp, 175
  - kmp\_get\_blocktime\_target
    - mic\_lib::kmp\_get\_blocktime\_target, 41
    - offload.h, 131
    - offload\_omp\_host.cpp, 162
    - offload\_omp\_target.cpp, 168
  - kmp\_get\_library\_lrb
    - offload\_omp\_target.cpp, 168
    - offload\_table.cpp, 175
  - kmp\_get\_library\_target
    - mic\_lib::kmp\_get\_library\_target, 41
    - offload.h, 131
    - offload\_omp\_host.cpp, 162
    - offload\_omp\_target.cpp, 168
  - kmp\_get\_stacksize\_lrb
    - offload\_omp\_target.cpp, 168
    - offload\_table.cpp, 175
  - kmp\_get\_stacksize\_s\_lrb
    - offload\_omp\_target.cpp, 168
    - offload\_table.cpp, 176
  - kmp\_get\_stacksize\_s\_target
    - mic\_lib::kmp\_get\_stacksize\_s\_target, 41
    - offload.h, 131
    - offload\_omp\_host.cpp, 163
    - offload\_omp\_target.cpp, 168
  - kmp\_get\_stacksize\_target
    - mic\_lib::kmp\_get\_stacksize\_target, 42
    - offload.h, 131
    - offload\_omp\_host.cpp, 163
    - offload\_omp\_target.cpp, 168
  - kmp\_set\_affinity\_lrb
    - offload\_omp\_target.cpp, 169
    - offload\_table.cpp, 176
  - kmp\_set\_affinity\_mask\_proc\_lrb
    - offload\_omp\_target.cpp, 169
    - offload\_table.cpp, 176
  - kmp\_set\_affinity\_mask\_proc\_target
    - mic\_lib::kmp\_set\_affinity\_mask\_proc\_target, 42
    - offload.h, 131
    - offload\_omp\_host.cpp, 163
    - offload\_omp\_target.cpp, 169
  - kmp\_set\_affinity\_target
    - mic\_lib::kmp\_set\_affinity\_target, 42
    - offload.h, 131
    - offload\_omp\_host.cpp, 163

- offload\_omp\_target.cpp, 169
- kmp\_set\_blocktime\_lrb
  - offload\_omp\_target.cpp, 169
  - offload\_table.cpp, 176
- kmp\_set\_blocktime\_target
  - mic\_lib::kmp\_set\_blocktime\_target, 43
  - offload.h, 131
  - offload\_omp\_host.cpp, 163
  - offload\_omp\_target.cpp, 169
- kmp\_set\_defaults\_lrb
  - offload\_omp\_target.cpp, 169
  - offload\_table.cpp, 176
- kmp\_set\_defaults\_target
  - mic\_lib::kmp\_set\_defaults\_target, 43
  - offload.h, 132
  - offload\_omp\_host.cpp, 163
  - offload\_omp\_target.cpp, 169
- kmp\_set\_library\_lrb
  - offload\_omp\_target.cpp, 169
  - offload\_table.cpp, 176
- kmp\_set\_library\_serial\_lrb
  - offload\_omp\_target.cpp, 169
  - offload\_table.cpp, 176
- kmp\_set\_library\_serial\_target
  - mic\_lib::kmp\_set\_library\_serial\_target, 43
  - offload.h, 132
  - offload\_omp\_host.cpp, 163
  - offload\_omp\_target.cpp, 169
- kmp\_set\_library\_target
  - mic\_lib::kmp\_set\_library\_target, 44
  - offload.h, 132
  - offload\_omp\_host.cpp, 163
  - offload\_omp\_target.cpp, 169
- kmp\_set\_library\_throughput\_lrb
  - offload\_omp\_target.cpp, 169
  - offload\_table.cpp, 176
- kmp\_set\_library\_throughput\_target
  - mic\_lib::kmp\_set\_library\_throughput\_target, 44
  - offload.h, 132
  - offload\_omp\_host.cpp, 163
  - offload\_omp\_target.cpp, 169
- kmp\_set\_library\_turnaround\_lrb
  - offload\_omp\_target.cpp, 169
  - offload\_table.cpp, 176
- kmp\_set\_library\_turnaround\_target
  - mic\_lib::kmp\_set\_library\_turnaround\_target, 44
  - offload.h, 132
  - offload\_omp\_host.cpp, 163
  - offload\_omp\_target.cpp, 170
- kmp\_set\_stacksize\_lrb
  - offload\_omp\_target.cpp, 170
  - offload\_table.cpp, 176
- kmp\_set\_stacksize\_s\_lrb
  - offload\_omp\_target.cpp, 170
  - offload\_table.cpp, 176
- kmp\_set\_stacksize\_s\_target
  - mic\_lib::kmp\_set\_stacksize\_s\_target, 45
  - offload.h, 132
- offload\_omp\_host.cpp, 163
- offload\_omp\_target.cpp, 170
- kmp\_set\_stacksize\_target
  - mic\_lib::kmp\_set\_stacksize\_target, 45
  - offload.h, 132
  - offload\_omp\_host.cpp, 163
  - offload\_omp\_target.cpp, 170
- kmp\_unset\_affinity\_mask\_proc\_lrb
  - offload\_omp\_target.cpp, 170
  - offload\_table.cpp, 176
- kmp\_unset\_affinity\_mask\_proc\_target
  - mic\_lib::kmp\_unset\_affinity\_mask\_proc\_target, 45
  - offload.h, 132
  - offload\_omp\_host.cpp, 163
  - offload\_omp\_target.cpp, 170
- LIBOFFLOAD\_ABORT
  - liboffload\_error\_codes.h, 116
- LIBOFFLOAD\_ERROR
  - liboffload\_error\_codes.h, 116
- lastMsg
  - liboffload\_msg.h, 126
- last\_left
  - cean\_util.cpp, 100
- last\_noncont\_ind
  - CeanReadRanges, 23
- last\_right
  - cean\_util.cpp, 100
- Len
  - ArrDesc, 19
- length
  - MemRange, 48
  - MicEnvVar::VarValue, 98
- length\_cur
  - OffloadDescriptor::ReadArrElements, 80
- lib\_handle
  - COI, 13
- LibFini
  - MyoWrapper, 55
- LibInit
  - MyoWrapper, 56
- liboffload\_error\_codes.h
  - c\_bad\_ptr\_mem\_range, 118
  - c\_buf\_add\_ref, 118
  - c\_buf\_copy, 118
  - c\_buf\_create, 118
  - c\_buf\_create\_from\_mem, 118
  - c\_buf\_create\_out\_of\_mem, 118
  - c\_buf\_destroy, 118
  - c\_buf\_get\_address, 118
  - c\_buf\_map, 118
  - c\_buf\_read, 118
  - c\_buf\_release\_ref, 118
  - c\_buf\_set\_state, 118
  - c\_buf\_unmap, 118
  - c\_buf\_write, 118
  - c\_coipipe\_max\_number, 119
  - c\_destination\_is\_over, 118
  - c\_device\_is\_not\_available, 117

c\_different\_src\_and\_dstn\_sizes, 118  
 c\_event\_wait, 118  
 c\_get\_engine\_handle, 117  
 c\_get\_engine\_index, 117  
 c\_invalid\_device\_number, 117  
 c\_invalid\_env\_report\_value, 117  
 c\_invalid\_env\_var\_int\_value, 117  
 c\_invalid\_env\_var\_value, 117  
 c\_load\_library, 117  
 c\_merge\_var\_descs1, 117  
 c\_merge\_var\_descs2, 117  
 c\_mic\_init3, 117  
 c\_mic\_init4, 117  
 c\_mic\_init5, 117  
 c\_mic\_init6, 117  
 c\_mic\_parse\_env\_var\_list1, 117  
 c\_mic\_parse\_env\_var\_list2, 117  
 c\_mic\_process\_exit, 117  
 c\_mic\_process\_exit\_ret, 117  
 c\_mic\_process\_exit\_sig, 117  
 c\_multiple\_target\_exes, 118  
 c\_myotarget\_checkresult, 117  
 c\_myowrapper\_checkresult, 117  
 c\_no\_ptr\_data, 117  
 c\_no\_static\_var\_data, 117  
 c\_no\_target\_exe, 118  
 c\_non\_contiguous\_dope\_vector, 118  
 c\_offload1, 117  
 c\_offload\_descriptor\_offload, 117  
 c\_offload\_host\_alloc\_buffers, 120  
 c\_offload\_host\_alloc\_data\_buffer, 120  
 c\_offload\_host\_destroy\_buffers, 120  
 c\_offload\_host\_gather\_inputs, 120  
 c\_offload\_host\_initialize, 120  
 c\_offload\_host\_map\_in\_data\_buffer, 120  
 c\_offload\_host\_map\_out\_data\_buffer, 120  
 c\_offload\_host\_max\_phase, 120  
 c\_offload\_host\_scatter\_outputs, 120  
 c\_offload\_host\_send\_pointers, 120  
 c\_offload\_host\_setup\_buffers, 120  
 c\_offload\_host\_setup\_misc\_data, 120  
 c\_offload\_host\_start\_buffers\_reads, 120  
 c\_offload\_host\_start\_compute, 120  
 c\_offload\_host\_target\_acquire, 120  
 c\_offload\_host\_total\_offload, 120  
 c\_offload\_host\_unmap\_in\_data\_buffer, 120  
 c\_offload\_host\_unmap\_out\_data\_buffer, 120  
 c\_offload\_host\_wait\_buffers\_reads, 120  
 c\_offload\_host\_wait\_compute, 120  
 c\_offload\_host\_wait\_deps, 120  
 c\_offload\_malloc, 117  
 c\_offload\_signaled1, 117  
 c\_offload\_signaled2, 117  
 c\_offload\_target\_add\_buffer\_refs, 120  
 c\_offload\_target\_compute, 120  
 c\_offload\_target\_descriptor\_setup, 120  
 c\_offload\_target\_func\_lookup, 120  
 c\_offload\_target\_func\_time, 120  
 c\_offload\_target\_gather\_outputs, 120  
 c\_offload\_target\_max\_phase, 120  
 c\_offload\_target\_release\_buffer\_refs, 120  
 c\_offload\_target\_scatter\_inputs, 120  
 c\_offload\_target\_total\_time, 120  
 c\_omp\_invalid\_device\_num, 118  
 c\_omp\_invalid\_device\_num\_env, 118  
 c\_pipeline\_create, 117  
 c\_pipeline\_run\_func, 118  
 c\_pipeline\_start\_run\_funcs, 118  
 c\_pointer\_array\_mismatch, 118  
 c\_process\_create, 117  
 c\_process\_get\_func\_handles, 117  
 c\_process\_proxy\_flush, 117  
 c\_process\_wait\_shutdown, 117  
 c\_ranges\_dont\_match, 118  
 c\_receive\_func\_ptr, 117  
 c\_report\_bytes, 118  
 c\_report\_compute, 119  
 c\_report\_copyin\_data, 119  
 c\_report\_copyout\_data, 119  
 c\_report\_cpu\_time, 118  
 c\_report\_cpu\_to\_mic\_data, 118  
 c\_report\_create\_buf\_host, 119  
 c\_report\_create\_buf\_mic, 119  
 c\_report\_destroy, 119  
 c\_report\_file, 118  
 c\_report\_from\_file, 118  
 c\_report\_gather\_copyin\_data, 119  
 c\_report\_gather\_copyout\_data, 119  
 c\_report\_host, 118  
 c\_report\_init, 119  
 c\_report\_init\_func, 119  
 c\_report\_line, 118  
 c\_report\_logical\_card, 119  
 c\_report\_mic, 118  
 c\_report\_mic\_myo\_fptr, 119  
 c\_report\_mic\_myo\_shared, 119  
 c\_report\_mic\_time, 118  
 c\_report\_mic\_to\_cpu\_data, 118  
 c\_report\_myoacquire, 119  
 c\_report\_myofini, 119  
 c\_report\_myoinit, 119  
 c\_report\_myoregister, 119  
 c\_report\_myorelease, 119  
 c\_report\_myosharedalignedfree, 119  
 c\_report\_myosharedalignedmalloc, 119  
 c\_report\_myosharedfree, 119  
 c\_report\_myosharedmalloc, 119  
 c\_report\_offload, 119  
 c\_report\_physical\_card, 119  
 c\_report\_receive\_pointer\_data, 119  
 c\_report\_received\_pointer\_data, 119  
 c\_report\_register, 119  
 c\_report\_scatter\_copyin\_data, 119  
 c\_report\_scatter\_copyout\_data, 119  
 c\_report\_seconds, 118  
 c\_report\_send\_pointer\_data, 119

