

My Project

Generated by Doxygen 1.8.14

Contents

1	Namespace Index	1
1.1	Namespace List	1
2	Hierarchical Index	3
2.1	Class Hierarchy	3
3	Class Index	9
3.1	Class List	9
4	File Index	15
4.1	File List	15
5	Namespace Documentation	17
5.1	nobsrule Namespace Reference	17
5.1.1	Detailed Description	17
5.1.2	Function Documentation	17
5.1.2.1	requires_upcxx_backend()	17
5.2	std Namespace Reference	17
5.3	upcxx Namespace Reference	18
5.3.1	Detailed Description	23
5.3.2	Typedef Documentation	23
5.3.2.1	future	23
5.3.2.2	inrank_t	23
5.3.2.3	make_index_sequence	23
5.3.2.4	tuple_types_into_t	24

5.3.2.5	uinrank_t	24
5.3.3	Enumeration Type Documentation	24
5.3.3.1	progress_level	24
5.3.4	Function Documentation	24
5.3.4.1	allocate() [1/2]	24
5.3.4.2	allocate() [2/2]	25
5.3.4.3	allreduce()	25
5.3.4.4	apply_tupled()	25
5.3.4.5	apply_tupled_as_future()	25
5.3.4.6	assert_failed()	25
5.3.4.7	atomic_fetch_add()	26
5.3.4.8	atomic_get()	26
5.3.4.9	atomic_put()	26
5.3.4.10	barrier()	26
5.3.4.11	bind() [1/2]	26
5.3.4.12	bind() [2/2]	27
5.3.4.13	bind_last() [1/2]	27
5.3.4.14	bind_last() [2/2]	27
5.3.4.15	broadcast()	27
5.3.4.16	command_execute()	27
5.3.4.17	command_pack() [1/2]	28
5.3.4.18	command_pack() [2/2]	28
5.3.4.19	command_size_ubound() [1/2]	28
5.3.4.20	command_size_ubound() [2/2]	28
5.3.4.21	constant()	28
5.3.4.22	dbgbrk()	29
5.3.4.23	deallocate() [1/2]	29
5.3.4.24	deallocate() [2/2]	29
5.3.4.25	delete_()	29
5.3.4.26	delete_array()	29

5.3.4.27	fast_hash()	30
5.3.4.28	finalize()	30
5.3.4.29	get_or_void()	30
5.3.4.30	init()	30
5.3.4.31	is_memory_shared_with()	30
5.3.4.32	make_future()	31
5.3.4.33	mod2n_hash()	31
5.3.4.34	new_() [1/2]	31
5.3.4.35	new_() [2/2]	31
5.3.4.36	new_array() [1/2]	31
5.3.4.37	new_array() [2/2]	32
5.3.4.38	nop()	32
5.3.4.39	operator+()	32
5.3.4.40	operator<<() [1/3]	32
5.3.4.41	operator<<() [2/3]	32
5.3.4.42	operator<<() [3/3]	33
5.3.4.43	operator>>()	33
5.3.4.44	operator>>=()	33
5.3.4.45	operator" ()	33
5.3.4.46	operxn_cx_as_promise()	34
5.3.4.47	os_env() [1/2]	34
5.3.4.48	os_env() [2/2]	34
5.3.4.49	print()	34
5.3.4.50	progress()	34
5.3.4.51	pshm_local_addr2remote()	35
5.3.4.52	pshm_remote_addr2local()	35
5.3.4.53	rank_me()	35
5.3.4.54	rank_n()	35
5.3.4.55	reflect_upon()	35
5.3.4.56	reinterpret_pointer_cast()	36

5.3.4.57	<code>remote_cx_as_rpc()</code>	36
5.3.4.58	<code>rget()</code> [1/2]	36
5.3.4.59	<code>rget()</code> [2/2]	36
5.3.4.60	<code>rpc()</code> [1/2]	37
5.3.4.61	<code>rpc()</code> [2/2]	37
5.3.4.62	<code>rpc_ff()</code>	37
5.3.4.63	<code>rput()</code> [1/2]	37
5.3.4.64	<code>rput()</code> [2/2]	38
5.3.4.65	<code>source_cx_as_promise()</code>	38
5.3.4.66	<code>to_future()</code> [1/2]	38
5.3.4.67	<code>to_future()</code> [2/2]	38
5.3.4.68	<code>tuple_rvals()</code>	38
5.3.4.69	<code>wait()</code>	39
5.3.4.70	<code>when_all()</code>	39
5.3.5	Variable Documentation	39
5.3.5.1	<code>dbgbrk_spin</code>	39
5.3.5.2	<code>dbgbrk_spin_init</code>	39
5.3.5.3	<code>operxn_cx_as_future</code>	39
5.3.5.4	<code>source_cx_as_future</code>	40
5.4	<code>upcxx::backend</code> Namespace Reference	40
5.4.1	Function Documentation	40
5.4.1.1	<code>during_level()</code> [1/2]	41
5.4.1.2	<code>during_level()</code> [2/2]	41
5.4.1.3	<code>during_user()</code> [1/4]	41
5.4.1.4	<code>during_user()</code> [2/4]	41
5.4.1.5	<code>during_user()</code> [3/4]	41
5.4.1.6	<code>during_user()</code> [4/4]	42
5.4.1.7	<code>make_rma_get_cb()</code>	42
5.4.1.8	<code>make_rma_put_cb()</code>	42
5.4.1.9	<code>rma_get()</code>	42

5.4.1.10	rma_put()	42
5.4.1.11	send_am()	43
5.4.2	Variable Documentation	43
5.4.2.1	rank_me	43
5.4.2.2	rank_n	43
5.5	upcxx::backend::gasnet1_seq Namespace Reference	43
5.5.1	Function Documentation	44
5.5.1.1	send_am_eager_queued()	44
5.5.1.2	send_am_eager_restricted()	44
5.5.1.3	send_am_rdzv()	44
5.5.1.4	send_am_restricted()	45
5.5.2	Variable Documentation	45
5.5.2.1	am_size_rdzv_cutover	45
5.5.2.2	in_user_progress_	45
5.5.2.3	user_actions_head_	45
5.5.2.4	user_actions_tailp_	45
5.6	upcxx::detail Namespace Reference	46
5.6.1	Typedef Documentation	49
5.6.1.1	apply_futured_as_future_return_t	49
5.6.1.2	future_from_tuple_t	49
5.6.1.3	promise_like_t	49
5.6.1.4	when_all_return_t	49
5.6.2	Function Documentation	49
5.6.2.1	apply_tupled()	50
5.6.2.2	bind_last()	50
5.6.2.3	broadcast_receive()	50
5.6.2.4	command_executor()	50
5.6.2.5	dist_promise()	51
5.6.2.6	new_()	51
5.6.2.7	new_array()	51
5.6.2.8	packing_funptr_basis()	51
5.6.2.9	tuple_rvals()	51
5.6.3	Variable Documentation	52
5.6.3.1	dist_master_id_bump	52
5.6.3.2	dist_master_promises	52

6 Class Documentation	53
6.1 <code>upcxx::backend::gasnet1_seq::action</code> Struct Reference	53
6.1.1 Detailed Description	53
6.1.2 Member Function Documentation	53
6.1.2.1 <code>fire_and_delete()</code>	53
6.1.3 Member Data Documentation	54
6.1.3.1 <code>next_</code>	54
6.2 <code>upcxx::backend::gasnet1_seq::action_impl< Fn ></code> Struct Template Reference	54
6.2.1 Detailed Description	54
6.2.2 Constructor & Destructor Documentation	54
6.2.2.1 <code>action_impl()</code>	55
6.2.3 Member Function Documentation	55
6.2.3.1 <code>fire_and_delete()</code>	55
6.2.4 Member Data Documentation	55
6.2.4.1 <code>fn</code>	55
6.3 <code>upcxx::add_lref_if_nonref< T ></code> Struct Template Reference	55
6.3.1 Detailed Description	56
6.3.2 Member Typedef Documentation	56
6.3.2.1 <code>type</code>	56
6.4 <code>upcxx::add_lref_if_nonref< T & ></code> Struct Template Reference	56
6.4.1 Detailed Description	56
6.4.2 Member Typedef Documentation	56
6.4.2.1 <code>type</code>	57
6.5 <code>upcxx::add_lref_if_nonref< T && ></code> Struct Template Reference	57
6.5.1 Detailed Description	57
6.5.2 Member Typedef Documentation	57
6.5.2.1 <code>type</code>	57
6.6 <code>upcxx::detail::allreduce_state< T, Op ></code> Struct Template Reference	57
6.6.1 Detailed Description	58
6.6.2 Member Function Documentation	58

6.6.2.1	broadcast()	58
6.6.2.2	contributed()	58
6.6.3	Member Data Documentation	58
6.6.3.1	accum	59
6.6.3.2	answer	59
6.6.3.3	incoming	59
6.6.3.4	op	59
6.7	upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator< Arg > Struct Template Reference	59
6.7.1	Detailed Description	60
6.7.2	Member Function Documentation	60
6.7.2.1	apply_()	60
6.7.2.2	operator()	60
6.7.3	Member Data Documentation	60
6.7.3.1	a	61
6.7.3.2	b	61
6.7.3.3	fn	61
6.8	upcxx::detail::apply_futured_as_future< Fn(future1< Kind, T... >)> Struct Template Reference	61
6.8.1	Detailed Description	62
6.8.2	Member Typedef Documentation	62
6.8.2.1	return_type	62
6.8.2.2	tupled_impl	62
6.8.3	Member Function Documentation	62
6.8.3.1	operator() [1/2]	62
6.8.3.2	operator() [2/2]	63
6.9	upcxx::detail::apply_tupled_as_future< Fn, ArgTup > Struct Template Reference	63
6.9.1	Detailed Description	63
6.10	upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, ArgTupDecayed > Struct Template Reference	63
6.10.1	Detailed Description	64
6.11	upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, std::tuple< T... > > Struct Template Reference	64

6.11.1 Detailed Description	64
6.12 upcxx::detail::apply_tupled_as_future_impl< Return > Struct Template Reference	64
6.12.1 Detailed Description	65
6.12.2 Member Function Documentation	65
6.12.2.1 <code>operator()</code>	65
6.12.3 Member Data Documentation	65
6.12.3.1 <code>return_type</code>	65
6.13 upcxx::detail::apply_tupled_as_future_impl< future1< Kind, T... > > Struct Template Reference	65
6.13.1 Detailed Description	66
6.13.2 Member Typedef Documentation	66
6.13.2.1 <code>return_type</code>	66
6.13.3 Member Function Documentation	66
6.13.3.1 <code>operator()</code>	66
6.14 upcxx::detail::apply_tupled_as_future_impl< void > Struct Template Reference	67
6.14.1 Detailed Description	67
6.14.2 Member Function Documentation	67
6.14.2.1 <code>operator()</code>	67
6.14.3 Member Data Documentation	67
6.14.3.1 <code>return_type</code>	67
6.15 upcxx::detail::bind< FnDecayed > Struct Template Reference	68
6.15.1 Detailed Description	68
6.15.2 Member Function Documentation	68
6.15.2.1 <code>operator()</code>	68
6.16 upcxx::detail::bind< bound_function< Fn0, B0... > > Struct Template Reference	68
6.16.1 Detailed Description	69
6.16.2 Member Function Documentation	69
6.16.2.1 <code>operator()</code>	69
6.17 upcxx::binding< T > Struct Template Reference	69
6.17.1 Detailed Description	70
6.17.2 Member Typedef Documentation	70

6.17.2.1	off_wire_type	70
6.17.2.2	on_wire_type	70
6.17.3	Member Function Documentation	70
6.17.3.1	off_wire()	70
6.17.3.2	on_wire()	71
6.17.4	Member Data Documentation	71
6.17.4.1	is_trivial	71
6.18	upcxx::binding< dist_object< T > & > Struct Template Reference	71
6.18.1	Detailed Description	71
6.18.2	Member Typedef Documentation	72
6.18.2.1	off_wire_type	72
6.18.2.2	on_wire_type	72
6.18.3	Member Function Documentation	72
6.18.3.1	off_wire()	72
6.18.3.2	on_wire()	72
6.19	upcxx::binding< dist_object< T > && > Struct Template Reference	73
6.19.1	Detailed Description	73
6.20	upcxx::binding< T & > Struct Template Reference	73
6.20.1	Detailed Description	73
6.21	upcxx::binding< T && > Struct Template Reference	74
6.21.1	Detailed Description	74
6.22	upcxx::binding< T const > Struct Template Reference	74
6.22.1	Detailed Description	74
6.23	upcxx::binding< T volatile > Struct Template Reference	75
6.23.1	Detailed Description	75
6.24	upcxx::binding_is_trivial< T, trivial > Struct Template Reference	75
6.24.1	Detailed Description	75
6.24.2	Member Data Documentation	75
6.24.2.1	value	76
6.25	upcxx::binding_is_trivial< T, std::integral_constant< bool, binding< T >::is_trivial > > Struct Template Reference	76

6.25.1	Detailed Description	76
6.25.2	Member Data Documentation	76
6.25.2.1	value	76
6.26	upcxx::bound_function< Fn, B > Struct Template Reference	77
6.26.1	Detailed Description	77
6.26.2	Member Typedef Documentation	77
6.26.2.1	base_type	77
6.26.3	Constructor & Destructor Documentation	78
6.26.3.1	bound_function()	78
6.26.4	Member Function Documentation	78
6.26.4.1	operator() [1/2]	78
6.26.4.2	operator() [2/2]	78
6.27	upcxx::detail::bound_function_base< Fn, BndTup, all_trivial > Struct Template Reference	78
6.27.1	Detailed Description	79
6.28	upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false > Struct Template Reference	79
6.28.1	Detailed Description	79
6.28.2	Member Function Documentation	79
6.28.2.1	apply_()	80
6.28.3	Member Data Documentation	80
6.28.3.1	b_	80
6.28.3.2	fn_	80
6.29	upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true > Struct Template Reference	80
6.29.1	Detailed Description	81
6.29.2	Member Function Documentation	81
6.29.2.1	apply_()	81
6.29.3	Member Data Documentation	81
6.29.3.1	b_	81
6.29.3.2	fn_	82
6.30	upcxx::commanding< Fn > Struct Template Reference	82
6.30.1	Detailed Description	82

6.30.2	Member Function Documentation	82
6.30.2.1	execute()	82
6.30.2.2	pack()	83
6.30.2.3	size_ubound()	83
6.31	upcxx::completions< SourceCx, RemoteCx, OpnxCx > Struct Template Reference	83
6.31.1	Detailed Description	83
6.31.2	Member Data Documentation	84
6.31.2.1	future_n	84
6.31.2.2	operxn	84
6.31.2.3	remote	84
6.31.2.4	source	84
6.32	upcxx::constant_function< T > Struct Template Reference	85
6.32.1	Detailed Description	85
6.32.2	Constructor & Destructor Documentation	85
6.32.2.1	constant_function()	85
6.32.3	Member Function Documentation	85
6.32.3.1	operator>()	85
6.32.4	Member Data Documentation	86
6.32.4.1	value_	86
6.33	upcxx::decay_tupled< Tup > Struct Template Reference	86
6.33.1	Detailed Description	86
6.34	upcxx::decay_tupled< std::tuple< T... > > Struct Template Reference	86
6.34.1	Detailed Description	86
6.34.2	Member Typedef Documentation	87
6.34.2.1	type	87
6.35	upcxx::detail::future_header::dependency_link Struct Reference	87
6.35.1	Detailed Description	87
6.35.2	Member Function Documentation	87
6.35.2.1	unlink()	87
6.35.3	Member Data Documentation	88

6.35.3.1	dep	88
6.35.3.2	suc	88
6.35.3.3	sucs_next	88
6.36	upcxx::digest Struct Reference	88
6.36.1	Detailed Description	89
6.36.2	Member Function Documentation	89
6.36.2.1	eat() [1/2]	89
6.36.2.2	eat() [2/2]	89
6.36.2.3	zero()	89
6.36.3	Friends And Related Function Documentation	90
6.36.3.1	operator"!="	90
6.36.3.2	operator<	90
6.36.3.3	operator<=	90
6.36.3.4	operator==	90
6.36.3.5	operator>	90
6.36.3.6	operator>=	91
6.36.4	Member Data Documentation	91
6.36.4.1	w0	91
6.36.4.2	w1	91
6.37	upcxx::detail::disjoin_cx< A, B, B_ord_bump > Struct Template Reference	91
6.37.1	Detailed Description	91
6.38	upcxx::detail::disjoin_cx< A, nil_cx, B_ord_bump > Struct Template Reference	92
6.38.1	Detailed Description	92
6.38.2	Member Typedef Documentation	92
6.38.2.1	type	92
6.38.3	Member Function Documentation	92
6.38.3.1	operator>()	92
6.39	upcxx::detail::disjoin_cx< nil_cx, B, B_ord_bump > Struct Template Reference	93
6.39.1	Detailed Description	93
6.39.2	Member Typedef Documentation	93

6.39.2.1	type	93
6.39.3	Member Function Documentation	93
6.39.3.1	operator()	93
6.40	upcxx::detail::disjoin_cx< nil_cx, future_cx< B_ord_old >, B_ord_bump > Struct Template Reference	94
6.40.1	Detailed Description	94
6.40.2	Member Typedef Documentation	94
6.40.2.1	type	94
6.40.3	Member Function Documentation	94
6.40.3.1	operator()	94
6.41	upcxx::detail::disjoin_cx< nil_cx, nil_cx, B_ord0 > Struct Template Reference	95
6.41.1	Detailed Description	95
6.41.2	Member Typedef Documentation	95
6.41.2.1	type	95
6.41.3	Member Function Documentation	95
6.41.3.1	operator()	95
6.42	upcxx::dist_id< T > Struct Template Reference	96
6.42.1	Detailed Description	96
6.42.2	Member Function Documentation	96
6.42.2.1	here()	96
6.42.2.2	when_here()	96
6.42.3	Member Data Documentation	96
6.42.3.1	dig_	97
6.43	upcxx::dist_object< T > Class Template Reference	97
6.43.1	Detailed Description	97
6.43.2	Constructor & Destructor Documentation	97
6.43.2.1	dist_object() [1/3]	97
6.43.2.2	dist_object() [2/3]	98
6.43.2.3	dist_object() [3/3]	98
6.43.2.4	~dist_object()	98
6.43.3	Member Function Documentation	98

6.43.3.1	fetch()	98
6.43.3.2	id()	98
6.43.3.3	operator*()	99
6.43.3.4	operator->()	99
6.44	upcxx::fast_hasher Struct Reference	99
6.44.1	Detailed Description	99
6.44.2	Constructor & Destructor Documentation	99
6.44.2.1	fast_hasher()	100
6.44.3	Member Function Documentation	100
6.44.3.1	operator()()	100
6.44.3.2	result()	100
6.44.4	Member Data Documentation	100
6.44.4.1	s	101
6.45	upcxx::fast_hashing< T > Struct Template Reference	101
6.45.1	Detailed Description	101
6.45.2	Member Function Documentation	101
6.45.2.1	operator()()	101
6.46	upcxx::function_ref< Sig > Class Template Reference	101
6.46.1	Detailed Description	102
6.47	upcxx::function_ref< Ret(Arg...) > Class Template Reference	102
6.47.1	Detailed Description	102
6.47.2	Constructor & Destructor Documentation	102
6.47.2.1	function_ref() [1/4]	102
6.47.2.2	function_ref() [2/4]	103
6.47.2.3	function_ref() [3/4]	103
6.47.2.4	function_ref() [4/4]	103
6.47.3	Member Function Documentation	103
6.47.3.1	operator()()	103
6.47.3.2	operator=() [1/2]	103
6.47.3.3	operator=() [2/2]	104

6.48	upcxx::future1< Kind, T > Struct Template Reference	104
6.48.1	Detailed Description	105
6.48.2	Member Typedef Documentation	105
6.48.2.1	impl_type	105
6.48.2.2	kind_type	105
6.48.2.3	results_type	105
6.48.3	Constructor & Destructor Documentation	105
6.48.3.1	future1() [1/7]	106
6.48.3.2	~future1()	106
6.48.3.3	future1() [2/7]	106
6.48.3.4	future1() [3/7]	106
6.48.3.5	future1() [4/7]	106
6.48.3.6	future1() [5/7]	106
6.48.3.7	future1() [6/7]	107
6.48.3.8	future1() [7/7]	107
6.48.4	Member Function Documentation	107
6.48.4.1	operator=() [1/4]	107
6.48.4.2	operator=() [2/4]	107
6.48.4.3	operator=() [3/4]	107
6.48.4.4	operator=() [4/4]	108
6.48.4.5	ready()	108
6.48.4.6	result()	108
6.48.4.7	result_moved()	108
6.48.4.8	results()	108
6.48.4.9	results_moved()	109
6.48.4.10	then()	109
6.48.4.11	then_pure()	109
6.48.5	Member Data Documentation	109
6.48.5.1	impl_	109
6.49	upcxx::detail::future_body Struct Reference	110