- c.report\_sent\_pointer\_data, 119
- c.report\_signal, 119
- c.report\_start, 119
- c.report\_start.target\_func, 119
- c.report\_state, 119
- c.report\_state\_signal, 119
- c.report\_tag, 118
- c.report\_target, 118
- c.report\_title, 118
- c.report\_unknown\_timer\_node, 118
- c.report\_unknown\_trace\_node, 119
- c.report\_unregister, 119
- c.report\_var, 119
- c.report\_w\_tag, 119
- c.report\_wait, 119
- c.send\_func\_ptr, 117
- c.slice\_of\_noncont\_array, 118
- c.unknown\_binary\_type, 118
- c.unknown\_var\_type, 117
- c.zero\_or\_neg\_ptr\_len, 118
- c.zero\_or\_neg\_transfer\_size, 118
- liboffload\_msg.h
  - \_\_dummy\_\_, 123
  - firstMsg, 126
  - lastMsg, 126
  - msg\_c\_bad\_ptr\_mem\_range, 124
  - msg\_c\_buf\_add\_ref, 124
  - msg\_c\_buf\_copy, 124
  - msg\_c\_buf\_create, 124
  - msg\_c\_buf\_create\_from\_mem, 124
  - msg\_c\_buf\_create\_out\_of\_mem, 124
  - msg\_c\_buf\_destroy, 124
  - msg\_c\_buf\_get\_address, 124
  - msg\_c\_buf\_map, 124
  - msg\_c\_buf\_read, 124
  - msg\_c\_buf\_release\_ref, 124
  - msg\_c\_buf\_set\_state, 124
  - msg\_c\_buf\_unmap, 124
  - msg\_c\_buf\_write, 124
  - msg\_c\_coi\_pipeline\_max\_number, 126
  - msg\_c\_destination\_is\_over, 126
  - msg\_c\_device\_is\_not\_available, 123
  - msg\_c\_different\_src\_and\_dstn\_sizes, 124
  - msg\_c\_event\_wait, 124
  - msg\_c\_get\_engine\_handle, 124
  - msg\_c\_get\_engine\_index, 124
  - msg\_c\_invalid\_device\_number, 123
  - msg\_c\_invalid\_env\_report\_value, 123
  - msg\_c\_invalid\_env\_var\_int\_value, 123
  - msg\_c\_invalid\_env\_var\_value, 123
  - msg\_c\_load\_library, 124
  - msg\_c\_merge\_var\_descs1, 123
  - msg\_c\_merge\_var\_descs2, 123
  - msg\_c\_mic\_init3, 123
  - msg\_c\_mic\_init4, 123
  - msg\_c\_mic\_init5, 123
  - msg\_c\_mic\_init6, 123
  - msg\_c\_mic\_parse\_env\_var\_list1, 123
  - msg\_c\_mic\_parse\_env\_var\_list2, 123
  - msg\_c\_mic\_process\_exit, 123
  - msg\_c\_mic\_process\_exit\_ret, 123
  - msg\_c\_mic\_process\_exit\_sig, 123
  - msg\_c\_multiple\_target\_exes, 124
  - msg\_c\_myotarget\_checkresult, 123
  - msg\_c\_myowrapper\_checkresult, 123
  - msg\_c\_no\_ptr\_data, 124
  - msg\_c\_no\_static\_var\_data, 124
  - msg\_c\_no\_target\_exe, 124
  - msg\_c\_non\_contiguous\_dope\_vector, 124
  - msg\_c\_offload1, 123
  - msg\_c\_offload\_descriptor\_offload, 123
  - msg\_c\_offload\_malloc, 123
  - msg\_c\_offload\_signaled1, 123
  - msg\_c\_offload\_signaled2, 123
  - msg\_c\_omp\_invalid\_device\_num, 124
  - msg\_c\_omp\_invalid\_device\_num\_env, 124
  - msg\_c\_pipeline\_create, 124
  - msg\_c\_pipeline\_run\_func, 124
  - msg\_c\_pipeline\_start\_run\_funcs, 124
  - msg\_c\_pointer\_array\_mismatch, 126
  - msg\_c\_process\_create, 124
  - msg\_c\_process\_get\_func\_handles, 124
  - msg\_c\_process\_proxy\_flush, 124
  - msg\_c\_process\_wait\_shutdown, 124
  - msg\_c\_ranges\_dont\_match, 126
  - msg\_c\_receive\_func\_ptr, 123
  - msg\_c\_report\_bytes, 124
  - msg\_c\_report\_compute, 125
  - msg\_c\_report\_copyin\_data, 125
  - msg\_c\_report\_copyout\_data, 125
  - msg\_c\_report\_cpu\_time, 124
  - msg\_c\_report\_cpu\_to\_mic\_data, 125
  - msg\_c\_report\_create\_buf\_host, 125
  - msg\_c\_report\_create\_buf\_mic, 125
  - msg\_c\_report\_destroy, 125
  - msg\_c\_report\_file, 125
  - msg\_c\_report\_from\_file, 125
  - msg\_c\_report\_gather\_copyin\_data, 125
  - msg\_c\_report\_gather\_copyout\_data, 125
  - msg\_c\_report\_host, 124
  - msg\_c\_report\_host\_alloc\_buffers, 126
  - msg\_c\_report\_host\_alloc\_data\_buffer, 126
  - msg\_c\_report\_host\_destroy\_buffers, 126
  - msg\_c\_report\_host\_gather\_inputs, 126
  - msg\_c\_report\_host\_initialize, 126
  - msg\_c\_report\_host\_map\_in\_data\_buffer, 126
  - msg\_c\_report\_host\_map\_out\_data\_buffer, 126
  - msg\_c\_report\_host\_scatter\_outputs, 126
  - msg\_c\_report\_host\_send\_pointers, 126
  - msg\_c\_report\_host\_setup\_buffers, 126
  - msg\_c\_report\_host\_setup\_misc\_data, 126
  - msg\_c\_report\_host\_start\_buffers\_reads, 126
  - msg\_c\_report\_host\_start\_compute, 126
  - msg\_c\_report\_host\_target\_acquire, 126
  - msg\_c\_report\_host\_total\_offload\_time, 126
  - msg\_c\_report\_host\_unmap\_in\_data\_buffer, 126

- msg\_c.report\_host\_unmap\_out\_data\_buffer, 126
- msg\_c.report\_host\_wait\_buffers\_reads, 126
- msg\_c.report\_host\_wait\_compute, 126
- msg\_c.report\_host\_wait\_deps, 126
- msg\_c.report\_init, 125
- msg\_c.report\_init\_func, 125
- msg\_c.report\_line, 125
- msg\_c.report\_logical\_card, 125
- msg\_c.report\_mic, 124
- msg\_c.report\_mic.myo\_fptr, 125
- msg\_c.report\_mic.myo\_shared, 125
- msg\_c.report\_mic.time, 124
- msg\_c.report\_mic.to\_cpu\_data, 125
- msg\_c.report\_myoacquire, 125
- msg\_c.report\_myofini, 125
- msg\_c.report\_myoinit, 125
- msg\_c.report\_myoregister, 125
- msg\_c.report\_myorelease, 126
- msg\_c.report\_myosharedalignedfree, 125
- msg\_c.report\_myosharedalignedmalloc, 125
- msg\_c.report\_myosharedfree, 125
- msg\_c.report\_myosharedmalloc, 125
- msg\_c.report\_offload, 125
- msg\_c.report\_physical\_card, 125
- msg\_c.report\_receive\_pointer\_data, 125
- msg\_c.report\_received\_pointer\_data, 125
- msg\_c.report\_register, 125
- msg\_c.report\_scatter\_copyin\_data, 125
- msg\_c.report\_scatter\_copyout\_data, 125
- msg\_c.report\_seconds, 124
- msg\_c.report\_send\_pointer\_data, 125
- msg\_c.report\_sent\_pointer\_data, 125
- msg\_c.report\_signal, 125
- msg\_c.report\_start, 125
- msg\_c.report\_start\_target\_func, 125
- msg\_c.report\_state, 125
- msg\_c.report\_state\_signal, 125
- msg\_c.report\_tag, 125
- msg\_c.report\_target\_add\_buffer\_refs, 126
- msg\_c.report\_target\_compute, 126
- msg\_c.report\_target\_descriptor\_setup, 126
- msg\_c.report\_target\_func\_lookup, 126
- msg\_c.report\_target\_func.time, 126
- msg\_c.report\_target\_gather\_outputs, 126
- msg\_c.report\_target\_release\_buffer\_refs, 126
- msg\_c.report\_target\_scatter\_inputs, 126
- msg\_c.report\_target\_total.time, 126
- msg\_c.report\_title, 124
- msg\_c.report\_unknown\_timer\_node, 124
- msg\_c.report\_unknown\_trace\_node, 124
- msg\_c.report\_unregister, 125
- msg\_c.report\_var, 125
- msg\_c.report\_w\_tag, 125
- msg\_c.report\_wait, 125
- msg\_c.send\_func\_ptr, 123
- msg\_c.slice\_of\_noncont\_array, 126
- msg\_c.unknown\_binary\_type, 124
- msg\_c.unknown\_var\_type, 123
- msg\_c.zero\_or\_neg\_ptr\_len, 124
- msg\_c.zero\_or\_neg\_transfer\_size, 124
- liboffload\_error.c, 114
  - \_\_liboffload\_error\_support, 114
  - report\_get\_host\_stage\_str, 114
  - report\_get\_message\_str, 114
  - report\_get\_target\_stage\_str, 115
  - va\_copy, 114
- liboffload\_error\_codes.h, 115
  - \_\_liboffload\_error\_support, 120
  - \_\_liboffload\_report\_support, 120
  - error\_types, 117
  - offload\_get\_message\_str, 121
  - OffloadHostPhase, 119
  - OffloadTargetPhase, 120
  - report\_get\_host\_stage\_str, 121
  - report\_get\_message\_str, 121
  - report\_get\_target\_stage\_str, 121
  - test\_msg\_cat, 116
  - test\_msg\_cat1, 116
  - write\_message, 121
- liboffload\_msg.c, 121
  - offload\_get\_message\_str, 121
  - write\_message, 121
- liboffload\_msg.h, 122
- index
  - dim\_desc, 24
- load\_libraries
  - Engine, 28
- LoadLibrary
  - MyoWrapper, 56
- localThunkAddr
  - FptrTableEntry, 32
- lock
  - mutex.t, 53
  - omp\_lock\_target.t, 72
  - omp\_nest\_lock\_target.t, 72
- lower
  - dim\_desc, 24
- LowerBound
  - DimDesc, 24
- m\_acquire
  - MyoWrapper, 57
- m\_addr\_coipipe\_counter
  - Thread, 84
- m\_auto\_vars
  - Thread, 84
- m\_buffers
  - OffloadDescriptor, 66
- m\_compute\_buffers
  - OffloadDescriptor, 66
- m\_destroy\_buffers
  - OffloadDescriptor, 66
- m\_destroy\_stack
  - OffloadDescriptor, 66
- m\_device
  - OffloadDescriptor, 66
- m\_entry



- VarList::Iterator, 38
- m\_func\_desc
  - OffloadDescriptor, 66
- m\_func\_desc\_size
  - OffloadDescriptor, 66
- m\_func\_names
  - Engine, 29
- m\_funcs
  - Engine, 29
- m\_get\_result
  - MyoWrapper, 57
- m\_head
  - TableList, 82
- m\_host\_fptr\_table\_register
  - MyoWrapper, 57
- m\_host\_var\_table\_propagate
  - MyoWrapper, 57
- m\_images
  - Engine, 29
- m\_in
  - OffloadDescriptor, 66
- m\_in\_datalen
  - OffloadDescriptor, 66
- m\_in\_deps
  - OffloadDescriptor, 67
- m\_in\_deps\_total
  - OffloadDescriptor, 67
- m\_index
  - Engine, 29
- m\_inout\_buf
  - OffloadDescriptor, 67
- m\_is\_available
  - MyoWrapper, 57
- m\_is\_mandatory
  - OffloadDescriptor, 67
- m\_is\_openmp
  - OffloadDescriptor, 67
- m\_length
  - MemRange, 48
- m\_lib\_fini
  - MyoWrapper, 57
- m\_lib\_handle
  - MyoWrapper, 57
- m\_lib\_init
  - MyoWrapper, 57
- m\_lock
  - Engine, 30
  - mutex\_t, 53
  - TableList, 82
- m\_max\_name\_len
  - FuncList, 34
- m\_mutex
  - mutex\_locker\_t, 52
- m\_need\_runfunction
  - OffloadDescriptor, 67
- m\_node
  - VarList::Iterator, 38
- m\_offload\_number
  - OffloadDescriptor, 67
- m\_out
  - OffloadDescriptor, 67
- m\_out\_datalen
  - OffloadDescriptor, 67
- m\_out\_deps
  - OffloadDescriptor, 67
- m\_out\_deps\_total
  - OffloadDescriptor, 67
- m\_persist\_list
  - Engine, 30
- m\_physical\_index
  - Engine, 30
- m\_pipelines
  - Thread, 84
- m\_proc\_number
  - Engine, 30
- m\_process
  - Engine, 30
- m\_ptr\_lock
  - Engine, 30
- m\_ptr\_set
  - Engine, 30
- m\_ready
  - Engine, 30
- m\_release
  - MyoWrapper, 57
- m\_remote\_call
  - MyoWrapper, 57
- m\_remote\_thunk\_call
  - MyoWrapper, 57
- m\_shared\_aligned\_free
  - MyoWrapper, 57
- m\_shared\_aligned\_malloc
  - MyoWrapper, 57
- m\_shared\_free
  - MyoWrapper, 58
- m\_shared\_malloc
  - MyoWrapper, 58
- m\_signal\_lock
  - Engine, 30
- m\_signal\_map
  - Engine, 30
- m\_stack\_ptr\_data
  - OffloadDescriptor, 68
- m\_start
  - MemRange, 48
- m\_status
  - OffloadDescriptor, 68
- m\_timer\_data
  - OffloadDescriptor, 68
- m\_vars
  - OffloadDescriptor, 68
- m\_vars\_extra
  - OffloadDescriptor, 68
- m\_vars\_total
  - OffloadDescriptor, 68
- MAIN\_



- oflbegin.cpp, 191
- MAX\_TARGET\_NAME
  - offload\_host.h, 150
- MESSAGE\_TABLE\_NAME
  - liboffload\_msg.h, 123, 126
- MIC\_ENGINES\_MAX
  - coi\_client.h, 103
- MYO\_VERSION1
  - offload\_myo\_host.cpp, 154
- main
  - offload\_target\_main.cpp, 183
  - oflbegin.cpp, 191
- make\_arr\_desc
  - offload\_host.cpp, 145
- Marshaller, 45
  - buffer\_ptr, 47
  - buffer\_size, 47
  - buffer\_start, 47
  - get\_buffer\_size, 46
  - get\_buffer\_start, 46
  - get\_tfr\_size, 46
  - init\_buffer, 46
  - Marshaller, 46
  - receive\_data, 46
  - receive\_func\_ptr, 46
  - send\_data, 47
  - send\_func\_ptr, 47
  - tfr\_size, 47
- mask
  - kmp\_affinity\_mask\_target\_t, 39
- max\_name\_len
  - FuncTable, 34
- max\_name\_length
  - FuncList, 34
- MemRange, 47
  - contains, 48
  - end, 48
  - length, 48
  - m\_length, 48
  - m\_start, 48
  - MemRange, 48
  - MemRange, 48
  - overlaps, 48
  - start, 48
- merge\_var\_descs
  - OffloadDescriptor, 64
- mic\_addr
  - PtrData, 78
- mic\_buf
  - PtrData, 78
- mic\_buffer\_size
  - offload\_host.cpp, 147
  - offload\_host.h, 152
- mic\_engines
  - offload\_host.cpp, 147
  - offload\_host.h, 152
- mic\_engines\_total
  - offload\_host.cpp, 147
- offload\_host.h, 152
  - offload\_target.cpp, 181
  - offload\_target.h, 183
- mic\_env\_vars
  - offload\_host.cpp, 147
  - offload\_host.h, 152
- mic\_frequency
  - offload\_target.cpp, 181
  - offload\_target.h, 183
- mic\_index
  - offload\_common.h, 139
  - offload\_target.cpp, 181
  - offload\_target.h, 183
  - offload\_trace.cpp, 186
- mic\_lib, 49
  - default\_target\_number, 50
  - default\_target\_type, 50
  - target\_mic, 50
- mic\_lib.f90, 127
- mic\_lib::kmp\_create\_affinity\_mask\_target, 39
  - kmp\_create\_affinity\_mask\_target, 39
- mic\_lib::kmp\_destroy\_affinity\_mask\_target, 39
  - kmp\_destroy\_affinity\_mask\_target, 39
- mic\_lib::kmp\_get\_affinity\_mask\_proc\_target, 39
- mic\_lib::kmp\_get\_affinity\_max\_proc\_target, 40
- mic\_lib::kmp\_get\_affinity\_target, 40
  - kmp\_get\_affinity\_target, 40
- mic\_lib::kmp\_get\_blocktime\_target, 40
  - kmp\_get\_blocktime\_target, 41
- mic\_lib::kmp\_get\_library\_target, 41
  - kmp\_get\_library\_target, 41
- mic\_lib::kmp\_get\_stacksize\_s\_target, 41
  - kmp\_get\_stacksize\_s\_target, 41
- mic\_lib::kmp\_get\_stacksize\_target, 41
  - kmp\_get\_stacksize\_target, 42
- mic\_lib::kmp\_set\_affinity\_mask\_proc\_target, 42
- mic\_lib::kmp\_set\_affinity\_target, 42
  - kmp\_set\_affinity\_target, 42
- mic\_lib::kmp\_set\_blocktime\_target, 42
  - kmp\_set\_blocktime\_target, 43
- mic\_lib::kmp\_set\_defaults\_target, 43
  - kmp\_set\_defaults\_target, 43
- mic\_lib::kmp\_set\_library\_serial\_target, 43
  - kmp\_set\_library\_serial\_target, 43
- mic\_lib::kmp\_set\_library\_target, 43
  - kmp\_set\_library\_target, 44
- mic\_lib::kmp\_set\_library\_throughput\_target, 44
  - kmp\_set\_library\_throughput\_target, 44
- mic\_lib::kmp\_set\_library\_turnaround\_target, 44
  - kmp\_set\_library\_turnaround\_target, 44
- mic\_lib::kmp\_set\_stacksize\_s\_target, 44
  - kmp\_set\_stacksize\_s\_target, 45
- mic\_lib::kmp\_set\_stacksize\_target, 45
  - kmp\_set\_stacksize\_target, 45
- mic\_lib::kmp\_unset\_affinity\_mask\_proc\_target, 45
- mic\_lib::offload\_get\_device\_number, 58
  - offload\_get\_device\_number, 59
- mic\_lib::offload\_get\_physical\_device\_number, 59

- mic\_lib::offload\_number\_of\_devices, 59
  - offload\_number\_of\_devices, 59
- mic\_lib::offload\_report, 59
  - offload\_report, 60
- mic\_lib::offload\_signaled, 60
  - offload\_signaled, 60
- mic\_lib::offload\_status, 60
  - data\_received, 60
  - data\_sent, 60
  - device\_number, 60
  - result, 60
- mic\_lib::omp\_destroy\_lock\_target, 68
  - omp\_destroy\_lock\_target, 69
- mic\_lib::omp\_destroy\_nest\_lock\_target, 69
  - omp\_destroy\_nest\_lock\_target, 69
- mic\_lib::omp\_get\_dynamic\_target, 69
  - omp\_get\_dynamic\_target, 69
- mic\_lib::omp\_get\_max\_threads\_target, 69
  - omp\_get\_max\_threads\_target, 70
- mic\_lib::omp\_get\_nested\_target, 70
  - omp\_get\_nested\_target, 70
- mic\_lib::omp\_get\_num\_procs\_target, 70
  - omp\_get\_num\_procs\_target, 70
- mic\_lib::omp\_get\_schedule\_target, 70
  - omp\_get\_schedule\_target, 71
- mic\_lib::omp\_init\_lock\_target, 71
  - omp\_init\_lock\_target, 71
- mic\_lib::omp\_init\_nest\_lock\_target, 71
  - omp\_init\_nest\_lock\_target, 71
- mic\_lib::omp\_set\_dynamic\_target, 72
  - omp\_set\_dynamic\_target, 72
- mic\_lib::omp\_set\_lock\_target, 73
  - omp\_set\_lock\_target, 73
- mic\_lib::omp\_set\_nest\_lock\_target, 73
  - omp\_set\_nest\_lock\_target, 73
- mic\_lib::omp\_set\_nested\_target, 73
  - omp\_set\_nested\_target, 73
- mic\_lib::omp\_set\_num\_threads\_target, 74
  - omp\_set\_num\_threads\_target, 74
- mic\_lib::omp\_set\_schedule\_target, 74
  - omp\_set\_schedule\_target, 74
- mic\_lib::omp\_test\_lock\_target, 74
  - omp\_test\_lock\_target, 74
- mic\_lib::omp\_test\_nest\_lock\_target, 75
  - omp\_test\_nest\_lock\_target, 75
- mic\_lib::omp\_unset\_lock\_target, 75
  - omp\_unset\_lock\_target, 75
- mic\_lib::omp\_unset\_nest\_lock\_target, 75
  - omp\_unset\_nest\_lock\_target, 75
- mic\_library\_path
  - offload\_host.cpp, 147
  - offload\_host.h, 152
- mic\_offset
  - PtrData, 79
  - VarDesc, 90
- mic\_parse\_env\_var\_list
  - MicEnvVar, 51
- mic\_proxy\_fs\_root
  - offload\_host.cpp, 147
  - offload\_host.h, 152
- mic\_proxy\_io
  - offload\_host.cpp, 147
  - offload\_host.h, 153
- mic\_stack\_size
  - offload\_host.cpp, 147
  - offload\_host.h, 153
- mic\_thread\_key
  - offload\_host.cpp, 147
  - offload\_host.h, 153
- mic\_use\_2mb\_buffers\_envname
  - offload\_host.cpp, 147
- mic\_use\_async\_buffer\_read\_envname
  - offload\_host.cpp, 148
- mic\_use\_async\_buffer\_write\_envname
  - offload\_host.cpp, 148
- MicEnvVar, 50
  - ~MicEnvVar, 51
  - add\_env\_var, 51
  - analyze\_env\_var, 51
  - any\_card, 51
  - card\_spec\_list, 51
  - common\_vars, 51
  - create\_environ\_for\_card, 51
  - get\_card, 51
  - get\_env\_var\_kind, 51
  - mic\_parse\_env\_var\_list, 51
  - MicEnvVar, 51
  - MicEnvVar, 51
  - prefix, 52
  - set\_prefix, 51
- MicEnvVar::CardEnvVars, 21
  - ~CardEnvVars, 21
  - add\_new\_env\_var, 21
  - card\_number, 22
  - CardEnvVars, 21
  - env\_vars, 22
  - find\_var, 21
- MicEnvVar::VarValue, 98
  - ~VarValue, 98
  - env\_var, 98
  - env\_var\_value, 98
  - length, 98
  - VarValue, 98
- MicEnvVarKind
  - offload\_env.h, 142
- msg\_c.bad\_ptr\_mem\_range
  - liboffload\_msg.h, 124
- msg\_c.buf\_add\_ref
  - liboffload\_msg.h, 124
- msg\_c.buf\_copy
  - liboffload\_msg.h, 124
- msg\_c.buf\_create
  - liboffload\_msg.h, 124
- msg\_c.buf\_create\_from\_mem
  - liboffload\_msg.h, 124
- msg\_c.buf\_create\_out\_of\_mem

- liboffload\_msg.h, 124
- msg\_c\_buf\_destroy
  - liboffload\_msg.h, 124
- msg\_c\_buf\_get\_address
  - liboffload\_msg.h, 124
- msg\_c\_buf\_map
  - liboffload\_msg.h, 124
- msg\_c\_buf\_read
  - liboffload\_msg.h, 124
- msg\_c\_buf\_release\_ref
  - liboffload\_msg.h, 124
- msg\_c\_buf\_set\_state
  - liboffload\_msg.h, 124
- msg\_c\_buf\_unmap
  - liboffload\_msg.h, 124
- msg\_c\_buf\_write
  - liboffload\_msg.h, 124
- msg\_c\_coi\_pipeline\_max\_number
  - liboffload\_msg.h, 126
- msg\_c\_destination\_is\_over
  - liboffload\_msg.h, 126
- msg\_c\_device\_is\_not\_available
  - liboffload\_msg.h, 123
- msg\_c\_different\_src\_and\_dstn\_sizes
  - liboffload\_msg.h, 124
- msg\_c\_event\_wait
  - liboffload\_msg.h, 124
- msg\_c\_get\_engine\_handle
  - liboffload\_msg.h, 124
- msg\_c\_get\_engine\_index
  - liboffload\_msg.h, 124
- msg\_c\_invalid\_device\_number
  - liboffload\_msg.h, 123
- msg\_c\_invalid\_env\_report\_value
  - liboffload\_msg.h, 123
- msg\_c\_invalid\_env\_var\_int\_value
  - liboffload\_msg.h, 123
- msg\_c\_invalid\_env\_var\_value
  - liboffload\_msg.h, 123
- msg\_c\_load\_library
  - liboffload\_msg.h, 124
- msg\_c\_merge\_var\_descs1
  - liboffload\_msg.h, 123
- msg\_c\_merge\_var\_descs2
  - liboffload\_msg.h, 123
- msg\_c\_mic\_init3
  - liboffload\_msg.h, 123
- msg\_c\_mic\_init4
  - liboffload\_msg.h, 123
- msg\_c\_mic\_init5
  - liboffload\_msg.h, 123
- msg\_c\_mic\_init6
  - liboffload\_msg.h, 123
- msg\_c\_mic\_parse\_env\_var\_list1
  - liboffload\_msg.h, 123
- msg\_c\_mic\_parse\_env\_var\_list2
  - liboffload\_msg.h, 123
- msg\_c\_mic\_process\_exit
  - liboffload\_msg.h, 123
- msg\_c\_mic\_process\_exit\_ret
  - liboffload\_msg.h, 123
- msg\_c\_mic\_process\_exit\_sig
  - liboffload\_msg.h, 123
- msg\_c\_multiple\_target\_exes
  - liboffload\_msg.h, 124
- msg\_c\_myotarget\_checkresult
  - liboffload\_msg.h, 123
- msg\_c\_myowrapper\_checkresult
  - liboffload\_msg.h, 123
- msg\_c\_no\_ptr\_data
  - liboffload\_msg.h, 124
- msg\_c\_no\_static\_var\_data
  - liboffload\_msg.h, 124
- msg\_c\_no\_target\_exe
  - liboffload\_msg.h, 124
- msg\_c\_non\_contiguous\_dope\_vector
  - liboffload\_msg.h, 124
- msg\_c\_offload1
  - liboffload\_msg.h, 123
- msg\_c\_offload\_descriptor\_offload
  - liboffload\_msg.h, 123
- msg\_c\_offload\_malloc
  - liboffload\_msg.h, 123
- msg\_c\_offload\_signaled1
  - liboffload\_msg.h, 123
- msg\_c\_offload\_signaled2
  - liboffload\_msg.h, 123
- msg\_c\_omp\_invalid\_device\_num
  - liboffload\_msg.h, 124
- msg\_c\_omp\_invalid\_device\_num\_env
  - liboffload\_msg.h, 124
- msg\_c\_pipeline\_create
  - liboffload\_msg.h, 124
- msg\_c\_pipeline\_run\_func
  - liboffload\_msg.h, 124
- msg\_c\_pipeline\_start\_run\_funcs
  - liboffload\_msg.h, 124
- msg\_c\_pointer\_array\_mismatch
  - liboffload\_msg.h, 126
- msg\_c\_process\_create
  - liboffload\_msg.h, 124
- msg\_c\_process\_get\_func\_handles
  - liboffload\_msg.h, 124
- msg\_c\_process\_proxy\_flush
  - liboffload\_msg.h, 124
- msg\_c\_process\_wait\_shutdown
  - liboffload\_msg.h, 124
- msg\_c\_ranges\_dont\_match
  - liboffload\_msg.h, 126
- msg\_c\_receive\_func\_ptr
  - liboffload\_msg.h, 123
- msg\_c\_report\_bytes
  - liboffload\_msg.h, 124
- msg\_c\_report\_compute
  - liboffload\_msg.h, 125
- msg\_c\_report\_copyin\_data