6.49.1	Detailed Description	110
6.49.2	Constructor & Destructor Documentation	110
6.49.2.1	future_body()	110
6.49.3	Member Function Documentation	110
6.49.3.1	destruct_early()	110
6.49.3.2	leave_active()	111
6.49.4	Member Data Documentation	111
6.49.4.1	storage_	111
6.50	upcxx::detail::future_body_proxy< T > Struct Template Reference	111
6.50.1	Detailed Description	112
6.50.2	Constructor & Destructor Documentation	112
6.50.2.1	future_body_proxy()	112
6.50.3	Member Function Documentation	112
6.50.3.1	destruct_early()	112
6.51	upcxx::detail::future_body_proxy_ Struct Reference	112
6.51.1	Detailed Description	113
6.51.2	Constructor & Destructor Documentation	113
6.51.2.1	future_body_proxy_()	113
6.51.3	Member Function Documentation	113
6.51.3.1	leave_active()	113
6.51.4	Member Data Documentation	113
6.51.4.1	link_	114
6.52	upcxx::detail::future_body_pure< FuArg > Struct Template Reference	114
6.52.1	Detailed Description	114
6.53	upcxx::detail::future_body_pure< future1< Kind, T... > > Struct Template Reference	114
6.53.1	Detailed Description	115
6.53.2	Constructor & Destructor Documentation	115
6.53.2.1	future_body_pure()	115
6.53.3	Member Function Documentation	115
6.53.3.1	destruct_early()	115

6.53.3.2	<code>leave_active()</code>	115
6.53.4	Member Data Documentation	116
6.53.4.1	<code>dep_</code>	116
6.54	<code>upcxx::detail::future_body_then< FuArg, Fn ></code> Struct Template Reference	116
6.54.1	Detailed Description	116
6.54.2	Constructor & Destructor Documentation	117
6.54.2.1	<code>future_body_then()</code>	117
6.54.3	Member Function Documentation	117
6.54.3.1	<code>leave_active()</code>	117
6.54.4	Member Data Documentation	117
6.54.4.1	<code>dep_</code>	117
6.54.4.2	<code>fn_</code>	118
6.55	<code>upcxx::detail::future_body_then_base</code> Struct Reference	118
6.55.1	Detailed Description	118
6.55.2	Constructor & Destructor Documentation	118
6.55.2.1	<code>future_body_then_base()</code>	118
6.55.3	Member Function Documentation	119
6.55.3.1	<code>leave_active_into_proxy()</code>	119
6.56	<code>upcxx::detail::future_body_then_pure< FuArg, Fn ></code> Struct Template Reference	119
6.56.1	Detailed Description	120
6.56.2	Constructor & Destructor Documentation	120
6.56.2.1	<code>future_body_then_pure()</code>	120
6.56.3	Member Function Documentation	120
6.56.3.1	<code>destruct_early()</code>	120
6.56.3.2	<code>leave_active()</code>	120
6.56.4	Member Data Documentation	121
6.56.4.1	<code>dep_</code>	121
6.56.4.2	<code>fn_</code>	121
6.57	<code>upcxx::future_cx< ordinal ></code> Struct Template Reference	121
6.57.1	Detailed Description	121

6.58	upcxx::detail::future_dependency< FuArg > Struct Template Reference	121
6.58.1	Detailed Description	122
6.59	upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... >> Struct Template Reference	122
6.59.1	Detailed Description	122
6.59.2	Member Typedef Documentation	122
6.59.2.1	result_lrefs_function	123
6.59.3	Constructor & Destructor Documentation	123
6.59.3.1	future_dependency()	123
6.59.4	Member Function Documentation	123
6.59.4.1	cleanup_early()	123
6.59.4.2	cleanup_ready()	123
6.59.4.3	cleanup_ready_get_header()	124
6.59.4.4	result_lrefs_getter()	124
6.59.5	Member Data Documentation	124
6.59.5.1	dep_	124
6.59.5.2	fn_	124
6.60	upcxx::detail::future_dependency< future1< future_kind_result >> Struct Template Reference	124
6.60.1	Detailed Description	125
6.60.2	Constructor & Destructor Documentation	125
6.60.2.1	future_dependency()	125
6.60.3	Member Function Documentation	125
6.60.3.1	cleanup_early()	125
6.60.3.2	cleanup_ready()	125
6.60.3.3	cleanup_ready_get_header()	126
6.60.3.4	result_lrefs_getter()	126
6.61	upcxx::detail::future_dependency< future1< future_kind_result, T... >> Struct Template Reference	126
6.61.1	Detailed Description	126
6.61.2	Constructor & Destructor Documentation	127
6.61.2.1	future_dependency()	127
6.61.3	Member Function Documentation	127

6.61.3.1	cleanup_early()	127
6.61.3.2	cleanup_ready()	127
6.61.3.3	cleanup_ready_get_header()	127
6.61.3.4	result_lrefs_getter()	128
6.61.4	Member Data Documentation	128
6.61.4.1	results_	128
6.62	upcxx::detail::future_dependency< future1< future_kind_shref< HeaderOps >, T... > > Struct Template Reference	128
6.62.1	Detailed Description	128
6.62.2	Constructor & Destructor Documentation	129
6.62.2.1	future_dependency()	129
6.63	upcxx::detail::future_dependency< future1< future_kind_when_all< Arg... >, T... > > Struct Template Reference	129
6.63.1	Detailed Description	129
6.63.2	Constructor & Destructor Documentation	129
6.63.2.1	future_dependency()	130
6.64	upcxx::detail::future_dependency_shref_base< is_trivially_ready_result > Struct Template Reference	130
6.64.1	Detailed Description	130
6.65	upcxx::detail::future_dependency_shref_base< false > Struct Template Reference	130
6.65.1	Detailed Description	131
6.65.2	Constructor & Destructor Documentation	131
6.65.2.1	future_dependency_shref_base() [1/3]	131
6.65.2.2	future_dependency_shref_base() [2/3]	131
6.65.2.3	future_dependency_shref_base() [3/3]	131
6.65.3	Member Function Documentation	131
6.65.3.1	header_()	131
6.65.3.2	unlink_()	132
6.65.4	Member Data Documentation	132
6.65.4.1	link_	132
6.66	upcxx::detail::future_dependency_shref_base< true > Struct Template Reference	132
6.66.1	Detailed Description	132

6.66.2	Constructor & Destructor Documentation	132
6.66.2.1	future_dependency_shref_base()	133
6.66.3	Member Function Documentation	133
6.66.3.1	header_()	133
6.66.3.2	unlink_()	133
6.66.4	Member Data Documentation	133
6.66.4.1	hdr_	133
6.67	upcxx::detail::future_dependency_when_all_arg< i, Arg > Struct Template Reference	134
6.67.1	Detailed Description	134
6.67.2	Constructor & Destructor Documentation	134
6.67.2.1	future_dependency_when_all_arg()	134
6.67.3	Member Data Documentation	134
6.67.3.1	dep_	135
6.68	upcxx::detail::future_dependency_when_all_base< AllArg, lxFSeq > Struct Template Reference	135
6.68.1	Detailed Description	135
6.69	upcxx::detail::future_dependency_when_all_base< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... > > Struct Template Reference	135
6.69.1	Detailed Description	136
6.69.2	Member Typedef Documentation	136
6.69.2.1	this_t	136
6.69.3	Constructor & Destructor Documentation	136
6.69.3.1	future_dependency_when_all_base()	136
6.69.4	Member Function Documentation	136
6.69.4.1	result_lrefs_getter()	137
6.70	upcxx::detail::future_from_tuple< Kind, Tup > Struct Template Reference	137
6.70.1	Detailed Description	137
6.71	upcxx::detail::future_from_tuple< Kind, std::tuple< T... > > Struct Template Reference	137
6.71.1	Detailed Description	137
6.71.2	Member Typedef Documentation	138
6.71.2.1	type	138
6.72	upcxx::detail::future_header Struct Reference	138

6.72.1	Detailed Description	139
6.72.2	Member Function Documentation	139
6.72.2.1	drop_for_proxied()	139
6.72.2.2	drop_for_result()	139
6.72.2.3	enter_ready()	140
6.72.2.4	refs_add()	140
6.72.2.5	refs_drop()	140
6.72.3	Member Data Documentation	140
6.72.3.1	"@2	140
6.72.3.2	body_	140
6.72.3.3	ref_n_	140
6.72.3.4	result_	141
6.72.3.5	status_	141
6.72.3.6	status_active	141
6.72.3.7	status_proxying	141
6.72.3.8	status_proxying_active	141
6.72.3.9	status_ready	141
6.72.3.10	sucs_head_	142
6.72.3.11	the_nil	142
6.73	upcxx::detail::future_header_dependent Struct Reference	142
6.73.1	Detailed Description	143
6.73.2	Constructor & Destructor Documentation	143
6.73.2.1	future_header_dependent()	143
6.73.3	Member Function Documentation	143
6.73.3.1	enter_proxying()	143
6.73.3.2	entered_active()	143
6.73.3.3	refs_add()	144
6.73.3.4	refs_drop()	144
6.73.4	Member Data Documentation	144
6.73.4.1	active_next_	144

6.74	upcxx::detail::future_header_ops_general Struct Reference	144
6.74.1	Detailed Description	145
6.74.2	Member Function Documentation	145
6.74.2.1	decref_header()	145
6.74.2.2	delete_header()	145
6.74.3	Member Data Documentation	145
6.74.3.1	is_trivially_ready_result	146
6.75	upcxx::detail::future_header_ops_result Struct Reference	146
6.75.1	Detailed Description	146
6.75.2	Member Function Documentation	146
6.75.2.1	decref_header()	147
6.75.2.2	delete_header()	147
6.75.3	Member Data Documentation	147
6.75.3.1	is_trivially_ready_result	147
6.76	upcxx::detail::future_header_ops_result_ready Struct Reference	147
6.76.1	Detailed Description	148
6.76.2	Member Function Documentation	148
6.76.2.1	decref_header()	148
6.76.2.2	delete_header()	148
6.76.3	Member Data Documentation	148
6.76.3.1	is_trivially_ready_result	149
6.77	upcxx::detail::future_header_result< T > Struct Template Reference	149
6.77.1	Detailed Description	150
6.77.2	Constructor & Destructor Documentation	150
6.77.2.1	future_header_result() [1/2]	150
6.77.2.2	future_header_result() [2/2]	150
6.77.3	Member Function Documentation	150
6.77.3.1	construct_results() [1/2]	150
6.77.3.2	construct_results() [2/2]	151
6.77.3.3	delete_me()	151

6.77.3.4	<code>delete_me_ready()</code>	151
6.77.3.5	<code>readify()</code>	151
6.77.3.6	<code>results_of()</code>	151
6.77.4	Member Data Documentation	151
6.77.4.1	<code>"@4</code>	152
6.77.4.2	<code>results_</code>	152
6.77.4.3	<code>status_results_no</code>	152
6.77.4.4	<code>status_results_yes</code>	152
6.78	<code>upcxx::detail::future_header_result<></code> Struct Template Reference	152
6.78.1	Detailed Description	153
6.78.2	Member Enumeration Documentation	153
6.78.2.1	<code>anonymous_enum</code>	153
6.78.3	Constructor & Destructor Documentation	154
6.78.3.1	<code>future_header_result()</code> [1/2]	154
6.78.3.2	<code>future_header_result()</code> [2/2]	154
6.78.4	Member Function Documentation	154
6.78.4.1	<code>construct_results()</code> [1/2]	154
6.78.4.2	<code>construct_results()</code> [2/2]	154
6.78.4.3	<code>delete_me()</code>	154
6.78.4.4	<code>delete_me_ready()</code>	155
6.78.4.5	<code>readify()</code>	155
6.78.4.6	<code>results_of()</code>	155
6.78.5	Member Data Documentation	155
6.78.5.1	<code>the_always</code>	155
6.79	<code>upcxx::detail::future_impl_mapped< FuArg, Fn, T ></code> Struct Template Reference	155
6.79.1	Detailed Description	156
6.79.2	Member Typedef Documentation	156
6.79.2.1	<code>header_ops</code>	156
6.79.3	Constructor & Destructor Documentation	156
6.79.3.1	<code>future_impl_mapped()</code>	156

6.79.4	Member Function Documentation	157
6.79.4.1	ready()	157
6.79.4.2	result_lrefs_getter()	157
6.79.4.3	result_rvals()	157
6.79.4.4	steal_header()	157
6.79.5	Member Data Documentation	157
6.79.5.1	arg_	158
6.79.5.2	fn_	158
6.80	upcxx::detail::future_impl_result< T > Struct Template Reference	158
6.80.1	Detailed Description	158
6.80.2	Member Typedef Documentation	159
6.80.2.1	header_ops	159
6.80.3	Constructor & Destructor Documentation	159
6.80.3.1	future_impl_result()	159
6.80.4	Member Function Documentation	159
6.80.4.1	ready()	159
6.80.4.2	result_lrefs_getter()	159
6.80.4.3	result_rvals()	160
6.80.4.4	steal_header()	160
6.80.5	Member Data Documentation	160
6.80.5.1	results_	160
6.81	upcxx::detail::future_impl_result<> Struct Template Reference	160
6.81.1	Detailed Description	161
6.81.2	Member Typedef Documentation	161
6.81.2.1	header_ops	161
6.81.3	Member Function Documentation	161
6.81.3.1	ready()	161
6.81.3.2	result_lrefs_getter()	161
6.81.3.3	result_rvals()	161
6.81.3.4	steal_header()	162

6.82	upcxx::detail::future_impl_shref< HeaderOps, T > Struct Template Reference	162
6.82.1	Detailed Description	163
6.82.2	Member Typedef Documentation	163
6.82.2.1	header_ops	163
6.82.3	Constructor & Destructor Documentation	163
6.82.3.1	future_impl_shref() [1/7]	163
6.82.3.2	future_impl_shref() [2/7]	163
6.82.3.3	future_impl_shref() [3/7]	164
6.82.3.4	future_impl_shref() [4/7]	164
6.82.3.5	future_impl_shref() [5/7]	164
6.82.3.6	future_impl_shref() [6/7]	164
6.82.3.7	future_impl_shref() [7/7]	164
6.82.3.8	~future_impl_shref()	165
6.82.4	Member Function Documentation	165
6.82.4.1	operator=() [1/3]	165
6.82.4.2	operator=() [2/3]	165
6.82.4.3	operator=() [3/3]	165
6.82.4.4	ready()	165
6.82.4.5	result_lrefs_getter()	166
6.82.4.6	result_rvals()	166
6.82.4.7	steal_header()	166
6.82.5	Member Data Documentation	166
6.82.5.1	hdr_	166
6.83	upcxx::detail::future_impl_when_all< ArgTuple, T > Struct Template Reference	166
6.83.1	Detailed Description	167
6.84	upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... > Struct Template Reference	167
6.84.1	Detailed Description	167
6.84.2	Member Typedef Documentation	167
6.84.2.1	header_ops	168
6.84.3	Constructor & Destructor Documentation	168

6.84.3.1	future_impl_when_all()	168
6.84.4	Member Function Documentation	168
6.84.4.1	ready()	168
6.84.4.2	result_lrefs_getter()	168
6.84.4.3	result_rvals()	169
6.84.4.4	steal_header()	169
6.84.5	Member Data Documentation	169
6.84.5.1	args_	169
6.85	upcxx::future_is_trivially_ready< FutureOrKind > Struct Template Reference	169
6.85.1	Detailed Description	169
6.85.2	Member Data Documentation	170
6.85.2.1	value	170
6.86	upcxx::future_is_trivially_ready< future1< detail::future_kind_mapped< Arg, Fn >, T... >> Struct Template Reference	170
6.86.1	Detailed Description	170
6.86.2	Member Data Documentation	170
6.86.2.1	value	170
6.87	upcxx::future_is_trivially_ready< future1< detail::future_kind_result, T... >> Struct Template Reference	171
6.87.1	Detailed Description	171
6.87.2	Member Data Documentation	171
6.87.2.1	value	171
6.88	upcxx::future_is_trivially_ready< future1< detail::future_kind_shref< HeaderOps >, T... >> Struct Template Reference	171
6.88.1	Detailed Description	172
6.88.2	Member Data Documentation	172
6.88.2.1	value	172
6.89	upcxx::future_is_trivially_ready< future1< detail::future_kind_when_all< Arg... >, T... >> Struct Template Reference	172
6.89.1	Detailed Description	172
6.89.2	Member Data Documentation	172
6.89.2.1	value	173

6.90	upcxx::detail::future_kind_mapped< FuArg, Fn > Struct Template Reference	173
6.90.1	Detailed Description	173
6.90.2	Member Typedef Documentation	173
6.90.2.1	with_types	173
6.91	upcxx::detail::future_kind_result Struct Reference	174
6.91.1	Detailed Description	174
6.91.2	Member Typedef Documentation	174
6.91.2.1	with_types	174
6.92	upcxx::detail::future_kind_shref< HeaderOps > Struct Template Reference	174
6.92.1	Detailed Description	174
6.92.2	Member Typedef Documentation	175
6.92.2.1	with_types	175
6.93	upcxx::detail::future_kind_when_all< FuArg > Struct Template Reference	175
6.93.1	Detailed Description	175
6.93.2	Member Typedef Documentation	175
6.93.2.1	with_types	175
6.94	upcxx::detail::future_then< Arg, Fn, FnRet, arg_trivial > Struct Template Reference	176
6.94.1	Detailed Description	176
6.95	upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, false > Struct Template Reference	176
6.95.1	Detailed Description	176
6.95.2	Member Function Documentation	176
6.95.2.1	operator()	177
6.96	upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, true > Struct Template Reference	177
6.96.1	Detailed Description	177
6.96.2	Member Function Documentation	177
6.96.2.1	operator()	177
6.97	upcxx::detail::future_then_pure< Arg, Fn, FnRet, arg_trivial, fnret_trivial > Struct Template Reference	178
6.97.1	Detailed Description	178
6.98	upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, false > Struct Template Reference	178

6.98.1 Detailed Description	178
6.98.2 Member Function Documentation	178
6.98.2.1 operator()	179
6.99 upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, true > Struct Template Reference	179
6.99.1 Detailed Description	179
6.99.2 Member Function Documentation	179
6.99.2.1 operator()	179
6.100 upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, true, fnret_trivial > Struct Template Reference	180
6.100.1 Detailed Description	180
6.100.2 Member Function Documentation	180
6.100.2.1 operator()	180
6.101 upcxx::global_ptr< T > Class Template Reference	180
6.101.1 Detailed Description	181
6.101.2 Member Typedef Documentation	181
6.101.2.1 element_type	182
6.101.3 Constructor & Destructor Documentation	182
6.101.3.1 global_ptr() [1/3]	182
6.101.3.2 global_ptr() [2/3]	182
6.101.3.3 global_ptr() [3/3]	182
6.101.4 Member Function Documentation	182
6.101.4.1 is_local()	183
6.101.4.2 is_null()	183
6.101.4.3 local()	183
6.101.4.4 operator"!=()	183
6.101.4.5 operator+()	183
6.101.4.6 operator++() [1/2]	184
6.101.4.7 operator++() [2/2]	184
6.101.4.8 operator-() [1/2]	184
6.101.4.9 operator-() [2/2]	184

6.101.4.10	<code>operator--()</code> [1/2]	184
6.101.4.11	<code>operator--()</code> [2/2]	185
6.101.4.12	<code>operator<()</code>	185
6.101.4.13	<code>operator<=()</code>	185
6.101.4.14	<code>operator==()</code>	185
6.101.4.15	<code>operator>()</code>	185
6.101.4.16	<code>operator>=()</code>	186
6.101.4.17	<code>where()</code>	186
6.101.5	Friends And Related Function Documentation	186
6.101.5.1	<code>operator<<</code>	186
6.101.5.2	<code>reinterpret_pointer_cast</code>	186
6.101.5.3	<code>std::greater< global_ptr< T >></code>	186
6.101.5.4	<code>std::greater_equal< global_ptr< T >></code>	187
6.101.5.5	<code>std::hash< global_ptr< T >></code>	187
6.101.5.6	<code>std::less< global_ptr< T >></code>	187
6.101.5.7	<code>std::less_equal< global_ptr< T >></code>	187
6.101.6	Member Data Documentation	187
6.101.6.1	<code>rank_</code>	187
6.101.6.2	<code>raw_ptr_</code>	188
6.102	<code>upcxx::detail::global_ptr_ctor_internal</code> Struct Reference	188
6.102.1	Detailed Description	188
6.103	<code>std::greater< upcxx::global_ptr< T >></code> Struct Template Reference	188
6.103.1	Detailed Description	188
6.103.2	Member Function Documentation	188
6.103.2.1	<code>operator()</code>	189
6.104	<code>std::greater_equal< upcxx::global_ptr< T >></code> Struct Template Reference	189
6.104.1	Detailed Description	189
6.104.2	Member Function Documentation	189
6.104.2.1	<code>operator()</code>	189
6.105	<code>std::hash< upcxx::digest ></code> Struct Template Reference	190

6.105.1 Detailed Description	190
6.105.2 Member Function Documentation	190
6.105.2.1 operator()	190
6.106std::hash< upcxx::dist_id< T > > Struct Template Reference	190
6.106.1 Detailed Description	190
6.106.2 Member Function Documentation	191
6.106.2.1 operator()	191
6.107std::hash< upcxx::global_ptr< T > > Struct Template Reference	191
6.107.1 Detailed Description	191
6.107.2 Member Function Documentation	191
6.107.2.1 operator()	192
6.108upcxx::hasher_reflector< Hasher > Struct Template Reference	192
6.108.1 Detailed Description	193
6.108.2 Member Function Documentation	193
6.108.2.1 opaque()	193
6.108.2.2 operator()	193
6.108.3 Member Data Documentation	193
6.108.3.1 h	193
6.109upcxx::index_sequence<... > Struct Template Reference	194
6.109.1 Detailed Description	194
6.110upcxx::integer_golden_ratio< T > Struct Template Reference	194
6.110.1 Detailed Description	194
6.110.2 Member Data Documentation	194
6.110.2.1 value	194
6.111upcxx::integer_golden_ratio_bits< bit_n > Struct Template Reference	195
6.111.1 Detailed Description	195
6.112upcxx::integer_golden_ratio_bits< 32 > Struct Template Reference	195
6.112.1 Detailed Description	195
6.112.2 Member Data Documentation	195
6.112.2.1 value	195