- liboffload\_msg.h, 125
- msg\_c\_report\_copyout\_data
  - liboffload\_msg.h, 125
- msg\_c\_report\_cpu\_time
  - liboffload\_msg.h, 124
- msg\_c\_report\_cpu\_to\_mic\_data
  - liboffload\_msg.h, 125
- msg\_c\_report\_create\_buf\_host
  - liboffload\_msg.h, 125
- msg\_c\_report\_create\_buf\_mic
  - liboffload\_msg.h, 125
- msg\_c\_report\_destroy
  - liboffload\_msg.h, 125
- msg\_c\_report\_file
  - liboffload\_msg.h, 125
- msg\_c\_report\_from\_file
  - liboffload\_msg.h, 125
- msg\_c\_report\_gather\_copyin\_data
  - liboffload\_msg.h, 125
- msg\_c\_report\_gather\_copyout\_data
  - liboffload\_msg.h, 125
- msg\_c\_report\_host
  - liboffload\_msg.h, 124
- msg\_c\_report\_host\_alloc\_buffers
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_alloc\_data\_buffer
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_destroy\_buffers
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_gather\_inputs
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_initialize
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_map\_in\_data\_buffer
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_map\_out\_data\_buffer
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_scatter\_outputs
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_send\_pointers
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_setup\_buffers
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_setup\_misc\_data
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_start\_buffers\_reads
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_start\_compute
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_target\_acquire
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_total\_offload\_time
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_unmap\_in\_data\_buffer
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_unmap\_out\_data\_buffer
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_wait\_buffers\_reads
  - liboffload\_msg.h, 126
- liboffload\_msg.h, 126
- msg\_c\_report\_host\_wait\_compute
  - liboffload\_msg.h, 126
- msg\_c\_report\_host\_wait\_deps
  - liboffload\_msg.h, 126
- msg\_c\_report\_init
  - liboffload\_msg.h, 125
- msg\_c\_report\_init\_func
  - liboffload\_msg.h, 125
- msg\_c\_report\_line
  - liboffload\_msg.h, 125
- msg\_c\_report\_logical\_card
  - liboffload\_msg.h, 125
- msg\_c\_report\_mic
  - liboffload\_msg.h, 124
- msg\_c\_report\_mic\_myo\_fptr
  - liboffload\_msg.h, 125
- msg\_c\_report\_mic\_myo\_shared
  - liboffload\_msg.h, 125
- msg\_c\_report\_mic\_time
  - liboffload\_msg.h, 124
- msg\_c\_report\_mic\_to\_cpu\_data
  - liboffload\_msg.h, 125
- msg\_c\_report\_myoacquire
  - liboffload\_msg.h, 125
- msg\_c\_report\_myofini
  - liboffload\_msg.h, 125
- msg\_c\_report\_myoint
  - liboffload\_msg.h, 125
- msg\_c\_report\_myoregister
  - liboffload\_msg.h, 125
- msg\_c\_report\_myorelease
  - liboffload\_msg.h, 126
- msg\_c\_report\_myosharedalignedfree
  - liboffload\_msg.h, 125
- msg\_c\_report\_myosharedalignedmalloc
  - liboffload\_msg.h, 125
- msg\_c\_report\_myosharedfree
  - liboffload\_msg.h, 125
- msg\_c\_report\_myosharedmalloc
  - liboffload\_msg.h, 125
- msg\_c\_report\_offload
  - liboffload\_msg.h, 125
- msg\_c\_report\_physical\_card
  - liboffload\_msg.h, 125
- msg\_c\_report\_receive\_pointer\_data
  - liboffload\_msg.h, 125
- msg\_c\_report\_received\_pointer\_data
  - liboffload\_msg.h, 125
- msg\_c\_report\_register
  - liboffload\_msg.h, 125
- msg\_c\_report\_scatter\_copyin\_data
  - liboffload\_msg.h, 125
- msg\_c\_report\_scatter\_copyout\_data
  - liboffload\_msg.h, 125
- msg\_c\_report\_seconds
  - liboffload\_msg.h, 124
- msg\_c\_report\_send\_pointer\_data

- liboffload\_msg.h, 125
- msg\_c\_report\_sent\_pointer\_data
  - liboffload\_msg.h, 125
- msg\_c\_report\_signal
  - liboffload\_msg.h, 125
- msg\_c\_report\_start
  - liboffload\_msg.h, 125
- msg\_c\_report\_start\_target\_func
  - liboffload\_msg.h, 125
- msg\_c\_report\_state
  - liboffload\_msg.h, 125
- msg\_c\_report\_state\_signal
  - liboffload\_msg.h, 125
- msg\_c\_report\_tag
  - liboffload\_msg.h, 125
- msg\_c\_report\_target\_add\_buffer\_refs
  - liboffload\_msg.h, 126
- msg\_c\_report\_target\_compute
  - liboffload\_msg.h, 126
- msg\_c\_report\_target\_descriptor\_setup
  - liboffload\_msg.h, 126
- msg\_c\_report\_target\_func\_lookup
  - liboffload\_msg.h, 126
- msg\_c\_report\_target\_func\_time
  - liboffload\_msg.h, 126
- msg\_c\_report\_target\_gather\_outputs
  - liboffload\_msg.h, 126
- msg\_c\_report\_target\_release\_buffer\_refs
  - liboffload\_msg.h, 126
- msg\_c\_report\_target\_scatter\_inputs
  - liboffload\_msg.h, 126
- msg\_c\_report\_target\_total\_time
  - liboffload\_msg.h, 126
- msg\_c\_report\_title
  - liboffload\_msg.h, 124
- msg\_c\_report\_unknown\_timer\_node
  - liboffload\_msg.h, 124
- msg\_c\_report\_unknown\_trace\_node
  - liboffload\_msg.h, 124
- msg\_c\_report\_unregister
  - liboffload\_msg.h, 125
- msg\_c\_report\_var
  - liboffload\_msg.h, 125
- msg\_c\_report\_w\_tag
  - liboffload\_msg.h, 125
- msg\_c\_report\_wait
  - liboffload\_msg.h, 125
- msg\_c\_send\_func\_ptr
  - liboffload\_msg.h, 123
- msg\_c\_slice\_of\_noncont\_array
  - liboffload\_msg.h, 126
- msg\_c\_unknown\_binary\_type
  - liboffload\_msg.h, 124
- msg\_c\_unknown\_var\_type
  - liboffload\_msg.h, 123
- msg\_c\_zero\_or\_neg\_ptr\_len
  - liboffload\_msg.h, 124
- msg\_c\_zero\_or\_neg\_transfer\_size

- liboffload\_msg.h, 124
- Mult
  - DimDesc, 25
- mutex\_locker\_t, 52
  - ~mutex\_locker\_t, 52
  - m\_mutex, 52
  - mutex\_locker\_t, 52
  - mutex\_locker\_t, 52
- mutex\_t, 52
  - ~mutex\_t, 53
  - lock, 53
  - m\_lock, 53
  - mutex\_t, 53
  - mutex\_t, 53
  - unlock, 53
- my\_tag
  - ORSL, 15
- myo\_is\_available
  - offload\_myo\_host.cpp, 156
- myo\_wrapper
  - offload\_myo\_host.cpp, 156
- MyoTable, 54
  - MyoTable, 54
  - MyoTable, 54
  - var\_tab, 54
  - var\_tab\_len, 54
- MyoTableList
  - offload\_myo\_host.cpp, 154
- MyoWrapper, 54
  - Acquire, 55
  - CheckResult, 55
  - GetResult, 55
  - HostFptrTableRegister, 55
  - HostVarTablePropagate, 55
  - is\_available, 55
  - LibFini, 55
  - LibInit, 56
  - LoadLibrary, 56
  - m\_acquire, 57
  - m\_get\_result, 57
  - m\_host\_fptr\_table\_register, 57
  - m\_host\_var\_table\_propagate, 57
  - m\_is\_available, 57
  - m\_lib\_fini, 57
  - m\_lib\_handle, 57
  - m\_lib\_init, 57
  - m\_release, 57
  - m\_remote\_call, 57
  - m\_remote\_thunk\_call, 57
  - m\_shared\_aligned\_free, 57
  - m\_shared\_aligned\_malloc, 57
  - m\_shared\_free, 58
  - m\_shared\_malloc, 58
  - MyoWrapper, 55
  - MyoWrapper, 55
  - Release, 56
  - RemoteCall, 56
  - RemoteThunkCall, 56

- SharedAlignedFree, 56
- SharedAlignedMalloc, 56
- SharedFree, 56
- SharedMalloc, 56
- UnloadLibrary, 56
- name
  - FuncTable::Entry, 31
  - TargetImage, 83
  - VarList::BufEntry, 21
  - VarTable::Entry, 32
- new\_node
  - VarList::Iterator, 38
- next
  - TableList::Node, 58
- nullify\_target\_stack
  - OffloadDescriptor, 64
- OFFLOAD\_DISABLED
  - offload.h, 130
- OFFLOAD\_ERROR
  - offload.h, 130
- OFFLOAD\_OUT\_OF\_MEMORY
  - offload.h, 130
- OFFLOAD\_PROCESS\_DIED
  - offload.h, 130
- OFFLOAD\_SUCCESS
  - offload.h, 130
- OFFLOAD\_UNAVAILABLE
  - offload.h, 130
- OFFLOAD
  - offload.common.h, 137
- OFFLOAD\_DEBUG\_LOG
  - offload.common.h, 136
- OFFLOAD\_DO\_TRACE
  - offload.common.h, 136
- OFFLOAD\_FREE
  - offload.common.h, 136
- OFFLOAD\_MALLOC
  - offload.common.cpp, 134
  - offload.common.h, 136, 138
- OFFLOAD\_OFFLOAD
  - compiler\_if\_host.cpp, 106
  - compiler\_if\_host.h, 108, 109
- OFFLOAD\_OFFLOAD1
  - compiler\_if\_host.cpp, 106
  - compiler\_if\_host.h, 108, 109
- OFFLOAD\_OFFLOAD2
  - compiler\_if\_host.cpp, 106
  - compiler\_if\_host.h, 108, 109
- OFFLOAD\_PREFIX
  - offload.common.h, 136
- OFFLOAD\_STATUS\_INIT
  - offload.h, 129
- OFFLOAD\_TIMER\_INIT
  - offload.timer.h, 184
- OFFLOAD\_TIMER\_STOP
  - offload.timer.h, 184
- OFFLOAD\_TRACE
  - offload.common.h, 136
- ORSL, 14
  - init, 15
  - is\_enabled, 15
  - my\_tag, 15
  - release, 15
  - reserve, 15
  - try\_reserve, 15
- ORSL\_MAX\_CARDS
  - orl-lite.h, 193
- ORSL\_MAX\_TAG\_LEN
  - orl-lite.h, 193
- ORSLBusySet, 76
  - orl-lite.h, 193
  - type, 76
- ORSLBusySetType
  - orl-lite.h, 194
- ORSLPartialGranularity
  - orl-lite.h, 193, 194
- ORSLRelease
  - orl-lite.h, 194
- ORSLRelease0
  - orl-lite.c, 197, 198
- ORSLReserve
  - orl-lite.h, 194
- ORSLReserve0
  - orl-lite.c, 197, 198
- ORSLReservePartial
  - orl-lite.h, 195
- ORSLReservePartial0
  - orl-lite.c, 197, 198
- ORSLTag
  - orl-lite.h, 193
- ORSLTryReserve
  - orl-lite.h, 195
- ORSLTryReserve0
  - orl-lite.c, 197, 198
- offload
  - OffloadDescriptor, 64
- offload.h
  - OFFLOAD\_DISABLED, 130
  - OFFLOAD\_ERROR, 130
  - OFFLOAD\_OUT\_OF\_MEMORY, 130
  - OFFLOAD\_PROCESS\_DIED, 130
  - OFFLOAD\_SUCCESS, 130
  - OFFLOAD\_UNAVAILABLE, 130
  - TARGET\_HOST, 130
  - TARGET\_MIC, 130
  - TARGET\_NONE, 130
- offload.h, 127
  - \_Offload\_get\_device\_number, 130
  - \_Offload\_get\_physical\_device\_number, 130
  - \_Offload\_number\_of\_devices, 130
  - \_Offload\_report, 130
  - \_Offload\_result, 130
  - \_Offload\_shared\_aligned\_free, 130
  - \_Offload\_shared\_aligned\_malloc, 130
  - \_Offload\_shared\_free, 130

- `_Offload_shared_malloc`, 130
- `_Offload_signaled`, 131
- `DEFAULT_TARGET_TYPE`, 129
- `kmp_create_affinity_mask_target`, 131
- `kmp_destroy_affinity_mask_target`, 131
- `kmp_get_affinity_mask_proc_target`, 131
- `kmp_get_affinity_max_proc_target`, 131
- `kmp_get_affinity_target`, 131
- `kmp_get_blocktime_target`, 131
- `kmp_get_library_target`, 131
- `kmp_get_stacksize_s_target`, 131
- `kmp_get_stacksize_target`, 131
- `kmp_set_affinity_mask_proc_target`, 131
- `kmp_set_affinity_target`, 131
- `kmp_set_blocktime_target`, 131
- `kmp_set_defaults_target`, 132
- `kmp_set_library_serial_target`, 132
- `kmp_set_library_target`, 132
- `kmp_set_library_throughput_target`, 132
- `kmp_set_library_turnaround_target`, 132
- `kmp_set_stacksize_s_target`, 132
- `kmp_set_stacksize_target`, 132
- `kmp_unset_affinity_mask_proc_target`, 132
- `OFFLOAD_STATUS_INIT`, 129
- `omp_destroy_lock_target`, 132
- `omp_destroy_nest_lock_target`, 132
- `omp_get_default_device`, 132
- `omp_get_dynamic_target`, 132
- `omp_get_max_threads_target`, 132
- `omp_get_nested_target`, 132
- `omp_get_num_devices`, 132
- `omp_get_num_procs_target`, 133
- `omp_get_schedule_target`, 133
- `omp_init_lock_target`, 133
- `omp_init_nest_lock_target`, 133
- `omp_set_default_device`, 133
- `omp_set_dynamic_target`, 133
- `omp_set_lock_target`, 133
- `omp_set_nest_lock_target`, 133
- `omp_set_nested_target`, 133
- `omp_set_num_threads_target`, 133
- `omp_set_schedule_target`, 133
- `omp_test_lock_target`, 133
- `omp_test_nest_lock_target`, 133
- `omp_unset_lock_target`, 133
- `omp_unset_nest_lock_target`, 134
- `TARGET_ATTRIBUTE`, 129
- `TARGET_TYPE`, 130
- `offload_common.h`
  - `c_cean_var`, 137
  - `c_cean_var_ptr`, 137
  - `c_data`, 137
  - `c_data_ptr`, 137
  - `c_data_ptr_array`, 137
  - `c_dv`, 137
  - `c_dv_data`, 137
  - `c_dv_data_slice`, 137
  - `c_dv_ptr`, 137
  - `c_dv_ptr_data`, 137
  - `c_dv_ptr_data_slice`, 137
  - `c_func_ptr`, 137
  - `c_func_ptr_array`, 138
  - `c_parameter_in`, 138
  - `c_parameter_inout`, 138
  - `c_parameter_nocopy`, 138
  - `c_parameter_out`, 138
  - `c_parameter_unknown`, 138
  - `c_string_ptr`, 137
  - `c_string_ptr_array`, 138
  - `c_void_ptr`, 137
  - `c_void_ptr_array`, 138
- `offload_env.h`
  - `c_mic_card_env`, 142
  - `c_mic_card_var`, 142
  - `c_mic_var`, 142
  - `c_no_mic`, 142
- `offload_host.h`
  - `c_init_on_offload`, 151
  - `c_init_on_offload_all`, 151
  - `c_init_on_start`, 151
- `offload_trace.h`
  - `c_offload_compute`, 187
  - `c_offload_copyin_data`, 187
  - `c_offload_copyout_data`, 187
  - `c_offload_create_buf_host`, 187
  - `c_offload_create_buf_mic`, 187
  - `c_offload_destroy`, 187
  - `c_offload_finish`, 187
  - `c_offload_gather_copyin_data`, 187
  - `c_offload_gather_copyout_data`, 187
  - `c_offload_init`, 187
  - `c_offload_init_func`, 187
  - `c_offload_mic_myo_fptr`, 187
  - `c_offload_mic_myo_shared`, 187
  - `c_offload_myo_acquire`, 187
  - `c_offload_myofini`, 187
  - `c_offload_myoinit`, 187
  - `c_offload_myoregister`, 187
  - `c_offload_myorelease`, 187
  - `c_offload_myosharedalignedfree`, 187
  - `c_offload_myosharedalignedmalloc`, 187
  - `c_offload_myosharedfree`, 187
  - `c_offload_myosharedmalloc`, 187
  - `c_offload_receive_pointer_data`, 187
  - `c_offload_received_pointer_data`, 187
  - `c_offload_register`, 187
  - `c_offload_scatter_copyin_data`, 187
  - `c_offload_scatter_copyout_data`, 187
  - `c_offload_send_pointer_data`, 187
  - `c_offload_sent_pointer_data`, 187
  - `c_offload_signal`, 187
  - `c_offload_start`, 187
  - `c_offload_start_target_func`, 187
  - `c_offload_unregister`, 187
  - `c_offload_var`, 187
  - `c_offload_wait`, 187



- offload\_active\_wait\_envname
  - offload\_host.cpp, 148
- offload\_call\_count
  - compiler\_if\_host.cpp, 107
- offload\_common.cpp, 134
  - OFFLOAD\_MALLOC, 134
- offload\_common.h, 134
  - console\_enabled, 138
  - flag\_align\_is\_array, 138
  - flag\_alloc\_elements\_is\_array, 138
  - flag\_alloc\_elements\_is\_scalar, 138
  - flag\_alloc\_if\_is\_array, 138
  - flag\_alloc\_start\_is\_array, 138
  - flag\_alloc\_start\_is\_scalar, 138
  - flag\_extent\_elements\_is\_array, 139
  - flag\_extent\_elements\_is\_scalar, 139
  - flag\_extent\_start\_is\_array, 139
  - flag\_extent\_start\_is\_scalar, 139
  - flag\_free\_if\_is\_array, 139
  - flag\_into\_elements\_is\_array, 139
  - flag\_into\_elements\_is\_scalar, 139
  - flag\_into\_start\_is\_array, 139
  - flag\_into\_start\_is\_scalar, 139
  - mic\_index, 139
  - OFFLOAD, 137
  - OFFLOAD\_DEBUG\_LOG, 136
  - OFFLOAD\_DO\_TRACE, 136
  - OFFLOAD\_FREE, 136
  - OFFLOAD\_MALLOC, 136, 138
  - OFFLOAD\_PREFIX, 136
  - OFFLOAD\_TRACE, 136
  - offload\_number, 139
  - offload\_report\_level, 139
  - OffloadItemType, 137
  - OffloadParameterType, 138
  - prefix, 140
  - VAR\_TYPE\_IS\_PTR, 137
- offload\_engine.cpp, 140
  - host\_entry\_cmp, 140
  - target\_entry\_cmp, 140
- offload\_engine.h, 140
  - AutoSet, 141
  - check\_result, 141
  - PersistDataList, 141
  - PtrDataList, 141
  - TargetImageList, 141
- offload\_env.cpp, 141
- offload\_env.h, 142
  - MicEnvVarKind, 142
- offload\_fini
  - ofldbbegin.cpp, 191
- offload\_finish
  - OffloadDescriptor, 64
- offload\_func\_with\_parms
  - offload\_target.cpp, 181
- offload\_get\_device\_number
  - mic\_lib::offload\_get\_device\_number, 59
- offload\_get\_message\_str
  - liboffload\_error\_codes.h, 121
  - liboffload\_msg.c, 121
- offload\_get\_physical\_device\_number
  - mic\_lib::offload\_get\_physical\_device\_number, 59
- offload\_get\_src\_base
  - offload\_host.cpp, 145
- offload\_host.cpp, 142
  - \_Offload\_get\_device\_number, 144
  - \_Offload\_get\_physical\_device\_number, 144
  - \_Offload\_number\_of\_devices, 145
  - \_Offload\_report, 145
  - \_Offload\_signaled, 145
  - \_dbg\_api\_major\_version, 145
  - \_dbg\_api\_minor\_version, 145
  - \_dbg\_is\_attached, 145
  - \_dbg\_target\_exe\_name, 145
  - \_dbg\_target\_id, 145
  - \_dbg\_target\_so\_loaded, 144
  - \_dbg\_target\_so\_pid, 145
  - \_dbg\_target\_so\_unloaded, 144
  - \_offload\_active\_wait, 146
  - \_offload\_console\_trace, 144
  - \_offload\_fini\_library, 144
  - \_offload\_init\_library, 144
  - \_offload\_init\_library\_once, 144
  - \_offload\_init\_type, 146
  - \_offload\_register\_image, 144
  - \_offload\_unregister\_image, 144
  - \_offload\_use\_2mb\_buffers, 146
  - \_offload\_use\_async\_buffer\_read, 146
  - \_offload\_use\_async\_buffer\_write, 146
  - \_omp\_device\_num, 146
  - \_target\_exe, 146
  - \_target\_libs, 146
  - \_target\_libs\_list, 146
  - \_target\_libs\_lock, 146
- console\_enabled, 146
- cpu\_frequency, 146
- get\_arr\_desc\_numbers, 145
- htrace\_envname, 147
- make\_arr\_desc, 145
- mic\_buffer\_size, 147
- mic\_engines, 147
- mic\_engines\_total, 147
- mic\_env\_vars, 147
- mic\_library\_path, 147
- mic\_proxy\_fs\_root, 147
- mic\_proxy\_io, 147
- mic\_stack\_size, 147
- mic\_thread\_key, 147
- mic\_use\_2mb\_buffers\_envname, 147
- mic\_use\_async\_buffer\_read\_envname, 148
- mic\_use\_async\_buffer\_write\_envname, 148
- offload\_active\_wait\_envname, 148
- offload\_get\_src\_base, 145
- offload\_init\_envname, 148
- offload\_number, 148
- offload\_report\_envname, 148