6.113	upcxx::integer_golden_ratio_bits< 64 > Struct Template Reference	196
6.113.1	Detailed Description	196
6.113.2	Member Data Documentation	196
6.113.2.1	value	196
6.114	upcxx::detail::is_future_cx< Cx > Struct Template Reference	196
6.114.1	Detailed Description	197
6.115	upcxx::detail::is_future_cx< future_cx< ordinal > > Struct Template Reference	197
6.115.1	Detailed Description	197
6.116	std::less< upcxx::global_ptr< T > > Struct Template Reference	197
6.116.1	Detailed Description	198
6.116.2	Member Function Documentation	198
6.116.2.1	operator()	198
6.117	std::less_equal< upcxx::global_ptr< T > > Struct Template Reference	198
6.117.1	Detailed Description	198
6.117.2	Member Function Documentation	198
6.117.2.1	operator()	199
6.118	upcxx::detail::make_future< T > Struct Template Reference	199
6.118.1	Detailed Description	199
6.119	upcxx::detail::make_future_< trivial, T > Struct Template Reference	199
6.119.1	Detailed Description	199
6.120	upcxx::detail::make_future_< false, T... > Struct Template Reference	200
6.120.1	Detailed Description	200
6.120.2	Member Function Documentation	200
6.120.2.1	operator()	200
6.121	upcxx::detail::make_future_< true, T... > Struct Template Reference	200
6.121.1	Detailed Description	201
6.121.2	Member Function Documentation	201
6.121.2.1	operator()	201
6.122	upcxx::detail::make_index_sequence< n, s > Struct Template Reference	201
6.122.1	Detailed Description	201

6.123upcxx::detail::make_index_sequence< 0, s... > Struct Template Reference	201
6.123.1 Detailed Description	202
6.123.2 Member Typedef Documentation	202
6.123.2.1 type	202
6.124mallinfo Struct Reference	202
6.124.1 Detailed Description	202
6.124.2 Member Data Documentation	203
6.124.2.1 arena	203
6.124.2.2 fordblks	203
6.124.2.3 fsmblks	203
6.124.2.4 hblkhd	203
6.124.2.5 hblks	203
6.124.2.6 keepcost	204
6.124.2.7 ordblks	204
6.124.2.8 smblks	204
6.124.2.9 uordblks	204
6.124.2.10usmblks	204
6.125malloc_chunk Struct Reference	204
6.125.1 Detailed Description	205
6.125.2 Member Data Documentation	205
6.125.2.1 bk	205
6.125.2.2 fd	205
6.125.2.3 head	205
6.125.2.4 prev_foot	205
6.126malloc_params Struct Reference	206
6.126.1 Detailed Description	206
6.126.2 Member Data Documentation	206
6.126.2.1 default_mflags	206
6.126.2.2 granularity	206
6.126.2.3 magic	206

6.126.2.4 mmap_threshold	207
6.126.2.5 page_size	207
6.126.2.6 trim_threshold	207
6.127 malloc_segment Struct Reference	207
6.127.1 Detailed Description	207
6.127.2 Member Data Documentation	207
6.127.2.1 base	208
6.127.2.2 next	208
6.127.2.3 sflags	208
6.127.2.4 size	208
6.128 malloc_state Struct Reference	208
6.128.1 Detailed Description	209
6.128.2 Member Data Documentation	209
6.128.2.1 dv	209
6.128.2.2 dvsize	209
6.128.2.3 extp	209
6.128.2.4 exts	209
6.128.2.5 footprint	209
6.128.2.6 footprint_limit	210
6.128.2.7 least_addr	210
6.128.2.8 magic	210
6.128.2.9 max_footprint	210
6.128.2.10 mflags	210
6.128.2.11 mutex	210
6.128.2.12 release_checks	211
6.128.2.13 seg	211
6.128.2.14 smallbins	211
6.128.2.15 smallmap	211
6.128.2.16 top	211
6.128.2.17 topsize	211

6.128.2.18	<code>treebins</code>	212
6.128.2.19	<code>treemap</code>	212
6.128.2.20	<code>trim_check</code>	212
6.129	<code>malloc_tree_chunk</code> Struct Reference	212
6.129.1	Detailed Description	212
6.129.2	Member Data Documentation	212
6.129.2.1	<code>bk</code>	213
6.129.2.2	<code>child</code>	213
6.129.2.3	<code>fd</code>	213
6.129.2.4	<code>head</code>	213
6.129.2.5	<code>index</code>	213
6.129.2.6	<code>parent</code>	213
6.129.2.7	<code>prev_foot</code>	214
6.130	<code>upcxx::modn_hashing< T ></code> Struct Template Reference	214
6.130.1	Detailed Description	214
6.130.2	Member Function Documentation	214
6.130.2.1	<code>operator()()</code>	214
6.131	<code>upcxx::nil_cx</code> Struct Reference	214
6.131.1	Detailed Description	215
6.132	<code>upcxx::nop_function< Sig ></code> Struct Template Reference	215
6.132.1	Detailed Description	215
6.133	<code>upcxx::nop_function< Ret(Arg...)></code> Struct Template Reference	215
6.133.1	Detailed Description	215
6.133.2	Member Function Documentation	215
6.133.2.1	<code>operator()()</code>	216
6.134	<code>upcxx::nop_function< void(Arg...)></code> Struct Template Reference	216
6.134.1	Detailed Description	216
6.134.2	Member Function Documentation	216
6.134.2.1	<code>operator()()</code>	216
6.135	<code>upcxx::detail::os_env_parse< T ></code> Struct Template Reference	217

6.135.1 Detailed Description	217
6.135.2 Member Function Documentation	217
6.135.2.1 operator()	217
6.136upcxx::detail::os_env_parse< std::string > Struct Template Reference	217
6.136.1 Detailed Description	217
6.136.2 Member Function Documentation	218
6.136.2.1 operator()	218
6.137upcxx::packing< T > Struct Template Reference	218
6.137.1 Detailed Description	218
6.138upcxx::packing< bound_function< Fn, B... > > Struct Template Reference	218
6.138.1 Detailed Description	219
6.138.2 Member Function Documentation	219
6.138.2.1 pack()	219
6.138.2.2 size_ubound()	219
6.138.2.3 unpack()	219
6.139upcxx::packing< std::array< T, n > > Struct Template Reference	220
6.139.1 Detailed Description	220
6.139.2 Member Function Documentation	220
6.139.2.1 pack()	220
6.139.2.2 size_ubound()	220
6.139.2.3 unpack()	221
6.140upcxx::packing< std::pair< A, B > > Struct Template Reference	221
6.140.1 Detailed Description	221
6.140.2 Member Function Documentation	221
6.140.2.1 pack()	221
6.140.2.2 size_ubound()	222
6.140.2.3 unpack()	222
6.141upcxx::packing< std::string > Struct Template Reference	222
6.141.1 Detailed Description	222
6.141.2 Member Function Documentation	222

6.141.2.1 pack()	223
6.141.2.2 size_ubound()	223
6.141.2.3 unpack()	223
6.142upcxx::packing< std::tuple< T... > > Struct Template Reference	223
6.142.1 Detailed Description	223
6.142.2 Member Function Documentation	224
6.142.2.1 move_from_storage()	224
6.142.2.2 pack()	224
6.142.2.3 size_ubound()	224
6.142.2.4 unpack()	224
6.143upcxx::packing< std::unordered_map< K, V > > Struct Template Reference	225
6.143.1 Detailed Description	225
6.143.2 Member Function Documentation	225
6.143.2.1 pack()	225
6.143.2.2 size_ubound()	225
6.143.2.3 unpack()	226
6.144upcxx::packing< std::unordered_set< T > > Struct Template Reference	226
6.144.1 Detailed Description	226
6.144.2 Member Function Documentation	226
6.144.2.1 pack()	226
6.144.2.2 size_ubound()	227
6.144.2.3 unpack()	227
6.145upcxx::packing< std::vector< T > > Struct Template Reference	227
6.145.1 Detailed Description	227
6.145.2 Member Function Documentation	227
6.145.2.1 pack()	228
6.145.2.2 size_ubound()	228
6.145.2.3 unpack()	228
6.146upcxx::packing< T & > Struct Template Reference	228
6.146.1 Detailed Description	229

6.147	upcxx::packing< T && > Struct Template Reference	229
6.147.1	Detailed Description	229
6.148	upcxx::packing< T const > Struct Template Reference	229
6.148.1	Detailed Description	230
6.149	upcxx::packing< T volatile > Struct Template Reference	230
6.149.1	Detailed Description	230
6.150	upcxx::packing_empty< T, is_default_constructible > Struct Template Reference	230
6.150.1	Detailed Description	230
6.151	upcxx::packing_empty< T, false > Struct Template Reference	231
6.151.1	Detailed Description	231
6.151.2	Member Function Documentation	231
6.151.2.1	pack()	231
6.151.2.2	size_ubound()	231
6.151.2.3	unpack()	232
6.151.3	Member Data Documentation	232
6.151.3.1	is_trivial	232
6.152	upcxx::packing_empty< T, true > Struct Template Reference	232
6.152.1	Detailed Description	232
6.152.2	Member Function Documentation	233
6.152.2.1	pack()	233
6.152.2.2	size_ubound()	233
6.152.2.3	unpack()	233
6.152.3	Member Data Documentation	233
6.152.3.1	is_trivial	233
6.153	upcxx::packing_function_pointer< T > Struct Template Reference	234
6.153.1	Detailed Description	234
6.153.2	Member Function Documentation	234
6.153.2.1	pack()	234
6.153.2.2	size_ubound()	235
6.153.2.3	unpack()	235

6.153.3 Member Data Documentation	235
6.153.3.1 is_trivial	235
6.154upcxx::packing_is_trivial< T, false_ > Struct Template Reference	235
6.154.1 Detailed Description	235
6.154.2 Member Data Documentation	236
6.154.2.1 value	236
6.155upcxx::packing_is_trivial< T, std::integral_constant< bool, false &packing< T >::is_trivial > > Struct Template Reference	236
6.155.1 Detailed Description	236
6.155.2 Member Data Documentation	236
6.155.2.1 value	236
6.156upcxx::packing_not_supported< T > Struct Template Reference	237
6.156.1 Detailed Description	237
6.156.2 Member Function Documentation	237
6.156.2.1 size_ubound()	237
6.157upcxx::packing_opaque< T, is_empty, is_trivially_copyable > Struct Template Reference	237
6.157.1 Detailed Description	237
6.158upcxx::packing_opaque< T, false, false > Struct Template Reference	238
6.158.1 Detailed Description	238
6.159upcxx::packing_opaque< T, false, true > Struct Template Reference	238
6.159.1 Detailed Description	238
6.160upcxx::packing_opaque< T, true, is_trivially_copyable > Struct Template Reference	239
6.160.1 Detailed Description	239
6.161upcxx::packing_pack_reflector Struct Reference	239
6.161.1 Detailed Description	239
6.161.2 Member Function Documentation	240
6.161.2.1 opaque()	240
6.161.2.2 operator>()	240
6.161.3 Member Data Documentation	240
6.161.3.1 w	240
6.162upcxx::packing_reflected< T, is_default_constructible > Struct Template Reference	240

6.162.1 Detailed Description	241
6.163upcxx::packing_reflected< T, false > Struct Template Reference	241
6.163.1 Detailed Description	241
6.163.2 Member Function Documentation	241
6.163.2.1 pack()	241
6.163.2.2 size_ubound()	242
6.163.2.3 unpack()	242
6.164upcxx::packing_reflected< T, true > Struct Template Reference	242
6.164.1 Detailed Description	242
6.164.2 Member Function Documentation	242
6.164.2.1 pack()	243
6.164.2.2 size_ubound()	243
6.164.2.3 unpack()	243
6.165upcxx::packing_specializer< T, is_empty, is_funptr, is_scalar > Struct Template Reference	243
6.165.1 Detailed Description	243
6.166upcxx::packing_specializer< T, false, false, false > Struct Template Reference	244
6.166.1 Detailed Description	244
6.167upcxx::packing_specializer< T, false, false, true > Struct Template Reference	244
6.167.1 Detailed Description	244
6.168upcxx::packing_specializer< T, false, true, is_scalar > Struct Template Reference	245
6.168.1 Detailed Description	245
6.169upcxx::packing_specializer< T, true, is_funptr, is_scalar > Struct Template Reference	245
6.169.1 Detailed Description	245
6.170upcxx::packing_trivial< T > Struct Template Reference	246
6.170.1 Detailed Description	246
6.170.2 Member Function Documentation	246
6.170.2.1 pack()	246
6.170.2.2 size_ubound()	247
6.170.2.3 unpack()	247
6.170.3 Member Data Documentation	247

6.170.3.1 <code>is_trivial</code>	247
6.171 <code>upcxx::detail::packing_tuple_each< n, i, T ></code> Struct Template Reference	247
6.171.1 Detailed Description	248
6.171.2 Member Typedef Documentation	248
6.171.2.1 <code>Ti</code>	248
6.171.3 Member Function Documentation	248
6.171.3.1 <code>destruct()</code>	248
6.171.3.2 <code>pack()</code>	249
6.171.3.3 <code>size_ubound()</code>	249
6.171.3.4 <code>unpack_into()</code>	249
6.172 <code>upcxx::detail::packing_tuple_each< n, n, T... ></code> Struct Template Reference	249
6.172.1 Detailed Description	250
6.172.2 Member Function Documentation	250
6.172.2.1 <code>destruct()</code>	250
6.172.2.2 <code>pack()</code>	250
6.172.2.3 <code>size_ubound()</code>	250
6.172.2.4 <code>unpack_into()</code>	251
6.173 <code>upcxx::packing_ubound_reflector</code> Struct Reference	251
6.173.1 Detailed Description	251
6.173.2 Member Function Documentation	251
6.173.2.1 <code>opaque()</code>	251
6.173.2.2 <code>operator()</code>	252
6.173.3 Member Data Documentation	252
6.173.3.1 <code>ub</code>	252
6.174 <code>upcxx::packing_unpack_reflector< member_assignment_not_construction ></code> Struct Template Reference	252
6.174.1 Detailed Description	252
6.175 <code>upcxx::packing_unpack_reflector< false ></code> Struct Template Reference	252
6.175.1 Detailed Description	253
6.175.2 Member Function Documentation	253
6.175.2.1 <code>opaque()</code>	253

6.175.2.2 operator()	253
6.175.3 Member Data Documentation	253
6.175.3.1 r	254
6.176upcxx::packing_unpack_reflector< true > Struct Template Reference	254
6.176.1 Detailed Description	254
6.176.2 Member Function Documentation	254
6.176.2.1 opaque()	254
6.176.2.2 operator()	255
6.176.3 Member Data Documentation	255
6.176.3.1 r	255
6.177upcxx::parcel_layout Class Reference	255
6.177.1 Detailed Description	256
6.177.2 Constructor & Destructor Documentation	256
6.177.2.1 parcel_layout()	256
6.177.3 Member Function Documentation	256
6.177.3.1 add_bytes() [1/2]	256
6.177.3.2 add_bytes() [2/2]	256
6.177.3.3 add_trivial_aligned()	256
6.177.3.4 add_trivial_unaligned()	257
6.177.3.5 alignment()	257
6.177.3.6 embed()	257
6.177.3.7 size()	257
6.177.3.8 size_aligned()	257
6.177.4 Friends And Related Function Documentation	257
6.177.4.1 operator<<	258
6.178upcxx::parcel_reader Class Reference	258
6.178.1 Detailed Description	258
6.178.2 Constructor & Destructor Documentation	258
6.178.2.1 parcel_reader()	259
6.178.2.2 ~parcel_reader()	259

6.178.3 Member Function Documentation	259
6.178.3.1 buffer()	259
6.178.3.2 layout()	259
6.178.3.3 pop()	259
6.178.3.4 pop_char()	260
6.178.3.5 pop_int8()	260
6.178.3.6 pop_trivial_aligned() [1/2]	260
6.178.3.7 pop_trivial_aligned() [2/2]	260
6.178.3.8 pop_trivial_unaligned() [1/2]	260
6.178.3.9 pop_trivial_unaligned() [2/2]	260
6.178.3.10pop_uint8()	261
6.179upcxx::parcel_writer Struct Reference	261
6.179.1 Detailed Description	261
6.179.2 Constructor & Destructor Documentation	261
6.179.2.1 parcel_writer()	262
6.179.3 Member Function Documentation	262
6.179.3.1 alignment()	262
6.179.3.2 buffer()	262
6.179.3.3 layout()	262
6.179.3.4 put()	262
6.179.3.5 put_char()	263
6.179.3.6 put_int8()	263
6.179.3.7 put_trivial_aligned() [1/2]	263
6.179.3.8 put_trivial_aligned() [2/2]	263
6.179.3.9 put_trivial_unaligned() [1/2]	263
6.179.3.10put_trivial_unaligned() [2/2]	264
6.179.3.11put_uint8()	264
6.179.3.12size()	264
6.179.4 Member Data Documentation	264
6.179.4.1 buf_	264

6.179.4.2 lay_	264
6.180upcxx::print_proxy< T > Struct Template Reference	265
6.180.1 Detailed Description	265
6.180.2 Friends And Related Function Documentation	265
6.180.2.1 operator<<	265
6.180.3 Member Data Documentation	265
6.180.3.1 subject	265
6.181upcxx::print_reflector Struct Reference	266
6.181.1 Detailed Description	266
6.181.2 Member Function Documentation	266
6.181.2.1 opaque()	266
6.181.2.2 operator>()	266
6.181.3 Member Data Documentation	266
6.181.3.1 comma	267
6.181.3.2 o	267
6.182upcxx::promise< T > Class Template Reference	267
6.182.1 Detailed Description	267
6.183upcxx::promise_cx< T > Struct Template Reference	267
6.183.1 Detailed Description	268
6.183.2 Member Data Documentation	268
6.183.2.1 pro_	268
6.184upcxx::detail::promise_like< Fu > Struct Template Reference	268
6.184.1 Detailed Description	268
6.185upcxx::detail::promise_like< future1< Kind, T... > > Struct Template Reference	268
6.185.1 Detailed Description	269
6.185.2 Member Typedef Documentation	269
6.185.2.1 type	269
6.186upcxx::reflection< Subject > Struct Template Reference	269
6.186.1 Detailed Description	269
6.186.2 Member Function Documentation	270

6.186.2.1 operator()	270
6.187upcxx::reflection< std::array< T, n > > Struct Template Reference	270
6.187.1 Detailed Description	270
6.187.2 Member Function Documentation	270
6.187.2.1 operator()	270
6.188upcxx::reflection< std::pair< A, B > > Struct Template Reference	271
6.188.1 Detailed Description	271
6.188.2 Member Function Documentation	271
6.188.2.1 operator()	271
6.189upcxx::reflection< std::tuple< Ts... > > Struct Template Reference	271
6.189.1 Detailed Description	272
6.189.2 Member Function Documentation	272
6.189.2.1 operator()	272
6.190upcxx::reflection_tuple< Tup, i, n > Struct Template Reference	272
6.190.1 Detailed Description	272
6.190.2 Member Function Documentation	272
6.190.2.1 operator()	273
6.191upcxx::reflection_tuple< Tup, n, n > Struct Template Reference	273
6.191.1 Detailed Description	273
6.191.2 Member Function Documentation	273
6.191.2.1 operator()	273
6.192upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function Struct Reference	274
6.192.1 Detailed Description	274
6.192.2 Member Typedef Documentation	274
6.192.2.1 fn_return_t	274
6.192.3 Constructor & Destructor Documentation	275
6.192.3.1 result_lrefs_function() [1/3]	275
6.192.3.2 result_lrefs_function() [2/3]	275
6.192.3.3 result_lrefs_function() [3/3]	275
6.192.4 Member Function Documentation	275

6.192.4.1 operator()	275
6.192.4.2 operator=() [1/2]	276
6.192.4.3 operator=() [2/2]	276
6.192.5 Member Data Documentation	276
6.192.5.1 fn_return_getter_t	276
6.192.5.2 getter_	276
6.192.5.3 result_	277
6.193upcxx::detail::rget_byval< T > Struct Template Reference	277
6.193.1 Detailed Description	277
6.194upcxx::detail::rget_futures_of< Mode, R, O > Struct Template Reference	277
6.194.1 Detailed Description	277
6.194.2 Member Typedef Documentation	278
6.194.2.1 return_type	278
6.194.3 Constructor & Destructor Documentation	278
6.194.3.1 rget_futures_of()	278
6.194.4 Member Function Documentation	278
6.194.4.1 operator()	278
6.195upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > > Struct Template Reference	278
6.195.1 Detailed Description	279
6.195.2 Member Typedef Documentation	279
6.195.2.1 return_type	279
6.195.3 Constructor & Destructor Documentation	279
6.195.3.1 rget_futures_of()	279
6.195.4 Member Function Documentation	280
6.195.4.1 operator()	280
6.195.5 Member Data Documentation	280
6.195.5.1 fut_	280
6.196upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > > Struct Template Reference	280
6.196.1 Detailed Description	281
6.196.2 Member Typedef Documentation	281