- omp\_device\_num\_envname, 148
- PATH\_SEPARATOR, 144
- prefix, 148
- stack\_alloc\_lock, 148
- timer\_envname, 148
- vardesc\_direction\_as\_string, 149
- vardesc\_type\_as\_string, 149
- offload\_host.h, 149
  - \_\_dbg\_api\_major\_version, 151
  - \_\_dbg\_api\_minor\_version, 151
  - \_\_dbg\_is\_attached, 151
  - \_\_dbg\_target\_exe\_name, 151
  - \_\_dbg\_target\_id, 151
  - \_\_dbg\_target\_so\_loaded, 151
  - \_\_dbg\_target\_so\_pid, 151
  - \_\_dbg\_target\_so\_unloaded, 151
  - \_\_offload\_init\_library, 151
  - \_\_offload\_init\_type, 152
  - \_\_offload\_register\_image, 151
  - \_\_offload\_unregister\_image, 151
  - \_\_offload\_use\_2mb\_buffers, 152
  - \_\_omp\_device\_num, 152
  - \_\_target\_exe, 152
- cpu\_frequency, 152
- MAX\_TARGET\_NAME, 150
- mic\_buffer\_size, 152
- mic\_engines, 152
- mic\_engines\_total, 152
- mic\_env\_vars, 152
- mic\_library\_path, 152
- mic\_proxy\_fs\_root, 152
- mic\_proxy\_io, 153
- mic\_stack\_size, 153
- mic\_thread\_key, 153
- OffloadInitType, 151
- offload\_init
  - ofldbbegin.cpp, 191
- offload\_init\_envname
  - offload\_host.cpp, 148
- offload\_myo\_host.cpp, 153
  - \_Offload\_shared\_aligned\_free, 155
  - \_Offload\_shared\_aligned\_malloc, 155
  - \_Offload\_shared\_free, 156
  - \_Offload\_shared\_malloc, 156
  - \_\_cilkrts\_cilk\_for\_32, 154
  - \_\_cilkrts\_cilk\_for\_64, 154
  - \_\_intel\_cilk\_for\_32\_offload, 154
  - \_\_intel\_cilk\_for\_64\_offload, 154
  - \_\_myo\_table\_list, 156
  - \_\_myo\_table\_lock, 156
  - \_\_myo\_tables, 156
  - \_\_offload\_myoFini, 155
  - \_\_offload\_myoInit, 155
  - \_\_offload\_myoInit\_once, 155
  - \_\_offload\_myoIsAvailable, 155
  - \_\_offload\_myoLoadLibrary, 155
  - \_\_offload\_myoLoadLibrary\_once, 155
  - \_\_offload\_myoRegisterTables, 155
  - \_\_offload\_myo\_fptr\_table\_register, 154
  - \_\_offload\_myo\_shared\_init\_table\_register, 155
  - \_\_offload\_myo\_shared\_table\_register, 155
  - \_\_offload\_myoRemotelThunkCall, 155
  - fptr\_table\_entries, 156
  - MYO\_VERSION1, 154
  - myo\_is\_available, 156
  - myo\_wrapper, 156
  - MyoTableList, 154
  - shared\_table\_entries, 156
- offload\_myo\_host.h, 156
  - \_\_offload\_myoFini, 158
  - \_\_offload\_myoRegisterTables, 158
  - SharedTableEntry, 157
- offload\_myo\_target.cpp, 158
  - \_Offload\_shared\_aligned\_free, 159
  - \_Offload\_shared\_aligned\_malloc, 159
  - \_Offload\_shared\_free, 159
  - \_Offload\_shared\_malloc, 159
  - \_\_cilkrts\_cilk\_for\_32, 158
  - \_\_cilkrts\_cilk\_for\_64, 158
  - \_\_intel\_cilk\_for\_32\_offload\_wrapper, 158
  - \_\_intel\_cilk\_for\_64\_offload\_wrapper, 158
  - \_\_offload\_myoAcquire, 159
  - \_\_offload\_myoLibFini, 159
  - \_\_offload\_myoLibInit, 159
  - \_\_offload\_myoRegisterTables, 159
  - \_\_offload\_myoRelease, 159
  - \_\_offload\_myo\_fptr\_table\_register, 158
  - \_\_offload\_myo\_once\_init, 159
  - \_\_offload\_myo\_shared\_table\_register, 159
  - CheckResult, 159
- offload\_myo\_target.h, 160
  - \_\_offload\_myoAcquire, 161
  - \_\_offload\_myoLibFini, 161
  - \_\_offload\_myoLibInit, 161
  - \_\_offload\_myoRegisterTables, 161
  - \_\_offload\_myoRelease, 161
  - FptrTableEntry, 160
  - SharedTableEntry, 160
- offload\_number
  - FunctionDescriptor, 35
  - offload\_common.h, 139
  - offload\_host.cpp, 148
  - offload\_target.cpp, 181
- offload\_number\_of\_devices
  - mic\_lib::offload\_number\_of\_devices, 59
- offload\_offload\_wrap
  - compiler\_if\_host.cpp, 107
- offload\_omp\_host.cpp, 161
  - kmp\_create\_affinity\_mask\_target, 162
  - kmp\_destroy\_affinity\_mask\_target, 162
  - kmp\_get\_affinity\_mask\_proc\_target, 162
  - kmp\_get\_affinity\_max\_proc\_target, 162
  - kmp\_get\_affinity\_target, 162
  - kmp\_get\_blocktime\_target, 162
  - kmp\_get\_library\_target, 162
  - kmp\_get\_stacksize\_s\_target, 163

- kmp\_get\_stacksize\_target, 163
- kmp\_set\_affinity\_mask\_proc\_target, 163
- kmp\_set\_affinity\_target, 163
- kmp\_set\_blocktime\_target, 163
- kmp\_set\_defaults\_target, 163
- kmp\_set\_library\_serial\_target, 163
- kmp\_set\_library\_target, 163
- kmp\_set\_library\_throughput\_target, 163
- kmp\_set\_library\_turnaround\_target, 163
- kmp\_set\_stacksize\_s\_target, 163
- kmp\_set\_stacksize\_target, 163
- kmp\_unset\_affinity\_mask\_proc\_target, 163
- omp\_destroy\_lock\_target, 163
- omp\_destroy\_nest\_lock\_target, 164
- omp\_get\_default\_device, 164
- omp\_get\_dynamic\_target, 164
- omp\_get\_int\_target, 164
- omp\_get\_max\_threads\_target, 164
- omp\_get\_nested\_target, 164
- omp\_get\_num\_devices, 164
- omp\_get\_num\_procs\_target, 164
- omp\_get\_schedule\_target, 164
- omp\_init\_lock\_target, 164
- omp\_init\_nest\_lock\_target, 164
- omp\_set\_default\_device, 164
- omp\_set\_dynamic\_target, 164
- omp\_set\_int\_target, 165
- omp\_set\_lock\_target, 165
- omp\_set\_nest\_lock\_target, 165
- omp\_set\_nested\_target, 165
- omp\_set\_num\_threads\_target, 165
- omp\_set\_schedule\_target, 165
- omp\_test\_lock\_target, 165
- omp\_test\_nest\_lock\_target, 165
- omp\_unset\_lock\_target, 165
- omp\_unset\_nest\_lock\_target, 165
- offload\_omp\_target.cpp, 165
  - kmp\_create\_affinity\_mask\_lrb, 167
  - kmp\_create\_affinity\_mask\_target, 167
  - kmp\_destroy\_affinity\_mask\_lrb, 167
  - kmp\_destroy\_affinity\_mask\_target, 167
  - kmp\_get\_affinity\_lrb, 168
  - kmp\_get\_affinity\_mask\_proc\_lrb, 168
  - kmp\_get\_affinity\_mask\_proc\_target, 168
  - kmp\_get\_affinity\_max\_proc\_lrb, 168
  - kmp\_get\_affinity\_max\_proc\_target, 168
  - kmp\_get\_affinity\_target, 168
  - kmp\_get\_blocktime\_lrb, 168
  - kmp\_get\_blocktime\_target, 168
  - kmp\_get\_library\_lrb, 168
  - kmp\_get\_library\_target, 168
  - kmp\_get\_stacksize\_lrb, 168
  - kmp\_get\_stacksize\_s\_lrb, 168
  - kmp\_get\_stacksize\_s\_target, 168
  - kmp\_get\_stacksize\_target, 168
  - kmp\_set\_affinity\_lrb, 169
  - kmp\_set\_affinity\_mask\_proc\_lrb, 169
  - kmp\_set\_affinity\_mask\_proc\_target, 169
  - kmp\_set\_affinity\_target, 169
  - kmp\_set\_blocktime\_lrb, 169
  - kmp\_set\_blocktime\_target, 169
  - kmp\_set\_defaults\_lrb, 169
  - kmp\_set\_defaults\_target, 169
  - kmp\_set\_library\_lrb, 169
  - kmp\_set\_library\_serial\_lrb, 169
  - kmp\_set\_library\_serial\_target, 169
  - kmp\_set\_library\_target, 169
  - kmp\_set\_library\_throughput\_lrb, 169
  - kmp\_set\_library\_throughput\_target, 169
  - kmp\_set\_library\_turnaround\_lrb, 169
  - kmp\_set\_library\_turnaround\_target, 170
  - kmp\_set\_stacksize\_lrb, 170
  - kmp\_set\_stacksize\_s\_lrb, 170
  - kmp\_set\_stacksize\_s\_target, 170
  - kmp\_set\_stacksize\_target, 170
  - kmp\_unset\_affinity\_mask\_proc\_lrb, 170
  - kmp\_unset\_affinity\_mask\_proc\_target, 170
  - omp\_destroy\_lock\_lrb, 170
  - omp\_destroy\_lock\_target, 170
  - omp\_destroy\_nest\_lock\_lrb, 170
  - omp\_destroy\_nest\_lock\_target, 170
  - omp\_get\_default\_device, 170
  - omp\_get\_dynamic\_lrb, 170
  - omp\_get\_dynamic\_target, 170
  - omp\_get\_int\_from\_host, 170
  - omp\_get\_max\_threads\_lrb, 171
  - omp\_get\_max\_threads\_target, 171
  - omp\_get\_nested\_lrb, 171
  - omp\_get\_nested\_target, 171
  - omp\_get\_num\_devices, 171
  - omp\_get\_num\_procs\_lrb, 171
  - omp\_get\_num\_procs\_target, 171
  - omp\_get\_schedule\_lrb, 171
  - omp\_get\_schedule\_target, 171
  - omp\_init\_lock\_lrb, 171
  - omp\_init\_lock\_target, 171
  - omp\_init\_nest\_lock\_lrb, 171
  - omp\_init\_nest\_lock\_target, 171
  - omp\_send\_int\_to\_host, 171
  - omp\_set\_default\_device, 172
  - omp\_set\_dynamic\_lrb, 172
  - omp\_set\_dynamic\_target, 172
  - omp\_set\_lock\_lrb, 172
  - omp\_set\_lock\_target, 172
  - omp\_set\_nest\_lock\_lrb, 172
  - omp\_set\_nest\_lock\_target, 172
  - omp\_set\_nested\_lrb, 172
  - omp\_set\_nested\_target, 172
  - omp\_set\_num\_threads\_lrb, 172
  - omp\_set\_num\_threads\_target, 172
  - omp\_set\_schedule\_lrb, 172
  - omp\_set\_schedule\_target, 172
  - omp\_test\_lock\_lrb, 172
  - omp\_test\_lock\_target, 173
  - omp\_test\_nest\_lock\_lrb, 173
  - omp\_test\_nest\_lock\_target, 173

- omp\_unset\_lock\_lrb, 173
- omp\_unset\_lock\_target, 173
- omp\_unset\_nest\_lock\_lrb, 173
- omp\_unset\_nest\_lock\_target, 173
- offload\_orisl.cpp, 173
- offload\_orisl.h, 174
- offload\_report
  - mic.lib::offload\_report, 60
- offload\_report\_envname
  - offload\_host.cpp, 148
- offload\_report\_level
  - FunctionDescriptor, 35
  - offload\_common.h, 139
  - offload\_target.cpp, 181
- offload\_signal
  - offload\_trace.cpp, 186
- offload\_signaled
  - mic.lib::offload\_signaled, 60
- offload\_stack\_memory\_manager
  - OffloadDescriptor, 64
- offload\_stage
  - offload\_trace.cpp, 186
- offload\_stage\_print
  - offload\_trace.cpp, 186
  - offload\_trace.h, 187
- offload\_table.cpp, 174
  - \_\_offload\_funcs, 178
  - \_\_offload\_register\_tables, 175
  - \_\_offload\_unregister\_tables, 175
  - \_\_offload\_vars, 178
  - kmp\_create\_affinity\_mask\_lrb, 175
  - kmp\_destroy\_affinity\_mask\_lrb, 175
  - kmp\_get\_affinity\_lrb, 175
  - kmp\_get\_affinity\_mask\_proc\_lrb, 175
  - kmp\_get\_affinity\_max\_proc\_lrb, 175
  - kmp\_get\_blocktime\_lrb, 175
  - kmp\_get\_library\_lrb, 175
  - kmp\_get\_stacksize\_lrb, 175
  - kmp\_get\_stacksize\_s\_lrb, 176
  - kmp\_set\_affinity\_lrb, 176
  - kmp\_set\_affinity\_mask\_proc\_lrb, 176
  - kmp\_set\_blocktime\_lrb, 176
  - kmp\_set\_defaults\_lrb, 176
  - kmp\_set\_library\_lrb, 176
  - kmp\_set\_library\_serial\_lrb, 176
  - kmp\_set\_library\_throughput\_lrb, 176
  - kmp\_set\_library\_turnaround\_lrb, 176
  - kmp\_set\_stacksize\_lrb, 176
  - kmp\_set\_stacksize\_s\_lrb, 176
  - kmp\_unset\_affinity\_mask\_proc\_lrb, 176
  - omp\_destroy\_lock\_lrb, 176
  - omp\_destroy\_nest\_lock\_lrb, 176
  - omp\_get\_dynamic\_lrb, 176
  - omp\_get\_max\_threads\_lrb, 177
  - omp\_get\_nested\_lrb, 177
  - omp\_get\_num\_procs\_lrb, 177
  - omp\_get\_schedule\_lrb, 177
  - omp\_init\_lock\_lrb, 177
  - omp\_init\_nest\_lock\_lrb, 177
  - omp\_set\_dynamic\_lrb, 177
  - omp\_set\_lock\_lrb, 177
  - omp\_set\_nest\_lock\_lrb, 177
  - omp\_set\_nested\_lrb, 177
  - omp\_set\_num\_threads\_lrb, 177
  - omp\_set\_schedule\_lrb, 177
  - omp\_test\_lock\_lrb, 177
  - omp\_test\_nest\_lock\_lrb, 177
  - omp\_unset\_lock\_lrb, 177
  - omp\_unset\_nest\_lock\_lrb, 178
  - predefined\_entries, 178
  - predefined\_table, 178
- offload\_table.h, 178
  - \_\_offload\_entries, 180
  - \_\_offload\_funcs, 180
  - \_\_offload\_register\_tables, 179
  - \_\_offload\_unregister\_tables, 179
  - \_\_offload\_vars, 180
- offload\_target.cpp, 180
  - \_Offload\_get\_device\_number, 181
  - \_Offload\_get\_physical\_device\_number, 181
  - \_Offload\_number\_of\_devices, 181
  - \_\_offload\_target\_init, 181
  - add\_ref\_lock, 181
  - console\_enabled, 181
  - mic\_engines\_total, 181
  - mic\_frequency, 181
  - mic\_index, 181
  - offload\_func\_with\_parms, 181
  - offload\_number, 181
  - offload\_report\_level, 181
  - prefix, 181
  - ref\_data, 182
  - vardesc\_direction\_as\_string, 182
  - vardesc\_type\_as\_string, 182
- offload\_target.h, 182
  - \_\_offload\_target\_init, 183
  - mic\_engines\_total, 183
  - mic\_frequency, 183
  - mic\_index, 183
- offload\_target\_main.cpp, 183
  - \_\_offload\_target\_main, 183
  - main, 183
- offload\_timer.h, 183
  - timer\_enabled, 184
- offload\_timer\_host.cpp, 185
  - timer\_enabled, 185
- offload\_timer\_target.cpp, 185
  - timer\_enabled, 185
- offload\_trace.cpp, 185
  - mic\_index, 186
  - offload\_signal, 186
  - offload\_stage, 186
  - offload\_stage\_print, 186
  - prefix, 186
- offload\_trace.h, 186
  - offload\_stage\_print, 187