6.196.2.1 return_type	281
6.196.3 Constructor & Destructor Documentation	281
6.196.3.1 rget_futures_of()	281
6.196.4 Member Function Documentation	281
6.196.4.1 operator()	281
6.196.5 Member Data Documentation	282
6.196.5.1 fut_	282
6.197upcxx::detail::rget_state_operxn< Mode, Cx > Struct Template Reference	282
6.197.1 Detailed Description	282
6.198upcxx::detail::rget_state_operxn< rget_byref, future_cx< ordinal > > Struct Template Reference	282
6.198.1 Detailed Description	283
6.198.2 Constructor & Destructor Documentation	283
6.198.2.1 rget_state_operxn()	283
6.198.3 Member Function Documentation	283
6.198.3.1 completed()	283
6.198.4 Member Data Documentation	283
6.198.4.1 pro	283
6.199upcxx::detail::rget_state_operxn< rget_byref, nil_cx > Struct Template Reference	284
6.199.1 Detailed Description	284
6.199.2 Constructor & Destructor Documentation	284
6.199.2.1 rget_state_operxn()	284
6.199.3 Member Function Documentation	284
6.199.3.1 completed()	284
6.200upcxx::detail::rget_state_operxn< rget_byref, promise_cx<> > Struct Template Reference	285
6.200.1 Detailed Description	285
6.200.2 Constructor & Destructor Documentation	285
6.200.2.1 rget_state_operxn()	285
6.200.3 Member Function Documentation	285
6.200.3.1 completed()	285
6.200.4 Member Data Documentation	286

6.200.4.1 pro	286
6.201upcxx::detail::rget_state_operxn< rget_byval< T >, future_cx< ordinal > > Struct Template Reference	286
6.201.1 Detailed Description	286
6.201.2 Constructor & Destructor Documentation	286
6.201.2.1 rget_state_operxn()	287
6.201.3 Member Function Documentation	287
6.201.3.1 completed()	287
6.201.4 Member Data Documentation	287
6.201.4.1 pro	287
6.202upcxx::detail::rget_state_operxn< rget_byval< T >, nil_cx > Struct Template Reference	287
6.202.1 Detailed Description	288
6.202.2 Constructor & Destructor Documentation	288
6.202.2.1 rget_state_operxn()	288
6.202.3 Member Function Documentation	288
6.202.3.1 completed()	288
6.203upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T > > Struct Template Reference	288
6.203.1 Detailed Description	289
6.203.2 Constructor & Destructor Documentation	289
6.203.2.1 rget_state_operxn()	289
6.203.3 Member Function Documentation	289
6.203.3.1 completed()	289
6.203.4 Member Data Documentation	289
6.203.4.1 pro	289
6.204upcxx::detail::rget_state_remote< Cx > Struct Template Reference	290
6.204.1 Detailed Description	290
6.205upcxx::detail::rget_state_remote< nil_cx > Struct Template Reference	290
6.205.1 Detailed Description	290
6.205.2 Constructor & Destructor Documentation	290
6.205.2.1 rget_state_remote()	290
6.205.3 Member Function Documentation	291

6.205.3.1 completed()	291
6.206upcxx::detail::rget_state_remote< rpc_cx< Fn > > Struct Template Reference	291
6.206.1 Detailed Description	291
6.206.2 Constructor & Destructor Documentation	291
6.206.2.1 rget_state_remote()	291
6.206.3 Member Function Documentation	292
6.206.3.1 completed()	292
6.206.4 Member Data Documentation	292
6.206.4.1 fn	292
6.207upcxx::detail::rget_states< Mode, R, O > Struct Template Reference	292
6.207.1 Detailed Description	293
6.207.2 Constructor & Destructor Documentation	293
6.207.2.1 rget_states()	293
6.207.3 Member Data Documentation	293
6.207.3.1 o	293
6.207.3.2 owner	293
6.207.3.3 r	294
6.208upcxx::backend::gasnet1_seq::rma_cb Struct Reference	294
6.208.1 Detailed Description	294
6.208.2 Member Function Documentation	294
6.208.2.1 fire_and_delete()	294
6.208.3 Member Data Documentation	295
6.208.3.1 handle	295
6.208.3.2 next	295
6.209upcxx::backend::rma_get_cb Struct Reference	295
6.209.1 Detailed Description	295
6.210upcxx::backend::gasnet1_seq::rma_get_cb_impl< State, OpCx > Struct Template Reference	296
6.210.1 Detailed Description	296
6.210.2 Constructor & Destructor Documentation	296
6.210.2.1 rma_get_cb_impl()	296

6.210.3 Member Data Documentation	297
6.210.3.1 op_cx	297
6.211upcxx::backend::rma_get_cb_wstate< State > Struct Template Reference	297
6.211.1 Detailed Description	297
6.211.2 Constructor & Destructor Documentation	298
6.211.2.1 rma_get_cb_wstate()	298
6.211.3 Member Data Documentation	298
6.211.3.1 state	298
6.212upcxx::backend::rma_put_cb Struct Reference	298
6.212.1 Detailed Description	299
6.213upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx > Struct Template Reference	299
6.213.1 Detailed Description	299
6.213.2 Constructor & Destructor Documentation	300
6.213.2.1 rma_put_cb_impl()	300
6.213.3 Member Data Documentation	300
6.213.3.1 op_cx	300
6.213.3.2 src_cx [1/2]	300
6.213.3.3 src_cx [2/2]	300
6.214upcxx::backend::rma_put_cb_wstate< State > Struct Template Reference	301
6.214.1 Detailed Description	301
6.214.2 Constructor & Destructor Documentation	301
6.214.2.1 rma_put_cb_wstate()	301
6.214.3 Member Data Documentation	301
6.214.3.1 state	302
6.215upcxx::rpc_cx< Fn > Struct Template Reference	302
6.215.1 Detailed Description	302
6.215.2 Constructor & Destructor Documentation	302
6.215.2.1 rpc_cx()	302
6.215.3 Member Data Documentation	302
6.215.3.1 fn_	303

6.216upcxx::detail::rpc_recipient_after< Pro > Struct Template Reference	303
6.216.1 Detailed Description	303
6.216.2 Member Function Documentation	303
6.216.2.1 operator()	303
6.216.3 Member Data Documentation	304
6.216.3.1 initiator_	304
6.216.3.2 pro_	304
6.217upcxx::detail::rpc_return< ValidType, Fn, Args > Struct Template Reference	304
6.217.1 Detailed Description	304
6.217.2 Member Typedef Documentation	304
6.217.2.1 type	305
6.218upcxx::detail::rput_byval< T > Struct Template Reference	305
6.218.1 Detailed Description	305
6.219upcxx::detail::rput_futures_of< S, R, O > Struct Template Reference	305
6.219.1 Detailed Description	306
6.219.2 Member Typedef Documentation	306
6.219.2.1 return_type	306
6.219.3 Constructor & Destructor Documentation	306
6.219.3.1 rput_futures_of()	306
6.219.4 Member Function Documentation	306
6.219.4.1 operator()	306
6.220upcxx::detail::rput_futures_of< future_cx< S_ord >, R, future_cx< O_ord > > Struct Template Reference	307
6.220.1 Detailed Description	307
6.220.2 Member Typedef Documentation	307
6.220.2.1 return_type	307
6.220.3 Constructor & Destructor Documentation	307
6.220.3.1 rput_futures_of()	308
6.220.4 Member Function Documentation	308
6.220.4.1 operator()	308
6.220.5 Member Data Documentation	308

6.220.5.1 o_	308
6.220.5.2 s_	308
6.221upcxx::detail::rput_futures_of< future_cx< S_ord >, R, O > Struct Template Reference	309
6.221.1 Detailed Description	309
6.221.2 Member Typedef Documentation	309
6.221.2.1 return_type	309
6.221.3 Constructor & Destructor Documentation	309
6.221.3.1 rput_futures_of()	310
6.221.4 Member Function Documentation	310
6.221.4.1 operator()	310
6.221.5 Member Data Documentation	310
6.221.5.1 fut_	310
6.222upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > > Struct Template Reference	310
6.222.1 Detailed Description	311
6.222.2 Member Typedef Documentation	311
6.222.2.1 return_type	311
6.222.3 Constructor & Destructor Documentation	311
6.222.3.1 rput_futures_of()	311
6.222.4 Member Function Documentation	311
6.222.4.1 operator()	312
6.222.5 Member Data Documentation	312
6.222.5.1 fut_	312
6.223upcxx::detail::rput_state_here< Cx > Struct Template Reference	312
6.223.1 Detailed Description	312
6.224upcxx::detail::rput_state_here< future_cx< ordinal > > Struct Template Reference	312
6.224.1 Detailed Description	313
6.224.2 Constructor & Destructor Documentation	313
6.224.2.1 rput_state_here()	313
6.224.3 Member Function Documentation	313
6.224.3.1 completed()	313

6.224.4 Member Data Documentation	313
6.224.4.1 pro	314
6.225upcxx::detail::rput_state_here< nil_cx > Struct Template Reference	314
6.225.1 Detailed Description	314
6.225.2 Constructor & Destructor Documentation	314
6.225.2.1 rput_state_here()	314
6.225.3 Member Function Documentation	314
6.225.3.1 completed()	315
6.226upcxx::detail::rput_state_here< promise_cx<> > Struct Template Reference	315
6.226.1 Detailed Description	315
6.226.2 Constructor & Destructor Documentation	315
6.226.2.1 rput_state_here()	315
6.226.3 Member Function Documentation	315
6.226.3.1 completed()	316
6.226.4 Member Data Documentation	316
6.226.4.1 pro	316
6.227upcxx::detail::rput_state_remote< Cx > Struct Template Reference	316
6.227.1 Detailed Description	316
6.228upcxx::detail::rput_state_remote< nil_cx > Struct Template Reference	316
6.228.1 Detailed Description	317
6.228.2 Constructor & Destructor Documentation	317
6.228.2.1 rput_state_remote()	317
6.228.3 Member Function Documentation	317
6.228.3.1 completed()	317
6.229upcxx::detail::rput_state_remote< rpc_cx< Fn > > Struct Template Reference	317
6.229.1 Detailed Description	318
6.229.2 Constructor & Destructor Documentation	318
6.229.2.1 rput_state_remote()	318
6.229.3 Member Function Documentation	318
6.229.3.1 completed()	318

6.229.4 Member Data Documentation	318
6.229.4.1 fn	318
6.230upcxx::detail::rput_states< Mode, S, R, O > Struct Template Reference	319
6.230.1 Detailed Description	319
6.231upcxx::detail::rput_states< rput_byref, S, R, O > Struct Template Reference	319
6.231.1 Detailed Description	319
6.231.2 Constructor & Destructor Documentation	320
6.231.2.1 rput_states()	320
6.231.3 Member Data Documentation	320
6.231.3.1 o	320
6.231.3.2 r	320
6.231.3.3 s	320
6.231.3.4 target	321
6.232upcxx::detail::rput_states< rput_byval< T >, S, R, O > Struct Template Reference	321
6.232.1 Detailed Description	321
6.232.2 Constructor & Destructor Documentation	321
6.232.2.1 rput_states()	322
6.232.3 Member Data Documentation	322
6.232.3.1 buffer	322
6.233upcxx::trait_all< Tr > Struct Template Reference	322
6.233.1 Detailed Description	322
6.234upcxx::trait_all< Tr0, Trs... > Struct Template Reference	322
6.234.1 Detailed Description	323
6.235upcxx::trait_all<> Struct Template Reference	323
6.235.1 Detailed Description	323
6.235.2 Member Typedef Documentation	323
6.235.2.1 type	323
6.236upcxx::trait_any< Tr > Struct Template Reference	324
6.236.1 Detailed Description	324
6.237upcxx::trait_any< Tr0, Trs... > Struct Template Reference	324

6.237.1 Detailed Description	324
6.238upcxx::trait_any<> Struct Template Reference	324
6.238.1 Detailed Description	325
6.238.2 Member Typedef Documentation	325
6.238.2.1 type	325
6.239upcxx::trait_forall< Test, T > Struct Template Reference	325
6.239.1 Detailed Description	325
6.240upcxx::trait_forall< Test > Struct Template Reference	325
6.240.1 Detailed Description	326
6.240.2 Member Data Documentation	326
6.240.2.1 value	326
6.241upcxx::trait_forall< Test, T, Ts... > Struct Template Reference	326
6.241.1 Detailed Description	326
6.241.2 Member Data Documentation	326
6.241.2.1 value	327
6.242upcxx::trait_forall_tupled< Test, Tuple > Struct Template Reference	327
6.242.1 Detailed Description	327
6.243upcxx::trait_forall_tupled< Test, std::tuple< T... > > Struct Template Reference	327
6.243.1 Detailed Description	327
6.243.2 Member Data Documentation	328
6.243.2.1 value	328
6.244upcxx::detail::tuple_get_or_void< i, TupRef, in_range > Struct Template Reference	328
6.244.1 Detailed Description	328
6.244.2 Member Function Documentation	328
6.244.2.1 operator()	328
6.245upcxx::detail::tuple_get_or_void< i, TupRef, false > Struct Template Reference	329
6.245.1 Detailed Description	329
6.245.2 Member Function Documentation	329
6.245.2.1 operator()	329
6.246upcxx::detail::tuple_rvals_get< i, Tup > Struct Template Reference	329

6.246.1 Detailed Description	329
6.247upcxx::detail::tuple_rvals_get< Tup &&, i, Ti & > Struct Template Reference	330
6.247.1 Detailed Description	330
6.247.2 Member Function Documentation	330
6.247.2.1 operator()	330
6.248upcxx::detail::tuple_rvals_get< Tup &&, i, Ti && > Struct Template Reference	330
6.248.1 Detailed Description	330
6.248.2 Member Function Documentation	331
6.248.2.1 operator()	331
6.249upcxx::detail::tuple_rvals_get< Tup &&, i, Ti > Struct Template Reference	331
6.249.1 Detailed Description	331
6.249.2 Member Function Documentation	331
6.249.2.1 operator()	331
6.250upcxx::detail::tuple_rvals_get< Tup &, i, Ti & > Struct Template Reference	332
6.250.1 Detailed Description	332
6.250.2 Member Function Documentation	332
6.250.2.1 operator()	332
6.251upcxx::detail::tuple_rvals_get< Tup &, i, Ti && > Struct Template Reference	332
6.251.1 Detailed Description	332
6.251.2 Member Function Documentation	333
6.251.2.1 operator()	333
6.252upcxx::detail::tuple_rvals_get< Tup &, i, Ti > Struct Template Reference	333
6.252.1 Detailed Description	333
6.252.2 Member Function Documentation	333
6.252.2.1 operator()	333
6.253upcxx::detail::tuple_rvals_get< Tup const &, i, Ti & > Struct Template Reference	334
6.253.1 Detailed Description	334
6.253.2 Member Function Documentation	334
6.253.2.1 operator()	334
6.254upcxx::detail::tuple_rvals_get< Tup const &, i, Ti && > Struct Template Reference	334

6.254.1 Detailed Description	334
6.254.2 Member Function Documentation	335
6.254.2.1 operator()	335
6.255upcxx::detail::tuple_rvals_get< Tup const &, i, Ti > Struct Template Reference	335
6.255.1 Detailed Description	335
6.255.2 Member Function Documentation	335
6.255.2.1 operator()	335
6.256upcxx::tuple_types_into< Tuple, Into > Struct Template Reference	336
6.256.1 Detailed Description	336
6.257upcxx::tuple_types_into< std::tuple< T... >, Into > Struct Template Reference	336
6.257.1 Detailed Description	336
6.257.2 Member Typedef Documentation	336
6.257.2.1 type	336
6.258upcxx::trait_all< Tr0, Trs... >::type< T > Struct Template Reference	337
6.258.1 Detailed Description	337
6.258.2 Member Data Documentation	337
6.258.2.1 value	337
6.259upcxx::trait_any< Tr0, Trs... >::type< T > Struct Template Reference	337
6.259.1 Detailed Description	338
6.259.2 Member Data Documentation	338
6.259.2.1 value	338

7 File Documentation	339
7.1 allocate.hpp File Reference	339
7.2 allreduce.hpp File Reference	340
7.3 atomic.hpp File Reference	340
7.4 backend/gasnet1_seq/backend.cpp File Reference	341
7.4.1 Enumeration Type Documentation	341
7.4.1.1 anonymous enum	341
7.5 backend/gasnet1_seq/backend.hpp File Reference	341
7.6 backend.hpp File Reference	342
7.6.1 Macro Definition Documentation	344
7.6.1.1 gasnet1_seq	344
7.7 bind.hpp File Reference	344
7.8 broadcast.hpp File Reference	345
7.9 command.hpp File Reference	345
7.10 completion.hpp File Reference	346
7.11 diagnostic.cpp File Reference	347
7.12 diagnostic.hpp File Reference	347
7.12.1 Macro Definition Documentation	348
7.12.1.1 UPCXX_ASSERT	348
7.12.1.2 UPCXX_ASSERT_1	348
7.12.1.3 UPCXX_ASSERT_2	349
7.12.1.4 UPCXX_ASSERT_ALWAYS	349
7.12.1.5 UPCXX_ASSERT_DISPATCH	349
7.12.1.6 UPCXX_INVOKE_UB	349
7.13 digest.cpp File Reference	349
7.14 digest.hpp File Reference	350
7.15 dist_object.cpp File Reference	350
7.16 dist_object.hpp File Reference	350
7.16.1 Macro Definition Documentation	351
7.16.1.1 COMPARATOR	351

7.17 dl_malloc.c File Reference	351
7.17.1 Macro Definition Documentation	356
7.17.1.1 _STRUCT_MALLINFO	356
7.17.1.2 ABORT	357
7.17.1.3 ABORT_ON_ASSERT_FAILURE	357
7.17.1.4 ACQUIRE_LOCK	357
7.17.1.5 ACQUIRE_MALLOC_GLOBAL_LOCK	357
7.17.1.6 align_as_chunk	357
7.17.1.7 align_offset	357
7.17.1.8 assert	358
7.17.1.9 bit_for_tree_index	358
7.17.1.10 CALL_DIRECT_MMAP	358
7.17.1.11 CALL_MMAP	358
7.17.1.12 CALL_MORECORE	358
7.17.1.13 CALL_MREMAP	359
7.17.1.14 CALL_MUNMAP	359
7.17.1.15 calloc_must_clear	359
7.17.1.16 check_free_chunk	359
7.17.1.17 check_inuse_chunk	359
7.17.1.18 check_malloc_state	360
7.17.1.19 check_mallosed_chunk	360
7.17.1.20 check_mmapped_chunk	360
7.17.1.21 check_top_chunk	360
7.17.1.22 chunk2mem	360
7.17.1.23 CHUNK_ALIGN_MASK	361
7.17.1.24 chunk_minus_offset	361
7.17.1.25 CHUNK_OVERHEAD	361
7.17.1.26 chunk_plus_offset	361
7.17.1.27 chunksize	361
7.17.1.28 cinuse	361

7.17.1.29 CINUSE_BIT	362
7.17.1.30 clear_flag4	362
7.17.1.31 clear_pinuse	362
7.17.1.32 clear_smallmap	362
7.17.1.33 clear_treemap	362
7.17.1.34 CMFAIL	363
7.17.1.35 compute_bit2idx	363
7.17.1.36 compute_tree_index	363
7.17.1.37 CORRUPTION_ERROR_ACTION	364
7.17.1.38 DEBUG	364
7.17.1.39 DEFAULT_GRANULARITY	364
7.17.1.40 DEFAULT_MMAP_THRESHOLD	364
7.17.1.41 DEFAULT_TRIM_THRESHOLD	364
7.17.1.42 DESTROY_LOCK	364
7.17.1.43 DIRECT_MMAP_DEFAULT	365
7.17.1.44 disable_contiguous	365
7.17.1.45 disable_lock	365
7.17.1.46 disable_mmap	365
7.17.1.47 DLMALLOC_EXPORT	365
7.17.1.48 DLMALLOC_VERSION	365
7.17.1.49 enable_lock	366
7.17.1.50 enable_mmap	366
7.17.1.51 ensure_initialization	366
7.17.1.52 EXTERN_BIT	366
7.17.1.53 FENCEPOST_HEAD	366
7.17.1.54 FLAG4_BIT	366
7.17.1.55 flag4inuse	367
7.17.1.56 FLAG_BITS	367
7.17.1.57 FOOTERS	367
7.17.1.58 FORCEINLINE	367

7.17.1.59 FOUR_SIZE_T_SIZES	367
7.17.1.60 get_foot	367
7.17.1.61 granularity_align	368
7.17.1.62 HALF_MAX_SIZE_T	368
7.17.1.63 HAVE_MMAP	368
7.17.1.64 HAVE_MORECORE	368
7.17.1.65 HAVE_MREMAP	368
7.17.1.66 idx2bit	369
7.17.1.67 INITIAL_LOCK	369
7.17.1.68 INSECURE	369
7.17.1.69 insert_chunk	369
7.17.1.70 insert_large_chunk	369
7.17.1.71 insert_small_chunk	370
7.17.1.72 internal_free	370
7.17.1.73 internal_malloc	370
7.17.1.74 INUSE_BITS	370
7.17.1.75 is_aligned	371
7.17.1.76 is_extern_segment	371
7.17.1.77 is_granularity_aligned	371
7.17.1.78 is_initialized	371
7.17.1.79 is_inuse	371
7.17.1.80 is_mmapped	371
7.17.1.81 is_mmapped_segment	372
7.17.1.82 is_page_aligned	372
7.17.1.83 is_small	372
7.17.1.84 least_bit	372
7.17.1.85 left_bits	372
7.17.1.86 leftmost_child	372
7.17.1.87 leftshift_for_tree_index	373
7.17.1.88 LOCK_AT_FORK	373

7.17.1.89 M_GRANULARITY	373
7.17.1.90 M_MMAP_THRESHOLD	373
7.17.1.91 M_TRIM_THRESHOLD	373
7.17.1.92 MALLINFO_FIELD_TYPE	374
7.17.1.93 MALLOC_ALIGNMENT	374
7.17.1.94 MALLOC_FAILURE_ACTION	374
7.17.1.95 malloc_getpagesize	374
7.17.1.96 MALLOC_INSPECT_ALL	374
7.17.1.97 mark_inuse_foot	374
7.17.1.98 mark_smallmap	375
7.17.1.99 mark_treemap	375
7.17.1.100 MAX_RELEASE_CHECK_RATE	375
7.17.1.101 MAX_REQUEST	375
7.17.1.102 MAX_SIZE_T	375
7.17.1.103 MAX_SMALL_REQUEST	375
7.17.1.104 MAX_SMALL_SIZE	376
7.17.1.105 MCHUNK_SIZE	376
7.17.1.106 mem2chunk	376
7.17.1.107 MFAIL	376
7.17.1.108 MIN_CHUNK_SIZE	376
7.17.1.109 MIN_LARGE_SIZE	376
7.17.1.110 MIN_REQUEST	377
7.17.1.111 MIN_SMALL_INDEX	377
7.17.1.112 minsize_for_tree_index	377
7.17.1.113 MLOCK_T	377
7.17.1.114 mmap_align	377
7.17.1.115 MMAP_CHUNK_OVERHEAD	378
7.17.1.116 MMAP_CLEARS	378
7.17.1.117 MMAP_DEFAULT	378
7.17.1.118 MMAP_FLAGS	378

7.17.1.119	MMAP_FOOT_PAD	378
7.17.1.120	MMAP_PROT	379
7.17.1.121	MORECORE_CONTIGUOUS	379
7.17.1.122	MSPACES	379
7.17.1.123	MUNMAP_DEFAULT	379
7.17.1.124	next_chunk	379
7.17.1.125	next_pinuse	379
7.17.1.126	NO_MALLINFO	380
7.17.1.127	NO_MALLOC_STATS	380
7.17.1.128	NO_SEGMENT_TRAVERSAL	380
7.17.1.129	NOINLINE	380
7.17.1.130	NSMALLBINS	380
7.17.1.131	NTREEBINS	380
7.17.1.132	bk_address	381
7.17.1.133	bk_inuse	381
7.17.1.134	bk_magic	381
7.17.1.135	bk_next	381
7.17.1.136	bk_pinuse	381
7.17.1.137	ONLY_MSPACES	382
7.17.1.138	overhead_for	382
7.17.1.139	pad_request	382
7.17.1.140	page_align	382
7.17.1.141	pinuse	382
7.17.1.142	PINUSE_BIT	382
7.17.1.143	POSTACTION	383
7.17.1.144	PREACTION	383
7.17.1.145	prev_chunk	383
7.17.1.146	PROCEED_ON_ERROR	383
7.17.1.147	RELEASE_LOCK	383
7.17.1.148	RELEASE_MALLOC_GLOBAL_LOCK	383

7.17.1.149	<code>replace_dv</code>	384
7.17.1.150	<code>request2size</code>	384
7.17.1.151	<code>RTCHECK</code>	384
7.17.1.152	<code>same_or_left_bits</code>	384
7.17.1.153	<code>segment_holds</code>	385
7.17.1.154	<code>set_flag4</code>	385
7.17.1.155	<code>set_foot</code>	385
7.17.1.156	<code>set_free_with_pinuse</code>	385
7.17.1.157	<code>set_inuse</code>	385
7.17.1.158	<code>set_inuse_and_pinuse</code>	386
7.17.1.159	<code>set_lock</code>	386
7.17.1.160	<code>set_size_and_pinuse_of_free_chunk</code>	386
7.17.1.161	<code>set_size_and_pinuse_of_inuse_chunk</code>	386
7.17.1.162	<code>should_trim</code>	387
7.17.1.163	<code>SIX_SIZE_T_SIZES</code>	387
7.17.1.164	<code>SIZE_T_BITSIZE</code>	387
7.17.1.165	<code>SIZE_T_FOUR</code>	387
7.17.1.166	<code>SIZE_T_ONE</code>	387
7.17.1.167	<code>SIZE_T_SIZE</code>	387
7.17.1.168	<code>SIZE_T_TWO</code>	388
7.17.1.169	<code>SIZE_T_ZERO</code>	388
7.17.1.170	<code>small_index</code>	388
7.17.1.171	<code>small_index2size</code>	388
7.17.1.172	<code>smallbin_at</code>	388
7.17.1.173	<code>SMALLBIN_SHIFT</code>	388
7.17.1.174	<code>SMALLBIN_WIDTH</code>	389
7.17.1.175	<code>smallmap_is_marked</code>	389
7.17.1.176	<code>STRUCT_MALLINFO_DECLARED</code>	389
7.17.1.177	<code>SYS_ALLOC_PADDING</code>	389
7.17.1.178	<code>TOP_FOOT_SIZE</code>	389

7.17.1.179	treebin_at	389
7.17.1.180	TREEBIN_SHIFT	390
7.17.1.181	treemap_is_marked	390
7.17.1.182	TRY_LOCK	390
7.17.1.183	TWO_SIZE_T_SIZES	390
7.17.1.184	unlink_chunk	390
7.17.1.185	unlink_first_small_chunk	391
7.17.1.186	unlink_large_chunk	391
7.17.1.187	unlink_small_chunk	391
7.17.1.188	USAGE_ERROR_ACTION	392
7.17.1.189	USE_BUILTIN_FFS	392
7.17.1.190	USE_DEV_RANDOM	392
7.17.1.191	use_lock	392
7.17.1.192	USE_LOCK_BIT	393
7.17.1.193	USE_LOCKS	393
7.17.1.194	use_mmap	393
7.17.1.195	USE_MMAP_BIT	393
7.17.1.196	use_noncontiguous	393
7.17.1.197	USE_NONCONTIGUOUS_BIT	394
7.17.1.198	USE_SPIN_LOCKS	394
7.17.2	Typedef Documentation	394
7.17.2.1	bindex_t	394
7.17.2.2	binmap_t	394
7.17.2.3	flag_t	394
7.17.2.4	mchunk	394
7.17.2.5	mchunkptr	395
7.17.2.6	msegment	395
7.17.2.7	msegmentptr	395
7.17.2.8	mspace	395
7.17.2.9	mstate	395

7.17.2.10	sbinptr	395
7.17.2.11	tbinptr	396
7.17.2.12	tchunk	396
7.17.2.13	tchunkptr	396
7.17.3	Function Documentation	396
7.17.3.1	create_mspace()	396
7.17.3.2	create_mspace_with_base()	396
7.17.3.3	destroy_mspace()	397
7.17.3.4	mspace_bulk_free()	397
7.17.3.5	mspace_calloc()	397
7.17.3.6	mspace_footprint()	397
7.17.3.7	mspace_footprint_limit()	397
7.17.3.8	mspace_free()	398
7.17.3.9	mspace_independent_calloc()	398
7.17.3.10	mspace_independent_comalloc()	398
7.17.3.11	mspace_mallinfo()	398
7.17.3.12	mspace_malloc()	398
7.17.3.13	mspace_malloc_stats()	399
7.17.3.14	mspace_mallopt()	399
7.17.3.15	mspace_max_footprint()	399
7.17.3.16	mspace_memalign()	399
7.17.3.17	mspace_realloc()	399
7.17.3.18	mspace_realloc_in_place()	400
7.17.3.19	mspace_set_footprint_limit()	400
7.17.3.20	mspace_track_large_chunks()	400
7.17.3.21	mspace_trim()	400
7.17.3.22	mspace_usable_size()	400
7.18	dl_malloc.h File Reference	401
7.18.1	Macro Definition Documentation	402
7.18.1.1	MALLINFO_FIELD_TYPE	402

7.18.1.2	MSPACES	402
7.18.1.3	NO_MALLINFO	402
7.18.1.4	ONLY_MSPACES [1/2]	402
7.18.1.5	ONLY_MSPACES [2/2]	402
7.18.1.6	STRUCT_MALLINFO_DECLARED	402
7.18.2	Typedef Documentation	403
7.18.2.1	mSPACE	403
7.18.3	Function Documentation	403
7.18.3.1	create_mSPACE()	403
7.18.3.2	create_mSPACE_with_base()	403
7.18.3.3	destroy_mSPACE()	403
7.18.3.4	dMalloc_usable_size()	404
7.18.3.5	mSPACE_bulk_free()	404
7.18.3.6	mSPACE_calloc()	404
7.18.3.7	mSPACE_footprint()	404
7.18.3.8	mSPACE_footprint_limit()	404
7.18.3.9	mSPACE_free()	405
7.18.3.10	mSPACE_independent_calloc()	405
7.18.3.11	mSPACE_independent_comalloc()	405
7.18.3.12	mSPACE_inspect_all()	405
7.18.3.13	mSPACE_mallinfo()	405
7.18.3.14	mSPACE_malloc()	406
7.18.3.15	mSPACE_malloc_stats()	406
7.18.3.16	mSPACE_mallopt()	406
7.18.3.17	mSPACE_max_footprint()	406
7.18.3.18	mSPACE_memalign()	406
7.18.3.19	mSPACE_realloc()	407
7.18.3.20	mSPACE_realloc_in_place()	407
7.18.3.21	mSPACE_set_footprint_limit()	407
7.18.3.22	mSPACE_track_large_chunks()	407

7.18.3.23	<code>mSPACE_trim()</code>	407
7.18.3.24	<code>mSPACE_usable_size()</code>	408
7.19	<code>future.hpp</code> File Reference	408
7.20	<code>future/apply.hpp</code> File Reference	408
7.21	<code>future/body_pure.hpp</code> File Reference	409
7.22	<code>future/core.cpp</code> File Reference	409
7.22.1	Typedef Documentation	409
7.22.1.1	<code>future_header_result</code>	409
7.23	<code>future/core.hpp</code> File Reference	409
7.24	<code>future/future1.hpp</code> File Reference	410
7.25	<code>future/impl_mapped.hpp</code> File Reference	411
7.26	<code>future/impl_result.hpp</code> File Reference	411
7.27	<code>future/impl_shref.hpp</code> File Reference	412
7.28	<code>future/impl_when_all.hpp</code> File Reference	412
7.29	<code>future/make_future.hpp</code> File Reference	412
7.30	<code>future/promise.hpp</code> File Reference	413
7.31	<code>future/then.hpp</code> File Reference	414
7.32	<code>future/when_all.hpp</code> File Reference	414
7.33	<code>global_ptr.hpp</code> File Reference	415
7.34	<code>nobsrule.py</code> File Reference	415
7.35	<code>os_env.hpp</code> File Reference	416
7.36	<code>packing.cpp</code> File Reference	416
7.37	<code>packing.hpp</code> File Reference	416
7.38	<code>reflection.hpp</code> File Reference	418
7.38.1	Function Documentation	418
7.38.1.1	<code>reflect()</code>	419
7.39	<code>rget.hpp</code> File Reference	419
7.40	<code>rpc.hpp</code> File Reference	420
7.41	<code>rput.hpp</code> File Reference	420
7.42	<code>upcxx.cpp</code> File Reference	421
7.43	<code>upcxx.hpp</code> File Reference	421
7.44	<code>utility.hpp</code> File Reference	421
7.45	<code>wait.hpp</code> File Reference	423

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

nobsrule	17
std	17
upcxx	18
upcxx::backend	40
upcxx::backend::gasnet1_seq	43
upcxx::detail	46

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

upcxx::backend::gasnet1_seq::action	53
upcxx::backend::gasnet1_seq::action_impl< Fn >	54
upcxx::add_lref_if_nonref< T >	55
upcxx::add_lref_if_nonref< T & >	56
upcxx::add_lref_if_nonref< T && >	57
upcxx::detail::allreduce_state< T, Op >	57
upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator< Arg >	59
upcxx::detail::apply_futered_as_future< Fn(future1< Kind, T... >)>	61
upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, ArgTupDecayed >	63
upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup >	63
upcxx::detail::apply_tupled_as_future< Fn, ArgTup >	63
upcxx::detail::apply_tupled_as_future_impl< Return >	64
upcxx::detail::apply_tupled_as_future_impl< future1< Kind, T... > >	65
upcxx::detail::apply_tupled_as_future_impl< std::result_of< Fn(T...)>::type >	64
upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, std::tuple< T... > >	64
upcxx::detail::apply_tupled_as_future_impl< void >	67
upcxx::detail::bind< FnDecayed >	68
upcxx::detail::bind< bound_function< Fn0, B0... > >	68
upcxx::binding< T >	69
upcxx::binding< T & >	73
upcxx::binding< T && >	74
upcxx::binding< T const >	74
upcxx::binding< T volatile >	75
upcxx::binding< dist_object< T > & >	71
upcxx::binding< dist_object< T > && >	73
upcxx::binding< Fn >	69
upcxx::binding_is_trivial< T, trivial >	75
upcxx::binding_is_trivial< T, std::integral_constant< bool, binding< T >::is_trivial > >	76
upcxx::detail::bound_function_base< Fn, BndTup, all_trivial >	78
upcxx::detail::bound_function_base< Fn, std::tuple< B... > >	78
upcxx::bound_function< Fn, B >	77
upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >	79
upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true >	80
upcxx::commanding< Fn >	82

upcxx::completions< SourceCx, RemoteCx, OpnxCx >	83
upcxx::constant_function< T >	85
upcxx::decay_tupled< Tup >	86
upcxx::decay_tupled< std::tuple< T... > >	86
upcxx::detail::future_header::dependency_link	87
upcxx::digest	88
upcxx::detail::disjoin_cx< A, B, B_ord_bump >	91
upcxx::detail::disjoin_cx< A, nil_cx, B_ord_bump >	92
upcxx::detail::disjoin_cx< nil_cx, B, B_ord_bump >	93
upcxx::detail::disjoin_cx< nil_cx, future_cx< B_ord_old >, B_ord_bump >	94
upcxx::detail::disjoin_cx< nil_cx, nil_cx, B_ord0 >	95
upcxx::dist_id< T >	96
upcxx::dist_object< T >	97
false_type	
upcxx::detail::is_future_cx< Cx >	196
upcxx::fast_hasher	99
upcxx::fast_hashing< T >	101
upcxx::function_ref< Sig >	101
upcxx::function_ref< Ret(Arg...) >	102
upcxx::future1< Kind, T >	104
upcxx::future1< detail::future_kind_shref< detail::future_header_ops_general >, T... >	104
upcxx::future1< T >	104
upcxx::detail::future_body	110
upcxx::detail::future_body_proxy	112
upcxx::detail::future_body_proxy< T >	111
upcxx::detail::future_body_pure< future1< Kind, T... > >	114
upcxx::detail::future_body_then_base	118
upcxx::detail::future_body_then< FuArg, Fn >	116
upcxx::detail::future_body_then_pure< FuArg, Fn >	119
upcxx::detail::future_body_pure< FuArg >	114
upcxx::future_cx< ordinal >	121
upcxx::detail::future_dependency< FuArg >	121
upcxx::detail::future_dependency< Arg >	121
upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >	122
upcxx::detail::future_dependency< future1< future_kind_result > >	124
upcxx::detail::future_dependency< future1< future_kind_result, T... > >	126
upcxx::detail::future_dependency< upcxx::future1< Kind, T... > >	121
upcxx::detail::future_dependency_shref_base< is_trivially_ready_result >	130
upcxx::detail::future_dependency_shref_base< false >	130
upcxx::detail::future_dependency_shref_base< HeaderOps::is_trivially_ready_result >	130
upcxx::detail::future_dependency< future1< future_kind_shref< HeaderOps >, T... > >	128
upcxx::detail::future_dependency_shref_base< true >	132
upcxx::detail::future_dependency_when_all_arg< i, Arg >	134
upcxx::detail::future_dependency_when_all_base< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... > >	135
upcxx::detail::future_dependency_when_all_base< AllArg, lxSeq >	135
upcxx::detail::future_dependency_when_all_base< future1< future_kind_when_all< Arg... >, T... >, upcxx::make_index_sequence< sizeof...(Arg) > >	135
upcxx::detail::future_dependency< future1< future_kind_when_all< Arg... >, T... > >	129
upcxx::detail::future_from_tuple< Kind, Tup >	137
upcxx::detail::future_from_tuple< Kind, std::tuple< T... > >	137
upcxx::detail::future_header	138
upcxx::detail::future_header_dependent	142
upcxx::detail::future_header_result< T >	149
upcxx::detail::future_header_result<>	152
upcxx::detail::future_header_ops_general	144
upcxx::detail::future_header_ops_result	146

upcxx::detail::future_header_ops_result_ready	147
upcxx::detail::future_impl_mapped< FuArg, Fn, T >	155
upcxx::detail::future_impl_result< T >	158
upcxx::detail::future_impl_result<>	160
upcxx::detail::future_impl_shref< HeaderOps, T >	162
upcxx::detail::future_impl_shref< detail::future_header_ops_result, T... >	162
upcxx::promise< T... >	267
upcxx::promise< T >	267
upcxx::detail::future_impl_when_all< ArgTuple, T >	166
upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >	167
upcxx::future_is_trivially_ready< FutureOrKind >	169
upcxx::future_is_trivially_ready< future1< detail::future_kind_mapped< Arg, Fn >, T... > >	170
upcxx::future_is_trivially_ready< future1< detail::future_kind_result, T... > >	171
upcxx::future_is_trivially_ready< future1< detail::future_kind_shref< HeaderOps >, T... > >	171
upcxx::future_is_trivially_ready< future1< detail::future_kind_when_all< Arg... >, T... > >	172
upcxx::detail::future_kind_mapped< FuArg, Fn >	173
upcxx::detail::future_kind_result	174
upcxx::detail::future_kind_shref< HeaderOps >	174
upcxx::detail::future_kind_shref< detail::future_header_ops_general >	174
upcxx::detail::future_kind_when_all< FuArg >	175
upcxx::detail::future_then< Arg, Fn, FnRet, arg_trivial >	176
upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, false >	176
upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, true >	177
upcxx::detail::future_then_pure< Arg, Fn, FnRet, arg_trivial, fnret_trivial >	178
upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, false >	178
upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, true >	179
upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, true, fnret_trivial >	180
upcxx::global_ptr< T >	180
upcxx::detail::global_ptr_ctor_internal	188
std::greater< upcxx::global_ptr< T > >	188
std::greater_equal< upcxx::global_ptr< T > >	189
std::hash< upcxx::digest >	190
std::hash< upcxx::dist_id< T > >	190
std::hash< upcxx::global_ptr< T > >	191
upcxx::hasher_reflector< Hasher >	192
upcxx::index_sequence<... >	194
upcxx::integer_golden_ratio< T >	194
upcxx::integer_golden_ratio_bits< bit_n >	195
upcxx::integer_golden_ratio_bits< 32 >	195
upcxx::integer_golden_ratio_bits< 64 >	196
std::less< upcxx::global_ptr< T > >	197
std::less_equal< upcxx::global_ptr< T > >	198
upcxx::detail::make_future< trivial, T >	199
upcxx::detail::make_future< false, T... >	200
upcxx::detail::make_future< true, T... >	200
upcxx::detail::make_future< upcxx::trait_forall< upcxx::trait_any< std::is_trivially_copyable, std::is_↵ reference >::type, T... >::value, T... >	199
upcxx::detail::make_future< T >	199
upcxx::detail::make_index_sequence< n, s >	201
upcxx::detail::make_index_sequence< 0, s... >	201
mallinfo	202
malloc_chunk	204
malloc_params	206
malloc_segment	207
malloc_state	208
malloc_tree_chunk	212
upcxx::mod2n_hashing< T >	214
upcxx::nil_cx	214