- OffloadTraceStage, 187
- offload\_util.cpp, 188
  - \_\_offload\_parse\_int\_string, 188
  - \_\_offload\_parse\_size\_string, 188
  - DL\_sym, 188
  - get\_el\_value, 188
- offload\_util.h, 188
  - \_\_offload\_parse\_int\_string, 190
  - \_\_offload\_parse\_size\_string, 190
  - \_\_offload\_run\_once, 189
  - DL\_addr, 189
  - DL\_close, 189
  - DL\_open, 189
  - DL\_sym, 190
  - get\_el\_value, 190
  - OffloadOnceControl, 190
  - thread\_getspecific, 189
  - thread\_key\_create, 189
  - thread\_key\_delete, 189
  - thread\_setspecific, 189
- OffloadDescriptor, 61
  - ~OffloadDescriptor, 63
  - alloc\_ptr\_data, 63
  - BufferList, 62
  - cleanup, 63
  - compute, 63
  - find\_ptr\_data, 63
  - gather\_copyin\_data, 63
  - gather\_copyout\_data, 63
  - gen\_var\_descs\_for\_pointer\_array, 63
  - get\_offload\_number, 63
  - get\_timer\_data, 64
  - init\_mic\_address, 64
  - init\_static\_ptr\_data, 64
  - is\_signaled, 64
  - m\_buffers, 66
  - m\_compute\_buffers, 66
  - m\_destroy\_buffers, 66
  - m\_destroy\_stack, 66
  - m\_device, 66
  - m\_func\_desc, 66
  - m\_func\_desc\_size, 66
  - m\_in, 66
  - m\_in\_datalen, 66
  - m\_in\_deps, 67
  - m\_in\_deps\_total, 67
  - m\_inout\_buf, 67
  - m\_is\_mandatory, 67
  - m\_is\_openmp, 67
  - m\_need\_runfunction, 67
  - m\_offload\_number, 67
  - m\_out, 67
  - m\_out\_datalen, 67
  - m\_out\_deps, 67
  - m\_out\_deps\_total, 67
  - m\_stack\_ptr\_data, 68
  - m\_status, 68
  - m\_timer\_data, 68
  - m\_vars, 68
  - m\_vars\_extra, 68
  - m\_vars\_total, 68
  - merge\_var\_descs, 64
  - nullify\_target\_stack, 64
  - offload, 64
  - offload\_finish, 64
  - offload\_stack\_memory\_manager, 64
  - OffloadDescriptor, 63
  - OffloadDescriptor, 63
  - receive\_pointer\_data, 64
  - recieve\_noncontiguous\_pointer\_data, 65
  - report\_coi\_error, 65
  - scatter\_copyin\_data, 65
  - scatter\_copyout\_data, 65
  - send\_noncontiguous\_pointer\_data, 65
  - send\_pointer\_data, 65
  - set\_offload\_number, 65
  - setup\_descriptors, 65
  - setup\_misc\_data, 65
  - translate\_coi\_error, 65
  - wait\_dependencies, 66
- OffloadDescriptor::ReadArrElements
  - base, 80
  - count, 80
  - el\_size, 80
  - is\_empty, 80
  - length\_cur, 80
  - offset, 80
  - ranges, 80
  - read\_next, 80
  - ReadArrElements, 79
  - size, 80
  - val, 80
- OffloadDescriptor::ReadArrElements< T >, 79
- OffloadDescriptor::VarExtra, 95
  - auto\_data, 95
  - cpu\_disp, 95
  - cpu\_offset, 95
  - dst\_data, 95
  - is\_arr\_ptr\_el, 95
  - ptr\_arr\_offset, 95
  - read\_rng\_dst, 95
  - read\_rng\_src, 96
  - src\_data, 96
- OffloadHostPhase
  - liboffload\_error\_codes.h, 119
- OffloadInitType
  - offload\_host.h, 151
- OffloadItemType
  - offload\_common.h, 137
- OffloadOnceControl
  - offload\_util.h, 190
- OffloadParameterType
  - offload\_common.h, 138
- OffloadTargetPhase
  - liboffload\_error\_codes.h, 120
- OffloadTraceStage

- offload\_trace.h, 187
- Offset
  - ArrDesc, 19
- offset
  - OffloadDescriptor::ReadArrElements, 80
  - TargetImage, 83
  - VarDesc, 90
- ofldbegin.cpp, 190
  - \_\_offload\_entry\_node, 191
  - \_\_offload\_entry\_table\_start, 191
  - \_\_offload\_func\_node, 191
  - \_\_offload\_func\_table\_start, 191
  - \_\_offload\_var\_node, 191
  - \_\_offload\_var\_table\_start, 192
  - ALLOCATE, 191
  - DLL\_LOCAL, 191
  - MAIN\_, 191
  - main, 191
  - offload\_fini, 191
  - offload\_init, 191
- ofldend.cpp, 192
  - \_\_offload\_entry\_table\_end, 192
  - \_\_offload\_func\_table\_end, 192
  - \_\_offload\_var\_table\_end, 192
  - ALLOCATE, 192
- omp\_destroy\_lock\_lrb
  - offload\_omp\_target.cpp, 170
  - offload\_table.cpp, 176
- omp\_destroy\_lock\_target
  - mic\_lib::omp\_destroy\_lock\_target, 69
  - offload.h, 132
  - offload\_omp\_host.cpp, 163
  - offload\_omp\_target.cpp, 170
- omp\_destroy\_nest\_lock\_lrb
  - offload\_omp\_target.cpp, 170
  - offload\_table.cpp, 176
- omp\_destroy\_nest\_lock\_target
  - mic\_lib::omp\_destroy\_nest\_lock\_target, 69
  - offload.h, 132
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 170
- omp\_device\_num\_envname
  - offload\_host.cpp, 148
- omp\_get\_default\_device
  - offload.h, 132
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 170
- omp\_get\_dynamic\_lrb
  - offload\_omp\_target.cpp, 170
  - offload\_table.cpp, 176
- omp\_get\_dynamic\_target
  - mic\_lib::omp\_get\_dynamic\_target, 69
  - offload.h, 132
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 170
- omp\_get\_int\_from\_host
  - offload\_omp\_target.cpp, 170
- omp\_get\_int\_target
  - offload\_omp\_host.cpp, 164
- omp\_get\_max\_threads\_lrb
  - offload\_omp\_target.cpp, 171
  - offload\_table.cpp, 177
- omp\_get\_max\_threads\_target
  - mic\_lib::omp\_get\_max\_threads\_target, 70
  - offload.h, 132
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 171
- omp\_get\_nested\_lrb
  - offload\_omp\_target.cpp, 171
  - offload\_table.cpp, 177
- omp\_get\_nested\_target
  - mic\_lib::omp\_get\_nested\_target, 70
  - offload.h, 132
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 171
- omp\_get\_num\_devices
  - offload.h, 132
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 171
- omp\_get\_num\_procs\_lrb
  - offload\_omp\_target.cpp, 171
  - offload\_table.cpp, 177
- omp\_get\_num\_procs\_target
  - mic\_lib::omp\_get\_num\_procs\_target, 70
  - offload.h, 133
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 171
- omp\_get\_schedule\_lrb
  - offload\_omp\_target.cpp, 171
  - offload\_table.cpp, 177
- omp\_get\_schedule\_target
  - mic\_lib::omp\_get\_schedule\_target, 71
  - offload.h, 133
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 171
- omp\_init\_lock\_lrb
  - offload\_omp\_target.cpp, 171
  - offload\_table.cpp, 177
- omp\_init\_lock\_target
  - mic\_lib::omp\_init\_lock\_target, 71
  - offload.h, 133
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 171
- omp\_init\_nest\_lock\_lrb
  - offload\_omp\_target.cpp, 171
  - offload\_table.cpp, 177
- omp\_init\_nest\_lock\_target
  - mic\_lib::omp\_init\_nest\_lock\_target, 71
  - offload.h, 133
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 171
- omp\_lock\_target\_t, 71
  - lock, 72
- omp\_nest\_lock\_target\_t, 72
  - lock, 72
- omp\_send\_int\_to\_host

- offload\_omp\_target.cpp, 171
- omp\_set\_default\_device
  - offload.h, 133
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 172
- omp\_set\_dynamic\_lrb
  - offload\_omp\_target.cpp, 172
  - offload\_table.cpp, 177
- omp\_set\_dynamic\_target
  - mic\_lib::omp\_set\_dynamic\_target, 72
  - offload.h, 133
  - offload\_omp\_host.cpp, 164
  - offload\_omp\_target.cpp, 172
- omp\_set\_int\_target
  - offload\_omp\_host.cpp, 165
- omp\_set\_lock\_lrb
  - offload\_omp\_target.cpp, 172
  - offload\_table.cpp, 177
- omp\_set\_lock\_target
  - mic\_lib::omp\_set\_lock\_target, 73
  - offload.h, 133
  - offload\_omp\_host.cpp, 165
  - offload\_omp\_target.cpp, 172
- omp\_set\_nest\_lock\_lrb
  - offload\_omp\_target.cpp, 172
  - offload\_table.cpp, 177
- omp\_set\_nest\_lock\_target
  - mic\_lib::omp\_set\_nest\_lock\_target, 73
  - offload.h, 133
  - offload\_omp\_host.cpp, 165
  - offload\_omp\_target.cpp, 172
- omp\_set\_nested\_lrb
  - offload\_omp\_target.cpp, 172
  - offload\_table.cpp, 177
- omp\_set\_nested\_target
  - mic\_lib::omp\_set\_nested\_target, 73
  - offload.h, 133
  - offload\_omp\_host.cpp, 165
  - offload\_omp\_target.cpp, 172
- omp\_set\_num\_threads\_lrb
  - offload\_omp\_target.cpp, 172
  - offload\_table.cpp, 177
- omp\_set\_num\_threads\_target
  - mic\_lib::omp\_set\_num\_threads\_target, 74
  - offload.h, 133
  - offload\_omp\_host.cpp, 165
  - offload\_omp\_target.cpp, 172
- omp\_set\_schedule\_lrb
  - offload\_omp\_target.cpp, 172
  - offload\_table.cpp, 177
- omp\_set\_schedule\_target
  - mic\_lib::omp\_set\_schedule\_target, 74
  - offload.h, 133
  - offload\_omp\_host.cpp, 165
  - offload\_omp\_target.cpp, 172
- omp\_test\_lock\_lrb
  - offload\_omp\_target.cpp, 172
  - offload\_table.cpp, 177
- omp\_test\_lock\_target
  - mic\_lib::omp\_test\_lock\_target, 74
  - offload.h, 133
  - offload\_omp\_host.cpp, 165
  - offload\_omp\_target.cpp, 173
- omp\_test\_nest\_lock\_lrb
  - offload\_omp\_target.cpp, 173
  - offload\_table.cpp, 177
- omp\_test\_nest\_lock\_target
  - mic\_lib::omp\_test\_nest\_lock\_target, 75
  - offload.h, 133
  - offload\_omp\_host.cpp, 165
  - offload\_omp\_target.cpp, 173
- omp\_unset\_lock\_lrb
  - offload\_omp\_target.cpp, 173
  - offload\_table.cpp, 177
- omp\_unset\_lock\_target
  - mic\_lib::omp\_unset\_lock\_target, 75
  - offload.h, 133
  - offload\_omp\_host.cpp, 165
  - offload\_omp\_target.cpp, 173
- omp\_unset\_nest\_lock\_lrb
  - offload\_omp\_target.cpp, 173
  - offload\_table.cpp, 178
- omp\_unset\_nest\_lock\_target
  - mic\_lib::omp\_unset\_nest\_lock\_target, 75
  - offload.h, 134
  - offload\_omp\_host.cpp, 165
  - offload\_omp\_target.cpp, 173
- operator<
  - AutoData, 20
  - PtrData, 78
- operator\*
  - VarList::Iterator, 38
- operator++
  - VarList::Iterator, 38
- operator==
  - VarList::Iterator, 38
- origin
  - TargetImage, 83
- orl-lite.h
  - BUSY\_SET\_EMPTY, 194
  - BUSY\_SET\_FULL, 194
  - BUSY\_SET\_PARTIAL, 194
  - GRAN\_CARD, 194
  - GRAN\_THREAD, 194
- orl-lite.c
  - can\_release\_card, 197
  - can\_reserve\_card, 197
  - check\_args, 197
  - check\_bsets, 198
  - ORSLRelease0, 197, 198
  - ORSLReserve0, 197, 198
  - ORSLReservePartial0, 197, 198
  - ORSLTryReserve0, 197, 198
  - owner, 199
  - release\_card, 198
  - reserve\_card, 198