<code>upcxx::nop_function< Sig ></code>	215
<code>upcxx::nop_function< Ret(Arg...)></code>	215
<code>upcxx::nop_function< void(Arg...)></code>	216
<code>upcxx::detail::os_env_parse< T ></code>	217
<code>upcxx::detail::os_env_parse< std::string ></code>	217
<code>upcxx::packing< bound_function< Fn, B... >></code>	218
<code>upcxx::packing< std::array< T, n >></code>	220
<code>upcxx::packing< std::pair< A, B >></code>	221
<code>upcxx::packing< std::string ></code>	222
<code>upcxx::packing< std::tuple< T... >></code>	223
<code>upcxx::packing< std::unordered_map< K, V >></code>	225
<code>upcxx::packing< std::unordered_set< T >></code>	226
<code>upcxx::packing< std::vector< T >></code>	227
<code>upcxx::packing_empty< T, is_default_constructible ></code>	230
<code>upcxx::packing_empty< T ></code>	230
<code>upcxx::packing_opaque< T, true, is_trivially_copyable ></code>	239
<code>upcxx::packing_specializer< T, true, is_funptr, is_scalar ></code>	245
<code>upcxx::packing_empty< T, false ></code>	231
<code>upcxx::packing_empty< T, true ></code>	232
<code>upcxx::packing_function_pointer< T ></code>	234
<code>upcxx::packing_specializer< T, false, true, is_scalar ></code>	245
<code>upcxx::packing_is_trivial< T, false_ ></code>	235
<code>upcxx::packing_is_trivial< T, std::integral_constant< bool, false & packing< T >::is_trivial >></code>	236
<code>upcxx::packing_not_supported< T ></code>	237
<code>upcxx::packing_not_supported< T &&></code>	237
<code>upcxx::packing< T && ></code>	229
<code>upcxx::packing_not_supported< T &></code>	237
<code>upcxx::packing< T & ></code>	228
<code>upcxx::packing_opaque< T, is_empty, is_trivially_copyable ></code>	237
<code>upcxx::packing_pack_reflector</code>	239
<code>upcxx::packing_reflected< T, is_default_constructible ></code>	240
<code>upcxx::packing_reflected< T ></code>	240
<code>upcxx::packing_specializer< T, false, false, false ></code>	244
<code>upcxx::packing_reflected< T, false ></code>	241
<code>upcxx::packing_reflected< T, true ></code>	242
<code>upcxx::packing_specializer< T, is_empty, is_funptr, is_scalar ></code>	243
<code>upcxx::packing_specializer< T ></code>	243
<code>upcxx::packing< T ></code>	218
<code>upcxx::packing< T const ></code>	229
<code>upcxx::packing< T volatile ></code>	230
<code>upcxx::packing_trivial< T ></code>	246
<code>upcxx::packing_opaque< T, false, false ></code>	238
<code>upcxx::packing_opaque< T, false, true ></code>	238
<code>upcxx::packing_specializer< T, false, false, true ></code>	244
<code>upcxx::detail::packing_tuple_each< n, i, T ></code>	247
<code>upcxx::detail::packing_tuple_each< n, n, T... ></code>	249
<code>upcxx::packing_ubound_reflector</code>	251
<code>upcxx::packing_unpack_reflector< member_assignment_not_construction ></code>	252
<code>upcxx::packing_unpack_reflector< false ></code>	252
<code>upcxx::packing_unpack_reflector< true ></code>	254
<code>upcxx::parcel_layout</code>	255
<code>upcxx::parcel_reader</code>	258
<code>upcxx::parcel_writer</code>	261
<code>upcxx::print_proxy< T ></code>	265
<code>upcxx::print_reflector</code>	266
<code>upcxx::promise_cx< T ></code>	267

upcxx::detail::promise_like< Fu >	268
upcxx::detail::promise_like< future1< Kind, T... > >	268
upcxx::reflection< Subject >	269
upcxx::reflection< std::array< T, n > >	270
upcxx::reflection< std::pair< A, B > >	271
upcxx::reflection< std::tuple< Ts... > >	271
upcxx::reflection_tuple< Tup, i, n >	272
upcxx::reflection_tuple< Tup, n, n >	273
upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function	274
upcxx::detail::rget_byval< T >	277
upcxx::detail::rget_futures_of< Mode, R, O >	277
upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > >	278
upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > >	280
upcxx::detail::rget_state_operxn< Mode, Cx >	282
upcxx::detail::rget_state_operxn< Mode, O >	282
upcxx::detail::rget_state_operxn< rget_byref, future_cx< ordinal > >	282
upcxx::detail::rget_state_operxn< rget_byref, nil_cx >	284
upcxx::detail::rget_state_operxn< rget_byref, promise_cx<> >	285
upcxx::detail::rget_state_operxn< rget_byval< T >, future_cx< ordinal > >	286
upcxx::detail::rget_state_operxn< rget_byval< T >, nil_cx >	287
upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T > >	288
upcxx::detail::rget_state_remote< Cx >	290
upcxx::detail::rget_state_remote< nil_cx >	290
upcxx::detail::rget_state_remote< R >	290
upcxx::detail::rget_state_remote< rpc_cx< Fn > >	291
upcxx::detail::rget_states< Mode, R, O >	292
upcxx::backend::gasnet1_seq::rma_cb	294
upcxx::backend::rma_get_cb	295
upcxx::backend::rma_get_cb_wstate< State >	297
upcxx::backend::gasnet1_seq::rma_get_cb_impl< State, OpCx >	296
upcxx::backend::rma_put_cb	298
upcxx::backend::rma_put_cb_wstate< State >	301
upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx >	299
upcxx::rpc_cx< Fn >	302
upcxx::detail::rpc_recipient_after< Pro >	303
upcxx::detail::rpc_return< ValidType, Fn, Args >	304
upcxx::detail::rput_byval< T >	305
upcxx::detail::rput_futures_of< S, R, O >	305
upcxx::detail::rput_futures_of< future_cx< S_ord >, R, future_cx< O_ord > >	307
upcxx::detail::rput_futures_of< future_cx< S_ord >, R, O >	309
upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > >	310
upcxx::detail::rput_state_here< Cx >	312
upcxx::detail::rput_state_here< future_cx< ordinal > >	312
upcxx::detail::rput_state_here< nil_cx >	314
upcxx::detail::rput_state_here< O >	312
upcxx::detail::rput_state_here< promise_cx<> >	315
upcxx::detail::rput_state_here< S >	312
upcxx::detail::rput_state_remote< Cx >	316
upcxx::detail::rput_state_remote< nil_cx >	316
upcxx::detail::rput_state_remote< R >	316
upcxx::detail::rput_state_remote< rpc_cx< Fn > >	317
upcxx::detail::rput_states< Mode, S, R, O >	319
upcxx::detail::rput_states< rput_byref, S, R, O >	319
upcxx::detail::rput_states< rput_byval< T >, S, R, O >	321
upcxx::trait_all< Tr >	322
upcxx::trait_all< Tr0, Trs... >	322
upcxx::trait_all<>	323
upcxx::trait_any< Tr >	324

upcxx::trait_any< Tr0, Trs... >	324
upcxx::trait_any<>	324
upcxx::trait_forall< Test, T >	325
upcxx::trait_forall< Test >	325
upcxx::trait_forall< Test, T, Ts... >	326
upcxx::trait_forall_tupled< Test, Tuple >	327
upcxx::trait_forall_tupled< Test, std::tuple< T... > >	327
true_type	
upcxx::detail::is_future_cx< future_cx< ordinal > >	197
upcxx::detail::tuple_get_or_void< i, TupRef, in_range >	328
upcxx::detail::tuple_get_or_void< i, TupRef, false >	329
upcxx::detail::tuple_rvals_get< i, Tup >	329
upcxx::detail::tuple_rvals_get< Tup &&, i, Ti & >	330
upcxx::detail::tuple_rvals_get< Tup &&, i, Ti && >	330
upcxx::detail::tuple_rvals_get< Tup &&, i, Ti >	331
upcxx::detail::tuple_rvals_get< Tup &, i, Ti & >	332
upcxx::detail::tuple_rvals_get< Tup &, i, Ti && >	332
upcxx::detail::tuple_rvals_get< Tup &, i, Ti >	333
upcxx::detail::tuple_rvals_get< Tup const &, i, Ti & >	334
upcxx::detail::tuple_rvals_get< Tup const &, i, Ti && >	334
upcxx::detail::tuple_rvals_get< Tup const &, i, Ti >	335
upcxx::tuple_types_into< Tuple, Into >	336
upcxx::tuple_types_into< std::tuple< T... >, Into >	336
upcxx::trait_all< Tr0, Trs... >::type< T >	337
upcxx::trait_any< Tr0, Trs... >::type< T >	337

Chapter 3

Class Index

3.1 Class List

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

upcxx::backend::gasnet1_seq::action	53
upcxx::backend::gasnet1_seq::action_impl< Fn >	54
upcxx::add_lref_if_nonref< T >	55
upcxx::add_lref_if_nonref< T & >	56
upcxx::add_lref_if_nonref< T && >	57
upcxx::detail::allreduce_state< T, Op >	57
upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator< Arg >	59
upcxx::detail::apply_futured_as_future< Fn(future1< Kind, T... >)>	61
upcxx::detail::apply_tupled_as_future< Fn, ArgTup >	63
upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, ArgTupDecayed >	63
upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, std::tuple< T... > >	64
upcxx::detail::apply_tupled_as_future_impl< Return >	64
upcxx::detail::apply_tupled_as_future_impl< future1< Kind, T... > >	65
upcxx::detail::apply_tupled_as_future_impl< void >	67
upcxx::detail::bind< FnDecayed >	68
upcxx::detail::bind< bound_function< Fn0, B0... > >	68
upcxx::binding< T >	69
upcxx::binding< dist_object< T > & >	71
upcxx::binding< dist_object< T > && >	73
upcxx::binding< T & >	73
upcxx::binding< T && >	74
upcxx::binding< T const >	74
upcxx::binding< T volatile >	75
upcxx::binding_is_trivial< T, trivial >	75
upcxx::binding_is_trivial< T, std::integral_constant< bool, binding< T >::is_trivial > >	76
upcxx::bound_function< Fn, B >	77
upcxx::detail::bound_function_base< Fn, BndTup, all_trivial >	78
upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >	79
upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true >	80
upcxx::commanding< Fn >	82
upcxx::completions< SourceCx, RemoteCx, OpxnCx >	83
upcxx::constant_function< T >	85
upcxx::decay_tupled< Tup >	86
upcxx::decay_tupled< std::tuple< T... > >	86
upcxx::detail::future_header::dependency_link	87

upcxx::digest	88
upcxx::detail::disjoin_cx< A, B, B_ord_bump >	91
upcxx::detail::disjoin_cx< A, nil_cx, B_ord_bump >	92
upcxx::detail::disjoin_cx< nil_cx, B, B_ord_bump >	93
upcxx::detail::disjoin_cx< nil_cx, future_cx< B_ord_old >, B_ord_bump >	94
upcxx::detail::disjoin_cx< nil_cx, nil_cx, B_ord0 >	95
upcxx::dist_id< T >	96
upcxx::dist_object< T >	97
upcxx::fast_hasher	99
upcxx::fast_hashing< T >	101
upcxx::function_ref< Sig >	101
upcxx::function_ref< Ret(Arg...)>	102
upcxx::future1< Kind, T >	104
upcxx::detail::future_body	110
upcxx::detail::future_body_proxy< T >	111
upcxx::detail::future_body_proxy_	112
upcxx::detail::future_body_pure< FuArg >	114
upcxx::detail::future_body_pure< future1< Kind, T... > >	114
upcxx::detail::future_body_then< FuArg, Fn >	116
upcxx::detail::future_body_then_base	118
upcxx::detail::future_body_then_pure< FuArg, Fn >	119
upcxx::future_cx< ordinal >	121
upcxx::detail::future_dependency< FuArg >	121
upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >	122
upcxx::detail::future_dependency< future1< future_kind_result > >	124
upcxx::detail::future_dependency< future1< future_kind_result, T... > >	126
upcxx::detail::future_dependency< future1< future_kind_shref< HeaderOps >, T... > >	128
upcxx::detail::future_dependency< future1< future_kind_when_all< Arg... >, T... > >	129
upcxx::detail::future_dependency_shref_base< is_trivially_ready_result >	130
upcxx::detail::future_dependency_shref_base< false >	130
upcxx::detail::future_dependency_shref_base< true >	132
upcxx::detail::future_dependency_when_all_arg< i, Arg >	134
upcxx::detail::future_dependency_when_all_base< AllArg, lxSeq >	135
upcxx::detail::future_dependency_when_all_base< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... > >	135
upcxx::detail::future_from_tuple< Kind, Tup >	137
upcxx::detail::future_from_tuple< Kind, std::tuple< T... > >	137
upcxx::detail::future_header	138
upcxx::detail::future_header_dependent	142
upcxx::detail::future_header_ops_general	144
upcxx::detail::future_header_ops_result	146
upcxx::detail::future_header_ops_result_ready	147
upcxx::detail::future_header_result< T >	149
upcxx::detail::future_header_result<>	152
upcxx::detail::future_impl_mapped< FuArg, Fn, T >	155
upcxx::detail::future_impl_result< T >	158
upcxx::detail::future_impl_result<>	160
upcxx::detail::future_impl_shref< HeaderOps, T >	162
upcxx::detail::future_impl_when_all< ArgTuple, T >	166
upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >	167
upcxx::future_is_trivially_ready< FutureOrKind >	169
upcxx::future_is_trivially_ready< future1< detail::future_kind_mapped< Arg, Fn >, T... > >	170
upcxx::future_is_trivially_ready< future1< detail::future_kind_result, T... > >	171
upcxx::future_is_trivially_ready< future1< detail::future_kind_shref< HeaderOps >, T... > >	171
upcxx::future_is_trivially_ready< future1< detail::future_kind_when_all< Arg... >, T... > >	172
upcxx::detail::future_kind_mapped< FuArg, Fn >	173
upcxx::detail::future_kind_result	174
upcxx::detail::future_kind_shref< HeaderOps >	174

upcxx::detail::future_kind_when_all< FuArg >	175
upcxx::detail::future_then< Arg, Fn, FnRet, arg_trivial >	176
upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, false >	176
upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, true >	177
upcxx::detail::future_then_pure< Arg, Fn, FnRet, arg_trivial, fnret_trivial >	178
upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, false >	178
upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, true >	179
upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, true, fnret_trivial >	180
upcxx::global_ptr< T >	180
upcxx::detail::global_ptr_ctor_internal	188
std::greater< upcxx::global_ptr< T > >	188
std::greater_equal< upcxx::global_ptr< T > >	189
std::hash< upcxx::digest >	190
std::hash< upcxx::dist_id< T > >	190
std::hash< upcxx::global_ptr< T > >	191
upcxx::hasher_reflector< Hasher >	192
upcxx::index_sequence<... >	194
upcxx::integer_golden_ratio< T >	194
upcxx::integer_golden_ratio_bits< bit_n >	195
upcxx::integer_golden_ratio_bits< 32 >	195
upcxx::integer_golden_ratio_bits< 64 >	196
upcxx::detail::is_future_cx< Cx >	196
upcxx::detail::is_future_cx< future_cx< ordinal > >	197
std::less< upcxx::global_ptr< T > >	197
std::less_equal< upcxx::global_ptr< T > >	198
upcxx::detail::make_future< T >	199
upcxx::detail::make_future_< trivial, T >	199
upcxx::detail::make_future_< false, T... >	200
upcxx::detail::make_future_< true, T... >	200
upcxx::detail::make_index_sequence< n, s >	201
upcxx::detail::make_index_sequence< 0, s... >	201
mallinfo	202
malloc_chunk	204
malloc_params	206
malloc_segment	207
malloc_state	208
malloc_tree_chunk	212
upcxx::mod2n_hashing< T >	214
upcxx::nil_cx	214
upcxx::nop_function< Sig >	215
upcxx::nop_function< Ret(Arg...)>	215
upcxx::nop_function< void(Arg...)>	216
upcxx::detail::os_env_parse< T >	217
upcxx::detail::os_env_parse< std::string >	217
upcxx::packing< T >	218
upcxx::packing< bound_function< Fn, B... > >	218
upcxx::packing< std::array< T, n > >	220
upcxx::packing< std::pair< A, B > >	221
upcxx::packing< std::string >	222
upcxx::packing< std::tuple< T... > >	223
upcxx::packing< std::unordered_map< K, V > >	225
upcxx::packing< std::unordered_set< T > >	226
upcxx::packing< std::vector< T > >	227
upcxx::packing< T & >	228
upcxx::packing< T && >	229
upcxx::packing< T const >	229
upcxx::packing< T volatile >	230
upcxx::packing_empty< T, is_default_constructible >	230

upcxx::packing_empty< T, false >	231
upcxx::packing_empty< T, true >	232
upcxx::packing_function_pointer< T >	234
upcxx::packing_is_trivial< T, false_ >	235
upcxx::packing_is_trivial< T, std::integral_constant< bool, false &packing< T >::is_trivial > >	236
upcxx::packing_not_supported< T >	237
upcxx::packing_opaque< T, is_empty, is_trivially_copyable >	237
upcxx::packing_opaque< T, false, false >	238
upcxx::packing_opaque< T, false, true >	238
upcxx::packing_opaque< T, true, is_trivially_copyable >	239
upcxx::packing_pack_reflector	239
upcxx::packing_reflected< T, is_default_constructible >	240
upcxx::packing_reflected< T, false >	241
upcxx::packing_reflected< T, true >	242
upcxx::packing_specializer< T, is_empty, is_funptr, is_scalar >	243
upcxx::packing_specializer< T, false, false, false >	244
upcxx::packing_specializer< T, false, false, true >	244
upcxx::packing_specializer< T, false, true, is_scalar >	245
upcxx::packing_specializer< T, true, is_funptr, is_scalar >	245
upcxx::packing_trivial< T >	246
upcxx::detail::packing_tuple_each< n, i, T >	247
upcxx::detail::packing_tuple_each< n, n, T... >	249
upcxx::packing_ubound_reflector	251
upcxx::packing_unpack_reflector< member_assignment_not_construction >	252
upcxx::packing_unpack_reflector< false >	252
upcxx::packing_unpack_reflector< true >	254
upcxx::parcel_layout	255
upcxx::parcel_reader	258
upcxx::parcel_writer	261
upcxx::print_proxy< T >	265
upcxx::print_reflector	266
upcxx::promise< T >	267
upcxx::promise_cx< T >	267
upcxx::detail::promise_like< Fu >	268
upcxx::detail::promise_like< future1< Kind, T... > >	268
upcxx::reflection< Subject >	269
upcxx::reflection< std::array< T, n > >	270
upcxx::reflection< std::pair< A, B > >	271
upcxx::reflection< std::tuple< Ts... > >	271
upcxx::reflection_tuple< Tup, i, n >	272
upcxx::reflection_tuple< Tup, n, n >	273
upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function	274
upcxx::detail::rget_byval< T >	277
upcxx::detail::rget_futures_of< Mode, R, O >	277
upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > >	278
upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > >	280
upcxx::detail::rget_state_operxn< Mode, Cx >	282
upcxx::detail::rget_state_operxn< rget_byref, future_cx< ordinal > >	282
upcxx::detail::rget_state_operxn< rget_byref, nil_cx >	284
upcxx::detail::rget_state_operxn< rget_byref, promise_cx<> >	285
upcxx::detail::rget_state_operxn< rget_byval< T >, future_cx< ordinal > >	286
upcxx::detail::rget_state_operxn< rget_byval< T >, nil_cx >	287
upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T > >	288
upcxx::detail::rget_state_remote< Cx >	290
upcxx::detail::rget_state_remote< nil_cx >	290
upcxx::detail::rget_state_remote< rpc_cx< Fn > >	291
upcxx::detail::rget_states< Mode, R, O >	292
upcxx::backend::gasnet1_seq::rma_cb	294

upcxx::backend::rma_get_cb	295
upcxx::backend::gasnet1_seq::rma_get_cb_impl< State, OpCx >	296
upcxx::backend::rma_get_cb_wstate< State >	297
upcxx::backend::rma_put_cb	298
upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx >	299
upcxx::backend::rma_put_cb_wstate< State >	301
upcxx::rpc_cx< Fn >	302
upcxx::detail::rpc_recipient_after< Pro >	303
upcxx::detail::rpc_return< ValidType, Fn, Args >	304
upcxx::detail::rput_byval< T >	305
upcxx::detail::rput_futures_of< S, R, O >	305
upcxx::detail::rput_futures_of< future_cx< S_ord >, R, future_cx< O_ord > >	307
upcxx::detail::rput_futures_of< future_cx< S_ord >, R, O >	309
upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > >	310
upcxx::detail::rput_state_here< Cx >	312
upcxx::detail::rput_state_here< future_cx< ordinal > >	312
upcxx::detail::rput_state_here< nil_cx >	314
upcxx::detail::rput_state_here< promise_cx<> >	315
upcxx::detail::rput_state_remote< Cx >	316
upcxx::detail::rput_state_remote< nil_cx >	316
upcxx::detail::rput_state_remote< rpc_cx< Fn > >	317
upcxx::detail::rput_states< Mode, S, R, O >	319
upcxx::detail::rput_states< rput_byref, S, R, O >	319
upcxx::detail::rput_states< rput_byval< T >, S, R, O >	321
upcxx::trait_all< Tr >	322
upcxx::trait_all< Tr0, Trs... >	322
upcxx::trait_all<>	323
upcxx::trait_any< Tr >	324
upcxx::trait_any< Tr0, Trs... >	324
upcxx::trait_any<>	324
upcxx::trait_forall< Test, T >	325
upcxx::trait_forall< Test >	325
upcxx::trait_forall< Test, T, Ts... >	326
upcxx::trait_forall_tupled< Test, Tuple >	327
upcxx::trait_forall_tupled< Test, std::tuple< T... > >	327
upcxx::detail::tuple_get_or_void< i, TupRef, in_range >	328
upcxx::detail::tuple_get_or_void< i, TupRef, false >	329
upcxx::detail::tuple_rvals_get< i, Tup >	329
upcxx::detail::tuple_rvals_get< Tup &&, i, Ti & >	330
upcxx::detail::tuple_rvals_get< Tup &&, i, Ti && >	330
upcxx::detail::tuple_rvals_get< Tup &&, i, Ti >	331
upcxx::detail::tuple_rvals_get< Tup &, i, Ti & >	332
upcxx::detail::tuple_rvals_get< Tup &, i, Ti && >	332
upcxx::detail::tuple_rvals_get< Tup &, i, Ti >	333
upcxx::detail::tuple_rvals_get< Tup const &, i, Ti & >	334
upcxx::detail::tuple_rvals_get< Tup const &, i, Ti && >	334
upcxx::detail::tuple_rvals_get< Tup const &, i, Ti >	335
upcxx::tuple_types_into< Tuple, Into >	336
upcxx::tuple_types_into< std::tuple< T... >, Into >	336
upcxx::trait_all< Tr0, Trs... >::type< T >	337
upcxx::trait_any< Tr0, Trs... >::type< T >	337

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

allocate.hpp	339
allreduce.hpp	340
atomic.hpp	340
backend.hpp	342
bind.hpp	344
broadcast.hpp	345
command.hpp	345
completion.hpp	346
diagnostic.cpp	347
diagnostic.hpp	347
digest.cpp	349
digest.hpp	350
dist_object.cpp	350
dist_object.hpp	350
dl_malloc.c	351
dl_malloc.h	401
future.hpp	408
global_ptr.hpp	415
nobsrule.py	415
os_env.hpp	416
packing.cpp	416
packing.hpp	416
reflection.hpp	418
rget.hpp	419
rpc.hpp	420
rput.hpp	420
upcxx.cpp	421
upcxx.hpp	421
utility.hpp	421
wait.hpp	423
backend/gasnet1_seq/backend.cpp	341
backend/gasnet1_seq/backend.hpp	341
future/apply.hpp	408
future/body_pure.hpp	409
future/core.cpp	409

future/core.hpp	409
future/future1.hpp	410
future/impl_mapped.hpp	411
future/impl_result.hpp	411
future/impl_shref.hpp	412
future/impl_when_all.hpp	412
future/make_future.hpp	412
future/promise.hpp	413
future/then.hpp	414
future/when_all.hpp	414

Chapter 5

Namespace Documentation

5.1 nobsrule Namespace Reference

Functions

- def [requires_upcxx_backend](#) (cxt, src)

5.1.1 Detailed Description

This is a nobs rule-file. See nobs/nobs/ruletree.py for documentation on the structure and interpretation of a rule-file.