- rsrv\_cnt, 199
- rsrv\_data, 199
- state\_lock, 198
- state\_signal\_release, 198
- state\_unlock, 198
- state\_wait\_for\_release, 198
- orl-lite.h
  - BusySetType, 193
  - ORSL\_MAX\_CARDS, 193
  - ORSL\_MAX\_TAG\_LEN, 193
  - ORSLBusySet, 193
  - ORSLBusySetType, 194
  - ORSLPartialGranularity, 193, 194
  - ORSLRelease, 194
  - ORSLReserve, 194
  - ORSLReservePartial, 195
  - ORSLTag, 193
  - ORSLTryReserve, 195
- orl-lite/include/orl-lite.h, 192
- orl-lite/lib/orl-lite.c, 196
- out
  - VarDesc, 90
- out\_data\_len
  - FunctionDescriptor, 35
- overlaps
  - MemRange, 48
- owner
  - orl-lite.c, 199
- PATH\_SEPARATOR
  - offload\_host.cpp, 144
- pArrDesc
  - dv\_util.h, 113
- PerfGetCycleFrequency
  - COI, 13
- PersistData, 76
  - cpu\_stack\_addr, 76
  - PersistData, 76
  - PersistData, 76
  - routine\_id, 76
  - stack\_cpu\_addr, 76
  - stack\_ptr\_data, 77
- PersistDataList
  - offload\_engine.h, 141
- PipelineCreate
  - COI, 13
- PipelineDestroy
  - COI, 14
- PipelineRunFunction
  - COI, 14
- PipelineStartExecutingRunFunctions
  - coi\_server.h, 105
- predefined\_entries
  - offload\_table.cpp, 178
- predefined\_table
  - offload\_table.cpp, 178
- prefix
  - MicEnvVar, 52
  - offload\_common.h, 140
  - offload\_host.cpp, 148
  - offload\_target.cpp, 181
  - offload\_trace.cpp, 186
- prev
  - TableList::Node, 58
- ProcessCreateFromMemory
  - COI, 14
- ProcessDestroy
  - COI, 14
- ProcessGetFunctionHandles
  - COI, 14
- ProcessLoadLibraryFromMemory
  - COI, 14
- ProcessRegisterLibraries
  - COI, 14
- ProcessWaitForShutdown
  - coi\_server.h, 105
- ptr
  - CeanReadRanges, 23
  - VarDesc, 90
- ptr\_arr\_offset
  - OffloadDescriptor::VarExtra, 95
  - VarDesc, 90
- ptr\_array
  - VarDesc3, 94
- PtrData, 77
  - add\_reference, 78
  - alloc\_disp, 78
  - alloc\_ptr\_data.lock, 78
  - cpu\_addr, 78
  - cpu\_buf, 78
  - get\_reference, 78
  - is\_static, 78
  - mic\_addr, 78
  - mic\_buf, 78
  - mic\_offset, 79
  - operator<, 78
  - PtrData, 77
  - PtrData, 77
  - ref\_count, 79
  - remove\_reference, 78
- PtrDataList
  - offload\_engine.h, 141
- PtrSet
  - Engine, 26
- range\_max\_number
  - CeanReadRanges, 23
- range\_size
  - CeanReadRanges, 23
- ranges
  - OffloadDescriptor::ReadArrElements, 80
- Rank
  - ArrDesc, 19
- rank
  - arr\_desc, 18
- read\_next
  - OffloadDescriptor::ReadArrElements, 80
- read\_rng\_dst



- OffloadDescriptor::VarExtra, 95
- read\_rng\_src
  - OffloadDescriptor::VarExtra, 96
- ReadArrElements
  - OffloadDescriptor::ReadArrElements, 79
- receive\_data
  - Marshaller, 46
- receive\_func\_ptr
  - Marshaller, 46
- receive\_pointer\_data
  - OffloadDescriptor, 64
- recieve\_noncontiguous\_pointer\_data
  - OffloadDescriptor, 65
- ref\_count
  - AutoData, 20
  - PtrData, 79
- ref\_data
  - offload\_target.cpp, 182
- RefInfo, 81
  - count, 81
  - is\_added, 81
  - RefInfo, 81
  - RefInfo, 81
- Release
  - MyoWrapper, 56
- release
  - ORSL, 15
- release\_card
  - orl-lite.c, 198
- RemoteCall
  - MyoWrapper, 56
- RemoteThunkCall
  - MyoWrapper, 56
- remove\_auto\_data
  - Engine, 28
- remove\_ptr\_data
  - Engine, 29
- remove\_reference
  - AutoData, 20
  - PtrData, 78
- remove\_table
  - TableList, 82
- report\_coi\_error
  - OffloadDescriptor, 65
- report\_get\_host\_stage\_str
  - liboffload\_error.c, 114
  - liboffload\_error\_codes.h, 121
- report\_get\_message\_str
  - liboffload\_error.c, 114
  - liboffload\_error\_codes.h, 121
- report\_get\_target\_stage\_str
  - liboffload\_error.c, 115
  - liboffload\_error\_codes.h, 121
- reserve
  - ORSL, 15
- reserve\_card
  - orl-lite.c, 198
- Reserved
  - ArrDesc, 19
- result
  - \_Offload\_status, 17
  - mic\_lib::offload\_status, 60
- routine\_id
  - PersistData, 76
- rsrv\_cnt
  - orl-lite.c, 199
- rsrv\_data
  - orl-lite.c, 199
- scatter\_copyin\_data
  - OffloadDescriptor, 65
- scatter\_copyout\_data
  - OffloadDescriptor, 65
- send\_data
  - Marshaller, 47
- send\_func\_ptr
  - Marshaller, 47
- send\_noncontiguous\_pointer\_data
  - OffloadDescriptor, 65
- send\_pointer\_data
  - OffloadDescriptor, 65
- server\_compute
  - coi\_server.cpp, 104
- server\_init
  - coi\_server.cpp, 104
- server\_var\_table\_copy
  - coi\_server.cpp, 104
- server\_var\_table\_size
  - coi\_server.cpp, 104
- set\_indexes
  - Engine, 29
- set\_offload\_number
  - OffloadDescriptor, 65
- set\_pipeline
  - Thread, 84
- set\_prefix
  - MicEnvVar, 51
- setup\_descriptors
  - OffloadDescriptor, 65
- setup\_misc\_data
  - OffloadDescriptor, 65
- shared\_table\_entries
  - offload\_myo\_host.cpp, 156
- SharedAlignedFree
  - MyoWrapper, 56
- SharedAlignedMalloc
  - MyoWrapper, 56
- SharedFree
  - MyoWrapper, 56
- SharedMalloc
  - MyoWrapper, 56
- SharedTableEntry
  - offload\_myo\_host.h, 157
  - offload\_myo\_target.h, 160
- SignalMap
  - Engine, 26
- sink\_addr



- VarDesc, 91
- size
  - CeanReadDim, 22
  - dim\_desc, 24
  - Image, 36
  - OffloadDescriptor::ReadArrElements, 80
  - TargetImage, 83
  - VarDesc, 91
- sname
  - VarDesc2, 92
- src
  - VarDesc, 91
- src\_data
  - OffloadDescriptor::VarExtra, 96
- stack\_alloc\_lock
  - offload\_host.cpp, 148
- stack\_cpu\_addr
  - PersistData, 76
- stack\_ptr\_data
  - PersistData, 77
- start
  - MemRange, 48
- state\_lock
  - orl-lite.c, 198
- state\_signal\_release
  - orl-lite.c, 198
- state\_unlock
  - orl-lite.c, 198
- state\_wait\_for\_release
  - orl-lite.c, 198
- stride
  - dim\_desc, 24
- TARGET\_HOST
  - offload.h, 130
- TARGET\_MIC
  - offload.h, 130
- TARGET\_NONE
  - offload.h, 130
- TARGET\_ATTRIBUTE
  - offload.h, 129
- TARGET\_TYPE
  - offload.h, 130
- Table
  - TableList, 82
- table
  - TableList::Node, 58
- table\_copy
  - VarList, 97
- table\_patch\_names
  - VarList, 97
- table\_size
  - VarList, 97
- TableList
  - add\_table, 82
  - m\_head, 82
  - m\_lock, 82
  - remove\_table, 82
  - Table, 82
  - TableList, 82
  - TableList, 82
  - TableList< T >, 81
  - TableList< T >::Node, 58
  - TableList::Node
    - next, 58
    - prev, 58
    - table, 58
  - target\_entry\_cmp
    - offload\_engine.cpp, 140
  - target\_mic
    - mic\_lib, 50
  - TargetImage, 82
    - data, 83
    - name, 83
    - offset, 83
    - origin, 83
    - size, 83
    - TargetImage, 83
    - TargetImage, 83
  - TargetImageList
    - offload\_engine.h, 141
  - test\_msg\_cat
    - liboffload\_error\_codes.h, 116
  - test\_msg\_cat1
    - liboffload\_error\_codes.h, 116
  - tfr\_size
    - Marshaller, 47
  - Thread, 84
    - ~Thread, 84
    - get\_auto\_vars, 84
    - get\_pipeline, 84
    - m\_addr\_coipipe\_counter, 84
    - m\_auto\_vars, 84
    - m\_pipelines, 84
    - set\_pipeline, 84
    - Thread, 84
  - thread\_getspecific
    - offload\_util.h, 189
  - thread\_key\_create
    - offload\_util.h, 189
  - thread\_key\_delete
    - offload\_util.h, 189
  - thread\_setspecific
    - offload\_util.h, 189
  - timer\_enabled
    - FunctionDescriptor, 35
    - offload\_timer.h, 184
    - offload\_timer\_host.cpp, 185
    - offload\_timer\_target.cpp, 185
  - timer\_envname
    - offload\_host.cpp, 148
  - translate\_coi\_error
    - OffloadDescriptor, 65
  - try\_reserve
    - ORSL, 15
  - type
    - ORSLBusySet, 76

- VarDesc, 91
- UnloadLibrary
  - MyoWrapper, 56
- unlock
  - mutex\_t, 53
- upper
  - dim\_desc, 24
- VAR\_TYPE\_IS\_PTR
  - offload\_common.h, 137
- va\_copy
  - liboffload\_error.c, 114
- val
  - OffloadDescriptor::ReadArrElements, 80
- var\_tab
  - MyoTable, 54
- var\_tab\_len
  - MyoTable, 54
- VarDesc, 85
  - align, 86
  - alloc, 86
  - alloc\_disp, 86
  - alloc\_if, 87
  - bits, 87
  - count, 87
  - direction, 87
  - disp, 88
  - dst, 88
  - flags, 88
  - free\_if, 88
  - has\_length, 88
  - in, 89
  - into, 89
  - is\_noncont\_dst, 89
  - is\_noncont\_src, 89
  - is\_stack\_buf, 89
  - is\_static, 89
  - is\_static\_dstn, 89
  - mic\_offset, 90
  - offset, 90
  - out, 90
  - ptr, 90
  - ptr\_arr\_offset, 90
  - sink\_addr, 91
  - size, 91
  - src, 91
  - type, 91
- VarDesc2, 92
  - dname, 92
  - sname, 92
- VarDesc3, 92
  - align\_array, 93
  - alloc\_elements, 93
  - alloc\_if\_array, 93
  - alloc\_start, 93
  - array\_fields, 93
  - extent\_elements, 94
  - extent\_start, 94
  - free\_if\_array, 94
  - into\_elements, 94
  - into\_start, 94
  - ptr\_array, 94
- VarList, 96
  - begin, 97
  - dump, 97
  - end, 97
  - table\_copy, 97
  - table\_patch\_names, 97
  - table\_size, 97
  - VarList, 97
  - VarList, 97
- VarList::BufEntry, 20
  - addr, 21
  - name, 21
- VarList::Iterator, 37
  - Iterator, 37
  - m\_entry, 38
  - m\_node, 38
  - new\_node, 38
  - operator\*, 38
  - operator++, 38
  - operator==, 38
- VarTable, 97
  - entries, 98
- VarTable::Entry, 31
  - addr, 32
  - name, 32
- VarValue
  - MicEnvVar::VarValue, 98
- vardesc\_direction\_as\_string
  - offload\_host.cpp, 149
  - offload\_target.cpp, 182
- vardesc\_type\_as\_string
  - offload\_host.cpp, 149
  - offload\_target.cpp, 182
- vars\_num
  - FunctionDescriptor, 36
- wait\_dependencies
  - OffloadDescriptor, 66
- write\_message
  - liboffload\_error\_codes.h, 121
  - liboffload\_msg.c, 121