5.1.2 Function Documentation

5.1.2.1 [requires_upcxx_backend\(\)](#)

```
def nobsrule.requires_upcxx_backend (
    cxt,
    src )
```

Definition at line 7 of file nobsrule.py.

5.2 std Namespace Reference

Classes

- struct [greater](#)< upcxx::global_ptr< T > >
- struct [greater_equal](#)< upcxx::global_ptr< T > >
- struct [hash](#)< upcxx::digest >
- struct [hash](#)< upcxx::dist_id< T > >
- struct [hash](#)< upcxx::global_ptr< T > >
- struct [less](#)< upcxx::global_ptr< T > >
- struct [less_equal](#)< upcxx::global_ptr< T > >

5.3 upcxx Namespace Reference

Namespaces

- [backend](#)
- [detail](#)

Classes

- struct [add_lref_if_nonref](#)
- struct [add_lref_if_nonref< T & >](#)
- struct [add_lref_if_nonref< T && >](#)
- struct [binding](#)
- struct [binding< dist_object< T > & >](#)
- struct [binding< dist_object< T > && >](#)
- struct [binding< T & >](#)
- struct [binding< T && >](#)
- struct [binding< T const >](#)
- struct [binding< T volatile >](#)
- struct [binding_is_trivial](#)
- struct [binding_is_trivial< T, std::integral_constant< bool, binding< T >::is_trivial > >](#)
- struct [bound_function](#)
- struct [commanding](#)
- struct [completions](#)
- struct [constant_function](#)
- struct [decay_tupled](#)
- struct [decay_tupled< std::tuple< T... > >](#)
- struct [digest](#)
- struct [dist_id](#)
- class [dist_object](#)
- struct [fast_hasher](#)
- struct [fast_hashing](#)
- class [function_ref](#)
- class [function_ref< Ret\(Arg...\)>](#)
- struct [future1](#)
- struct [future_cx](#)
- struct [future_is_trivially_ready](#)
- struct [future_is_trivially_ready< future1< detail::future_kind_mapped< Arg, Fn >, T... > >](#)
- struct [future_is_trivially_ready< future1< detail::future_kind_result, T... > >](#)
- struct [future_is_trivially_ready< future1< detail::future_kind_shref< HeaderOps >, T... > >](#)
- struct [future_is_trivially_ready< future1< detail::future_kind_when_all< Arg... >, T... > >](#)
- class [global_ptr](#)
- struct [hasher_reflector](#)
- struct [index_sequence](#)
- struct [integer_golden_ratio](#)
- struct [integer_golden_ratio_bits](#)
- struct [integer_golden_ratio_bits< 32 >](#)
- struct [integer_golden_ratio_bits< 64 >](#)
- struct [mod2n_hashing](#)
- struct [nil_cx](#)
- struct [nop_function](#)
- struct [nop_function< Ret\(Arg...\)>](#)
- struct [nop_function< void\(Arg...\)>](#)

- struct [packing](#)
- struct [packing< bound_function< Fn, B... > >](#)
- struct [packing< std::array< T, n > >](#)
- struct [packing< std::pair< A, B > >](#)
- struct [packing< std::string >](#)
- struct [packing< std::tuple< T... > >](#)
- struct [packing< std::unordered_map< K, V > >](#)
- struct [packing< std::unordered_set< T > >](#)
- struct [packing< std::vector< T > >](#)
- struct [packing< T & >](#)
- struct [packing< T && >](#)
- struct [packing< T const >](#)
- struct [packing< T volatile >](#)
- struct [packing_empty](#)
- struct [packing_empty< T, false >](#)
- struct [packing_empty< T, true >](#)
- struct [packing_function_pointer](#)
- struct [packing_is_trivial](#)
- struct [packing_is_trivial< T, std::integral_constant< bool, false &packing< T >::is_trivial > >](#)
- struct [packing_not_supported](#)
- struct [packing_opaque](#)
- struct [packing_opaque< T, false, false >](#)
- struct [packing_opaque< T, false, true >](#)
- struct [packing_opaque< T, true, is_trivially_copyable >](#)
- struct [packing_pack_reflector](#)
- struct [packing_reflected](#)
- struct [packing_reflected< T, false >](#)
- struct [packing_reflected< T, true >](#)
- struct [packing_specializer](#)
- struct [packing_specializer< T, false, false, false >](#)
- struct [packing_specializer< T, false, false, true >](#)
- struct [packing_specializer< T, false, true, is_scalar >](#)
- struct [packing_specializer< T, true, is_funptr, is_scalar >](#)
- struct [packing_trivial](#)
- struct [packing_ubound_reflector](#)
- struct [packing_unpack_reflector](#)
- struct [packing_unpack_reflector< false >](#)
- struct [packing_unpack_reflector< true >](#)
- class [parcel_layout](#)
- class [parcel_reader](#)
- struct [parcel_writer](#)
- struct [print_proxy](#)
- struct [print_reflector](#)
- class [promise](#)
- struct [promise_cx](#)
- struct [reflection](#)
- struct [reflection< std::array< T, n > >](#)
- struct [reflection< std::pair< A, B > >](#)
- struct [reflection< std::tuple< Ts... > >](#)
- struct [reflection_tuple](#)
- struct [reflection_tuple< Tup, n, n >](#)
- struct [rpc_cx](#)
- struct [trait_all](#)
- struct [trait_all< Tr0, Trs... >](#)
- struct [trait_all<>](#)

- struct [trait_any](#)
- struct [trait_any](#)< Tr0, Trs... >
- struct [trait_any](#)<>
- struct [trait_forall](#)
- struct [trait_forall](#)< Test >
- struct [trait_forall](#)< Test, T, Ts... >
- struct [trait_forall_tupled](#)
- struct [trait_forall_tupled](#)< Test, std::tuple< T... > >
- struct [tuple_types_into](#)
- struct [tuple_types_into](#)< std::tuple< T... >, Into >

Typedefs

- typedef int [inrank_t](#)
- typedef unsigned int [uinrank_t](#)
- template<typename ... T>
using [future](#) = [future1](#)< [detail::future_kind_shref](#)< [detail::future_header_ops_general](#) >, T... >
- template<typename Tuple , template< typename... > class Into>
using [tuple_types_into_t](#) = typename [tuple_types_into](#)< Tuple, Into >::type
- template<int n>
using [make_index_sequence](#) = typename [detail::make_index_sequence](#)< n >::type

Enumerations

- enum [progress_level](#) { [progress_level::internal](#), [progress_level::user](#) }

Functions

- template<typename T , std::size_t alignment = alignof(T)>
[global_ptr](#)< T > [allocate](#) (std::size_t n=1)
- template<typename T >
void [deallocate](#) ([global_ptr](#)< T > gptr)
- template<typename T , typename ... Args>
[global_ptr](#)< T > [new_](#) (Args &&...args)
- template<typename T , typename ... Args>
[global_ptr](#)< T > [new_](#) (const std::nothrow_t &tag, Args &&...args)
- template<typename T >
[global_ptr](#)< T > [new_array](#) (std::size_t n)
- template<typename T >
[global_ptr](#)< T > [new_array](#) (std::size_t n, const std::nothrow_t &tag)
- template<typename T >
void [delete_](#) ([global_ptr](#)< T > gptr)
- template<typename T >
void [delete_array](#) ([global_ptr](#)< T > gptr)
- template<typename T1 , typename BinaryOp , typename T = typename std::decay<T1>::type>
[future](#)< T > [allreduce](#) (T1 &&value, BinaryOp op)
- template<typename T >
[future](#)< T > [atomic_get](#) ([global_ptr](#)< T > p, std::memory_order order)
- template<typename T >
[future](#) [atomic_put](#) ([global_ptr](#)< T > p, T val, std::memory_order order)
- template<typename T >
[future](#)< T > [atomic_fetch_add](#) ([global_ptr](#)< T > p, T val, std::memory_order order)
- void [init](#) ()

- void `finalize` ()
- `inrank_t rank_n` ()
- `inrank_t rank_me` ()
- void * `allocate` (std::size_t size, std::size_t alignment=alignof(std::max_align_t))
- void `deallocate` (void *p)
- void `progress` (progress_level lev=progress_level::user)
- void `barrier` ()
- template<typename Fn , typename ... B>
auto `bind` (Fn &&fn, B &&...b) -> decltype(detail::bind< typename std::decay< Fn >::type >()(std::forward< Fn >(fn), std::forward< B >(b)...))
- template<typename Fn >
Fn && `bind` (Fn &&fn)
- template<typename ... P>
auto `bind_last` (P &&...parm) -> decltype(detail::bind_last(std::tuple< P &&... >
- template<typename Fn >
Fn && `bind_last` (Fn &&fn)
- template<typename T1 , typename T = typename std::decay<T1>::type>
`future`< T > `broadcast` (T1 &&value, `inrank_t` root)
- template<typename Fn >
void `command_size_ubound` (parcel_layout &ub, Fn &&fn)
- template<typename Fn >
void `command_pack` (parcel_writer &w, std::size_t size_ub, Fn &&fn)
- `future command_execute` (parcel_reader &r)
- template<typename Fn1 >
void `command_size_ubound` (parcel_layout &ub, Fn1 &&fn)
- template<typename Fn1 >
void `command_pack` (parcel_writer &w, std::size_t size_ub, Fn1 &&fn)
- template<typename AS , typename AR , typename AO , typename BS , typename BR , typename BO >
`completions`< typename detail::disjoin_cx< AS, BS, completions< AS, AR, AO >::future_n >::type, type-name detail::disjoin_cx< AR, BR, completions< AS, AR, AO >::future_n >::type, typename detail::disjoin_cx< AO, BO, completions< AS, AR, AO >::future_n >::type > operator| (completions< AS, AR, AO > a, completions< BS, BR, BO > b)
- template<typename ... T>
`completions`< promise_cx< T... >, nil_cx, nil_cx > `source_cx_as_promise` (promise< T... > &pro)
- template<typename ... T>
`completions`< nil_cx, nil_cx, promise_cx< T... > > `operxn_cx_as_promise` (promise< T... > &pro)
- template<typename Fn , typename ... Args>
auto `remote_cx_as_rpc` (Fn &&fn, Args &&...args) -> completions< nil_cx, rpc_cx< decltype(upcxx::bind(std::forward< Fn >(fn), std::forward< Args >(args)...))>, nil_cx >
- void `dbgbrk` ()
- void `assert_failed` (const char *file, int line, const char *msg=nullptr)
- std::ostream & `operator<<` (std::ostream &o, digest x)
- template<typename T >
std::ostream & `operator<<` (std::ostream &o, dist_id< T > x)
- template<typename Fn , typename ArgTup >
auto `apply_tupled_as_future` (Fn &&fn, ArgTup &&argtup) -> decltype(detail::apply_tupled_as_future< Fn, ArgTup >()(std::forward< Fn >(fn), std::forward< ArgTup >(argtup)))
- template<typename ... T>
auto `make_future` (T ...values) -> decltype(detail::make_future< T... >()(std::declval< T >()...))
- template<typename T >
auto `to_future` (T x) -> decltype(make_future(std::forward< T >(x)))
- template<typename Kind , typename ... T>
`future1`< Kind, T... > `to_future` (future1< Kind, T... > x)
- template<typename ArgKind , typename Fn1 , typename ... ArgT>
auto `operator>>` (future1< ArgKind, ArgT... > arg, Fn1 &&fn) -> decltype(detail::future_then< future1< ArgKind, ArgT... >, typename std::decay< Fn1 >::type >()(std::move(arg), std::forward< Fn1 >(fn)))

- `template<typename Fn1 , typename ... ArgT>`
`future< ArgT... > & operator>>= (future< ArgT... > &arg, Fn1 &&fn)`
- `template<typename ... Arg>`
`detail::when_all_return_t< Arg... > when_all (Arg ...args)`
- `bool is_memory_shared_with (inrank_t r)`
- `void * pshm_remote_addr2local (inrank_t r, void *addr)`
- `void * pshm_local_addr2remote (void *addr, inrank_t &rank_out)`
- `template<typename T >`
`global_ptr< T > operator+ (std::ptrdiff_t diff, global_ptr< T > ptr)`
- `template<typename T , typename U >`
`global_ptr< T > reinterpret_pointer_cast (global_ptr< U > ptr)`
- `template<typename T >`
`std::ostream & operator<< (std::ostream &os, global_ptr< T > ptr)`
- `template<class T >`
`T os_env (const std::string &name)`
- `template<class T >`
`T os_env (const std::string &name, const T &otherwise)`
- `template<typename Reflector , typename Subject >`
`void reflect_upon (Reflector &re, Subject &&subj)`
- `template<typename T >`
`std::size_t fast_hash (const T &x, std::size_t salt=0)`
- `template<typename T >`
`std::size_t mod2n_hash (const T &x, int bits)`
- `template<typename T >`
`print_proxy< T > print (T const &subject)`
- `template<typename T , typename R = nil_cx, typename O = future_cx<0>>`
`detail::rget_futures_of< detail::rget_byval< T >, R, O >::return_type rget (global_ptr< T > gp_s, comple-
 tions< nil_cx, R, O > cxs=completions< nil_cx, R, O >{ })`
- `template<typename T , typename R = nil_cx, typename O = future_cx<0>>`
`detail::rget_futures_of< detail::rget_byref, R, O >::return_type rget (global_ptr< T > gp_s, T *buf_d, std::←
 ::size_t n, completions< nil_cx, R, O > cxs=completions< nil_cx, R, O >{ })`
- `template<typename Fn , typename ... Args>`
`void rpc_ff (inrank_t recipient, Fn &&fn, Args &&...args)`
- `template<typename Fn , typename ... Args>`
`auto rpc (inrank_t recipient, Fn &&fn, Args &&...args) -> typename detail::rpc_return< decltype(fn(args...)),
 Fn, Args... >::type`
- `template<typename Fn , typename ... T, typename ... Args>`
`void rpc (inrank_t recipient, promise< T... > &prom, Fn &&fn, Args &&...args)`
- `template<typename T , typename S = nil_cx, typename R = nil_cx, typename O = future_cx<0>>`
`detail::rput_futures_of< S, R, O >::return_type rput (T value_s, global_ptr< T > gp_d, completions< S, R,
 O > cxs=completions< S, R, O >{ })`
- `template<typename T , typename S = nil_cx, typename R = nil_cx, typename O = future_cx<0>>`
`detail::rput_futures_of< S, R, O >::return_type rput (T const *buf_s, std::size_t n, global_ptr< T > gp_d,
 completions< S, R, O > cxs=completions< S, R, O >{ })`
- `template<typename Sig >`
`nop_function< Sig > nop ()`
- `template<typename T >`
`constant_function< T > constant (T value)`
- `template<int i, typename Tup >`
`auto get_or_void (Tup &&tup) -> decltype(detail::tuple_get_or_void< i, Tup >()(std::forward< Tup >(tup)))`
- `template<typename Tup >`
`auto tuple_rvals (Tup &&tup) -> decltype(detail::tuple_rvals(std::forward< Tup >(tup), make_index_←
 sequence< std::tuple_size< typename std::decay< Tup >::type >::value >()))`
- `template<typename Fn , typename Tup >`
`auto apply_tupled (Fn &&fn, Tup &&args) -> decltype(detail::apply_tupled(std::forward< Fn >(fn), std::←
 ::forward< Tup >(args), make_index_sequence< std::tuple_size< Tup >::value >()))`
- `template<typename Kind , typename ... T>`
`auto wait (future1< Kind, T... > const &f) -> decltype(f.result())`

Variables

- constexpr `completions< future_cx< 0 >, nil_cx, nil_cx > source_cx_as_future = {}`
- constexpr `completions< nil_cx, nil_cx, future_cx< 0 > > operxn_cx_as_future = {}`
- bool `dbgbrk_spin_init = false`
- bool `dbgbrk_spin = true`

5.3.1 Detailed Description

[allocate.hpp](#)

[atomic.hpp](#)

[global_ptr.hpp](#)

5.3.2 Typedef Documentation

5.3.2.1 future

```
template<typename ... T>
using upcxx::future = typedef future1< detail::future_kind_shref<detail::future_header_ops_↵
general>, T... >
```

Definition at line 83 of file core.hpp.

5.3.2.2 intrank_t

```
typedef int upcxx::inrank_t
```

Definition at line 18 of file backend.hpp.

5.3.2.3 make_index_sequence

```
template<int n>
upcxx::make_index_sequence< sizeof...(P) -1 >
```

Definition at line 203 of file utility.hpp.

5.3.2.4 tuple_types_into_t

```
template<typename Tuple , template< typename... > class Into>
using upcxx::tuple_types_into_t = typedef typename tuple_types_into<Tuple,Into>::type
```

Definition at line 183 of file utility.hpp.

5.3.2.5 uinrank_t

```
typedef unsigned int upcxx::uinrank_t
```

Definition at line 19 of file backend.hpp.

5.3.3 Enumeration Type Documentation

5.3.3.1 progress_level

```
enum upcxx::progress_level [strong]
```

Enumerator

internal	
user	

Definition at line 21 of file backend.hpp.

5.3.4 Function Documentation

5.3.4.1 allocate() [1/2]

```
void* upcxx::allocate (
    std::size_t size,
    std::size_t alignment = alignof(std::max_align_t) )
```

5.3.4.2 allocate() [2/2]

```
template<typename T , std::size_t alignment = alignof(T)>
global_ptr<T> upcxx::allocate (
    std::size_t n = 1 )
```

Definition at line 33 of file allocate.hpp.

5.3.4.3 allreduce()

```
template<typename T1 , typename BinaryOp , typename T = typename std::decay<T1>::type>
future<T> upcxx::allreduce (
    T1 && value,
    BinaryOp op )
```

Definition at line 31 of file allreduce.hpp.

5.3.4.4 apply_tupled()

```
template<typename Fn , typename Tup >
auto upcxx::apply_tupled (
    Fn && fn,
    Tup && args ) -> decltype( detail::apply_tupled( std::forward<Fn>(fn), std::
::forward<Tup>(args), make_index_sequence<std::tuple_size<Tup>::value>() ) ) [inline]
```

Definition at line 436 of file utility.hpp.

5.3.4.5 apply_tupled_as_future()

```
template<typename Fn , typename ArgTup >
auto upcxx::apply_tupled_as_future (
    Fn && fn,
    ArgTup && argtup ) -> decltype( detail::apply_tupled_as_future<Fn,ArgTup>() (
std::forward<Fn>(fn), std::forward<ArgTup>(argtup) ) )
```

Definition at line 102 of file apply.hpp.

5.3.4.6 assert_failed()

```
void upcxx::assert_failed (
    const char * file,
    int line,
    const char * msg = nullptr )
```

Definition at line 42 of file diagnostic.cpp.

5.3.4.7 atomic_fetch_add()

```
template<typename T >
future<T> upcxx::atomic_fetch_add (
    global_ptr< T > p,
    T val,
    std::memory_order order )
```

Definition at line 26 of file atomic.hpp.

5.3.4.8 atomic_get()

```
template<typename T >
future<T> upcxx::atomic_get (
    global_ptr< T > p,
    std::memory_order order )
```

Definition at line 16 of file atomic.hpp.

5.3.4.9 atomic_put()

```
template<typename T >
future upcxx::atomic_put (
    global_ptr< T > p,
    T val,
    std::memory_order order )
```

Definition at line 21 of file atomic.hpp.

5.3.4.10 barrier()

```
void upcxx::barrier ( )
```

Definition at line 169 of file backend.cpp.

5.3.4.11 bind() [1/2]

```
template<typename Fn , typename ... B>
auto upcxx::bind (
    Fn && fn,
    B &&... b ) -> decltype( detail::bind<typename std::decay<Fn>::type>() ( std::
::forward<Fn>(fn), std::forward<B>(b)... ) )
```

Definition at line 292 of file bind.hpp.

5.3.4.12 `bind()` [2/2]

```
template<typename Fn >
Fn&& upcxx::bind (
    Fn && fn )
```

Definition at line 304 of file `bind.hpp`.

5.3.4.13 `bind_last()` [1/2]

```
template<typename ... P>
auto upcxx::bind_last (
    P &&... parm ) -> decltype( detail::bind_last( std::tuple<P&&...>
```

Definition at line 335 of file `bind.hpp`.

5.3.4.14 `bind_last()` [2/2]

```
template<typename Fn >
Fn&& upcxx::bind_last (
    Fn && fn )
```

Definition at line 351 of file `bind.hpp`.

5.3.4.15 `broadcast()`

```
template<typename T1 , typename T = typename std::decay<T1>::type>
future<T> upcxx::broadcast (
    T1 && value,
    intrank_t root )
```

Definition at line 58 of file `broadcast.hpp`.

5.3.4.16 `command_execute()`

```
future upcxx::command_execute (
    parcel_reader & r ) [inline]
```

Definition at line 74 of file `command.hpp`.

5.3.4.17 `command_pack()` [1/2]

```
template<typename Fn >
void upcxx::command_pack (
    parcel_writer & w,
    std::size_t size_ub,
    Fn && fn )
```

5.3.4.18 `command_pack()` [2/2]

```
template<typename Fn1 >
void upcxx::command_pack (
    parcel_writer & w,
    std::size_t size_ub,
    Fn1 && fn ) [inline]
```

Definition at line 90 of file `command.hpp`.

5.3.4.19 `command_size_ubound()` [1/2]

```
template<typename Fn >
void upcxx::command_size_ubound (
    parcel_layout & ub,
    Fn && fn )
```

5.3.4.20 `command_size_ubound()` [2/2]

```
template<typename Fn1 >
void upcxx::command_size_ubound (
    parcel_layout & ub,
    Fn1 && fn ) [inline]
```

Definition at line 80 of file `command.hpp`.

5.3.4.21 `constant()`

```
template<typename T >
constexpr function<T> upcxx::constant (
    T value ) [inline]
```

Definition at line 55 of file `utility.hpp`.

5.3.4.22 dbgbrk()

```
void upcxx::dbgbrk ( )
```

Definition at line 15 of file diagnostic.cpp.

5.3.4.23 deallocate() [1/2]

```
void upcxx::deallocate (
    void * p )
```

Definition at line 186 of file backend.cpp.

5.3.4.24 deallocate() [2/2]

```
template<typename T >
void upcxx::deallocate (
    global_ptr< T > gptr )
```

Definition at line 45 of file allocate.hpp.

5.3.4.25 delete_()

```
template<typename T >
void upcxx::delete_ (
    global_ptr< T > gptr )
```

Definition at line 152 of file allocate.hpp.

5.3.4.26 delete_array()

```
template<typename T >
void upcxx::delete_array (
    global_ptr< T > gptr )
```

Definition at line 169 of file allocate.hpp.

5.3.4.27 fast_hash()

```
template<typename T >
std::size_t upcxx::fast_hash (
    const T & x,
    std::size_t salt = 0 ) [inline]
```

Definition at line 133 of file reflection.hpp.

5.3.4.28 finalize()

```
void upcxx::finalize ( )
```

Definition at line 162 of file backend.cpp.

5.3.4.29 get_or_void()

```
template<int i, typename Tup >
auto upcxx::get_or_void (
    Tup && tup ) -> decltype( detail::tuple\_get\_or\_void<i,Tup>() (std::forward<Tup>(tup))
)
)
```

Definition at line 251 of file utility.hpp.

5.3.4.30 init()

```
void upcxx::init ( )
```

Definition at line 107 of file backend.cpp.

5.3.4.31 is_memory_shared_with()

```
bool upcxx::is_memory_shared_with (
    inrank\_t r ) [inline]
```

Definition at line 22 of file global_ptr.hpp.

5.3.4.32 make_future()

```
template<typename ... T>
auto upcxx::make_future (
    T ... values ) -> decltype(detail::make_future<T...>() (std::declval<T>()...))
```

Definition at line 59 of file make_future.hpp.

5.3.4.33 mod2n_hash()

```
template<typename T >
std::size_t upcxx::mod2n_hash (
    const T & x,
    int bits ) [inline]
```

Definition at line 172 of file reflection.hpp.

5.3.4.34 new_() [1/2]

```
template<typename T , typename ... Args>
global_ptr<T> upcxx::new_ (
    Args &&... args )
```

Definition at line 85 of file allocate.hpp.

5.3.4.35 new_() [2/2]

```
template<typename T , typename ... Args>
global_ptr<T> upcxx::new_ (
    const std::nothrow_t & tag,
    Args &&... args )
```

Definition at line 90 of file allocate.hpp.

5.3.4.36 new_array() [1/2]

```
template<typename T >
global_ptr<T> upcxx::new_array (
    std::size_t n )
```

Definition at line 142 of file allocate.hpp.

5.3.4.37 new_array() [2/2]

```
template<typename T >
global_ptr<T> upcxx::new_array (
    std::size_t n,
    const std::nothrow_t & tag )
```

Definition at line 147 of file allocate.hpp.

5.3.4.38 nop()

```
template<typename Sig >
nop_function<Sig> upcxx::nop ( )
```

Definition at line 36 of file utility.hpp.

5.3.4.39 operator+()

```
template<typename T >
global_ptr<T> upcxx::operator+ (
    std::ptrdiff_t diff,
    global_ptr< T > ptr )
```

Definition at line 171 of file global_ptr.hpp.

5.3.4.40 operator<<() [1/3]

```
std::ostream& upcxx::operator<< (
    std::ostream & o,
    digest x ) [inline]
```

Definition at line 31 of file digest.hpp.

5.3.4.41 operator<<() [2/3]

```
template<typename T >
std::ostream& upcxx::operator<< (
    std::ostream & o,
    dist_id< T > x )
```

Definition at line 78 of file dist_object.hpp.

5.3.4.42 operator<<() [3/3]

```
template<typename T >
std::ostream& upcxx::operator<< (
    std::ostream & os,
    global_ptr< T > ptr )
```

Definition at line 182 of file global_ptr.hpp.

5.3.4.43 operator>>()

```
template<typename ArgKind , typename Fn1 , typename ... ArgT>
auto upcxx::operator>> (
    future1< ArgKind, ArgT... > arg,
    Fn1 && fn ) -> decltype( detail::future_then< future1<ArgKind,ArgT...>, typename
std::decay<Fn1>::type >() ( std::move(arg), std::forward<Fn1>(fn) ) ) [inline]
```

Definition at line 179 of file then.hpp.

5.3.4.44 operator>>=()

```
template<typename Fn1 , typename ... ArgT>
future<ArgT...>& upcxx::operator>>= (
    future< ArgT... > & arg,
    Fn1 && fn ) [inline]
```

Definition at line 199 of file then.hpp.

5.3.4.45 operator" |()

```
template<typename AS , typename AR , typename AO , typename BS , typename BR , typename BO >
completions< typename detail::disjoin_cx<AS, BS, completions<AS, AR, AO>::future_n>::type,
typename detail::disjoin_cx<AR, BR, completions<AS, AR, AO>::future_n>::type, typename detail←
::disjoin_cx<AO, BO, completions<AS, AR, AO>::future_n>::type > upcxx::operator| (
    completions< AS, AR, AO > a,
    completions< BS, BR, BO > b )
```

Definition at line 89 of file completion.hpp.

5.3.4.46 operxn_cx_as_promise()

```
template<typename ... T>
completions<nil_cx, nil_cx, promise_cx<T...> > upcxx::operxn_cx_as_promise (
    promise< T... > & pro )
```

Definition at line 118 of file completion.hpp.

5.3.4.47 os_env() [1/2]

```
template<class T >
T upcxx::os_env (
    const std::string & name )
```

Definition at line 41 of file os_env.hpp.

5.3.4.48 os_env() [2/2]

```
template<class T >
T upcxx::os_env (
    const std::string & name,
    const T & otherwise )
```

Definition at line 52 of file os_env.hpp.

5.3.4.49 print()

```
template<typename T >
print_proxy<T> upcxx::print (
    T const & subject ) [inline]
```

Definition at line 265 of file reflection.hpp.

5.3.4.50 progress()

```
void upcxx::progress (
    progress_level lev = progress_level::user )
```

Definition at line 438 of file backend.cpp.

5.3.4.51 pshm_local_addr2remote()

```
void* upcxx::pshm_local_addr2remote (
    void * addr,
    intrank_t & rank_out ) [inline]
```

Definition at line 30 of file `global_ptr.hpp`.

5.3.4.52 pshm_remote_addr2local()

```
void* upcxx::pshm_remote_addr2local (
    intrank_t r,
    void * addr ) [inline]
```

Definition at line 26 of file `global_ptr.hpp`.

5.3.4.53 rank_me()

```
intrank_t upcxx::rank_me ( ) [inline]
```

Definition at line 98 of file `backend.hpp`.

5.3.4.54 rank_n()

```
intrank_t upcxx::rank_n ( ) [inline]
```

Definition at line 97 of file `backend.hpp`.

5.3.4.55 reflect_upon()

```
template<typename Reflector , typename Subject >
void upcxx::reflect_upon (
    Reflector & re,
    Subject && subj )
```

Definition at line 51 of file `reflection.hpp`.

5.3.4.56 reinterpret_pointer_cast()

```
template<typename T , typename U >
global_ptr<T> upcxx::reinterpret_pointer_cast (
    global_ptr< U > ptr )
```

Definition at line 176 of file global_ptr.hpp.

5.3.4.57 remote_cx_as_rpc()

```
template<typename Fn , typename ... Args>
auto upcxx::remote_cx_as_rpc (
    Fn && fn,
    Args &&... args ) -> completions< nil_cx, rpc_cx<decltype (upcxx::bind (std::
::forward<Fn>(fn), std::forward<Args>(args)...))>, nil_cx >
```

Definition at line 123 of file completion.hpp.

5.3.4.58 rget() [1/2]

```
template<typename T , typename R = nil_cx, typename O = future_cx<0>>
detail::rget_futures_of<detail::rget_byval<T>,R,O>::return_type upcxx::rget (
    global_ptr< T > gp_s,
    completions< nil_cx, R, O > cxs = completions<nil_cx,R,O>{} )
```

Definition at line 168 of file rget.hpp.

5.3.4.59 rget() [2/2]

```
template<typename T , typename R = nil_cx, typename O = future_cx<0>>
detail::rget_futures_of<detail::rget_byref,R,O>::return_type upcxx::rget (
    global_ptr< T > gp_s,
    T * buf_d,
    std::size_t n,
    completions< nil_cx, R, O > cxs = completions<nil_cx,R,O>{} )
```

Definition at line 198 of file rget.hpp.

5.3.4.60 `rpc()` [1/2]

```
template<typename Fn , typename ... Args>
auto upcxx::rpc (
    intrank_t recipient,
    Fn && fn,
    Args &&... args ) -> typename detail::rpc_return<decltype(fn(args...)), Fn,
Args...>::type
```

Definition at line 68 of file `rpc.hpp`.

5.3.4.61 `rpc()` [2/2]

```
template<typename Fn , typename ... T, typename ... Args>
void upcxx::rpc (
    intrank_t recipient,
    promise< T... > & prom,
    Fn && fn,
    Args &&... args )
```

Definition at line 103 of file `rpc.hpp`.

5.3.4.62 `rpc_ff()`

```
template<typename Fn , typename ... Args>
void upcxx::rpc_ff (
    intrank_t recipient,
    Fn && fn,
    Args &&... args )
```

Definition at line 13 of file `rpc.hpp`.

5.3.4.63 `rput()` [1/2]

```
template<typename T , typename S = nil_cx, typename R = nil_cx, typename O = future_cx<0>>
detail::rput_futures_of<S,R,O>::return_type upcxx::rput (
    T value_s,
    global_ptr< T > gp_d,
    completions< S, R, O > cxs = completions<S,R,O>{ } )
```

Definition at line 164 of file `rput.hpp`.

5.3.4.64 `rput()` [2/2]

```
template<typename T , typename S = nil_cx, typename R = nil_cx, typename O = future_cx<0>>
detail::rput_futures_of<S,R,O>::return_type upcxx::rput (
    T const * buf_s,
    std::size_t n,
    global_ptr< T > gp_d,
    completions< S, R, O > cxs = completions<S,R,O>{} )
```

Definition at line 197 of file `rput.hpp`.

5.3.4.65 `source_cx_as_promise()`

```
template<typename ... T>
completions<promise_cx<T...>, nil_cx, nil_cx> upcxx::source_cx_as_promise (
    promise< T... > & pro )
```

Definition at line 113 of file `completion.hpp`.

5.3.4.66 `to_future()` [1/2]

```
template<typename T >
auto upcxx::to_future (
    T x ) -> decltype(make_future(std::forward<T>(x)))
```

Definition at line 79 of file `make_future.hpp`.

5.3.4.67 `to_future()` [2/2]

```
template<typename Kind , typename ... T>
future1<Kind,T...> upcxx::to_future (
    future1< Kind, T... > x )
```

Definition at line 85 of file `make_future.hpp`.

5.3.4.68 `tuple_rvals()`

```
template<typename Tup >
auto upcxx::tuple_rvals (
    Tup && tup ) -> decltype( detail::tuple_rvals( std::forward<Tup>(tup), make_index_sequence<std::tuple_size<typename std::decay<Tup>::type>::value>() ) ) [inline]
```

Definition at line 408 of file `utility.hpp`.

5.3.4.69 wait()

```
template<typename Kind , typename ... T>
auto upcxx::wait (
    future1< Kind, T... > const & f ) -> decltype(f.result())
```

Definition at line 9 of file wait.hpp.

5.3.4.70 when_all()

```
template<typename ... Arg>
detail::when_all_return_t<Arg...> upcxx::when_all (
    Arg ... args )
```

Definition at line 24 of file when_all.hpp.

5.3.5 Variable Documentation

5.3.5.1 dbgbrk_spin

```
bool upcxx::dbgbrk_spin = true
```

Definition at line 12 of file diagnostic.cpp.

5.3.5.2 dbgbrk_spin_init

```
bool upcxx::dbgbrk_spin_init = false
```

Definition at line 11 of file diagnostic.cpp.

5.3.5.3 operxn_cx_as_future

```
constexpr completions<nil_cx, nil_cx, future_cx<0> > upcxx::operxn_cx_as_future = {}
```

Definition at line 110 of file completion.hpp.

5.3.5.4 source_cx_as_future

```
constexpr completions<future_cx<0>, nil_cx, nil_cx> upcxx::source_cx_as_future = {}
```

Definition at line 109 of file completion.hpp.

5.4 upcxx::backend Namespace Reference

Namespaces

- [gasnet1_seq](#)

Classes

- [struct rma_get_cb](#)
- [struct rma_get_cb_wstate](#)
- [struct rma_put_cb](#)
- [struct rma_put_cb_wstate](#)

Functions

- [template<typename Fn1 > void during_user \(Fn1 &&fn\)](#)
- [void during_user \(promise<> &&pro\)](#)
- [void during_user \(promise<> *pro\)](#)
- [template<typename Fn1 > void during_level \(progress_level level, Fn1 &&fn\)](#)
- [template<upcxx::progress_level level, typename Fn > void send_am \(inrank_t recipient, Fn &&fn\)](#)
- [template<typename State , typename SrcCx , typename OpCx > rma_put_cb_wstate< State > * make_rma_put_cb \(State state, SrcCx src_cx, OpCx op_cx\)](#)
- [template<typename State , typename OpCx > rma_get_cb_wstate< State > * make_rma_get_cb \(State state, OpCx op_cx\)](#)
- [template<typename Fn > void during_user \(Fn &&fn\)](#)
- [template<typename Fn > void during_level \(progress_level level, Fn &&fn\)](#)
- [void rma_put \(inrank_t rank_d, void *buf_d, void *buf_s, std::size_t buf_size, rma_put_cb *cb\)](#)
- [void rma_get \(void *buf_d, inrank_t rank_s, void *buf_s, std::size_t buf_size, rma_get_cb *cb\)](#)

Variables

- [inrank_t rank_n](#)
- [inrank_t rank_me](#)

5.4.1 Function Documentation

5.4.1.1 during_level() [1/2]

```
template<typename Fn >
void upcxx::backend::during_level (
    progress_level level,
    Fn && fn )
```

5.4.1.2 during_level() [2/2]

```
template<typename Fn1 >
void upcxx::backend::during_level (
    progress_level level,
    Fn1 && fn ) [inline]
```

Definition at line 114 of file backend.hpp.

5.4.1.3 during_user() [1/4]

```
template<typename Fn >
void upcxx::backend::during_user (
    Fn && fn )
```

5.4.1.4 during_user() [2/4]

```
template<typename Fn1 >
void upcxx::backend::during_user (
    Fn1 && fn ) [inline]
```

Definition at line 87 of file backend.hpp.

5.4.1.5 during_user() [3/4]

```
void upcxx::backend::during_user (
    promise<> && pro ) [inline]
```

Definition at line 101 of file backend.hpp.

5.4.1.6 `during_user()` [4/4]

```
void upcxx::backend::during_user (
    promise<> * pro ) [inline]
```

Definition at line 109 of file backend.hpp.

5.4.1.7 `make_rma_get_cb()`

```
template<typename State , typename OpCx >
rma_get_cb_wstate< State > * upcxx::backend::make_rma_get_cb (
    State state,
    OpCx op_cx ) [inline]
```

Definition at line 226 of file backend.hpp.

5.4.1.8 `make_rma_put_cb()`

```
template<typename State , typename SrcCx , typename OpCx >
rma_put_cb_wstate< State > * upcxx::backend::make_rma_put_cb (
    State state,
    SrcCx src_cx,
    OpCx op_cx ) [inline]
```

Definition at line 213 of file backend.hpp.

5.4.1.9 `rma_get()`

```
void upcxx::backend::rma_get (
    void * buf_d,
    intrank_t rank_s,
    void * buf_s,
    std::size_t buf_size,
    rma_get_cb * cb )
```

5.4.1.10 `rma_put()`

```
void upcxx::backend::rma_put (
    intrank_t rank_d,
    void * buf_d,
    void * buf_s,
    std::size_t buf_size,
    rma_put_cb * cb )
```


5.4.1.11 send_am()

```
template<upcxx::progress_level level, typename Fn >
void upcxx::backend::send_am (
    intrank_t recipient,
    Fn && fn )
```

Definition at line 132 of file backend.hpp.

5.4.2 Variable Documentation

5.4.2.1 rank_me

```
intrank_t upcxx::backend::rank_me
```

Definition at line 30 of file backend.cpp.

5.4.2.2 rank_n

```
intrank_t upcxx::backend::rank_n
```

Definition at line 24 of file backend.cpp.

5.5 upcxx::backend::gasnet1_seq Namespace Reference

Classes

- struct [action](#)
- struct [action_impl](#)
- struct [rma_cb](#)
- struct [rma_get_cb_impl](#)
- struct [rma_put_cb_impl](#)

Functions

- void [send_am_eager_restricted](#) (intrank_t recipient, void *command_buf, std::size_t buf_size, std::size_t buf_align)
- template<typename Fn >
void [send_am_restricted](#) (intrank_t recipient, Fn &&fn)
- void [send_am_eager_queued](#) (progress_level level, intrank_t recipient, void *command_buf, std::size_t buf_size, std::size_t buf_align)
- void [send_am_rdzv](#) (progress_level level, intrank_t recipient, void *command_buf, std::size_t buf_size, std::size_t buf_align)

Variables

- `std::size_t am_size_rdzv_cutover`
- `bool in_user_progress_ = false`
- `action * user_actions_head_ = nullptr`
- `action ** user_actions_tailp_ = &gasnet1_seq::user_actions_head_`

5.5.1 Function Documentation

5.5.1.1 `send_am_eager_queued()`

```
void upcxx::backend::gasnet1_seq::send_am_eager_queued (
    progress_level level,
    intrank_t recipient,
    void * command_buf,
    std::size_t buf_size,
    std::size_t buf_align )
```

Definition at line 245 of file backend.cpp.

5.5.1.2 `send_am_eager_restricted()`

```
void upcxx::backend::gasnet1_seq::send_am_eager_restricted (
    intrank_t recipient,
    void * command_buf,
    std::size_t buf_size,
    std::size_t buf_align )
```

Definition at line 231 of file backend.cpp.

5.5.1.3 `send_am_rdzv()`

```
void upcxx::backend::gasnet1_seq::send_am_rdzv (
    progress_level level,
    intrank_t recipient,
    void * command_buf,
    std::size_t buf_size,
    std::size_t buf_align )
```

5.5.1.4 send_am_restricted()

```
template<typename Fn >
void upcxx::backend::gasnet1_seq::send_am_restricted (
    intrank_t recipient,
    Fn && fn )
```

Definition at line 247 of file backend.hpp.

5.5.2 Variable Documentation

5.5.2.1 am_size_rdzv_cutover

```
size_t upcxx::backend::gasnet1_seq::am_size_rdzv_cutover
```

Definition at line 35 of file backend.cpp.

5.5.2.2 in_user_progress_

```
bool upcxx::backend::gasnet1_seq::in_user_progress_ = false
```

Definition at line 37 of file backend.cpp.

5.5.2.3 user_actions_head_

```
gasnet1_seq::action * upcxx::backend::gasnet1_seq::user_actions_head_ = nullptr
```

Definition at line 39 of file backend.cpp.

5.5.2.4 user_actions_tailp_

```
gasnet1_seq::action ** upcxx::backend::gasnet1_seq::user_actions_tailp_ = &gasnet1_seq::user_↔
actions_head_
```

Definition at line 40 of file backend.cpp.

5.6 upcxx::detail Namespace Reference

Classes

- struct [allreduce_state](#)
- struct [apply_futured_as_future](#)< Fn(future1< Kind, T... >)>
- struct [apply_tupled_as_future](#)
- struct [apply_tupled_as_future_help](#)
- struct [apply_tupled_as_future_help](#)< Fn, ArgTup, std::tuple< T... > >
- struct [apply_tupled_as_future_impl](#)
- struct [apply_tupled_as_future_impl](#)< future1< Kind, T... > >
- struct [apply_tupled_as_future_impl](#)< void >
- struct [bind](#)
- struct [bind](#)< bound_function< Fn0, B0... > >
- struct [bound_function_base](#)
- struct [bound_function_base](#)< Fn, std::tuple< B... >, false >
- struct [bound_function_base](#)< Fn, std::tuple< B... >, true >
- struct [disjoin_cx](#)
- struct [disjoin_cx](#)< A, nil_cx, B_ord_bump >
- struct [disjoin_cx](#)< nil_cx, B, B_ord_bump >
- struct [disjoin_cx](#)< nil_cx, future_cx< B_ord_old >, B_ord_bump >
- struct [disjoin_cx](#)< nil_cx, nil_cx, B_ord0 >
- struct [future_body](#)
- struct [future_body_proxy](#)
- struct [future_body_proxy_](#)
- struct [future_body_pure](#)
- struct [future_body_pure](#)< future1< Kind, T... > >
- struct [future_body_then](#)
- struct [future_body_then_base](#)
- struct [future_body_then_pure](#)
- struct [future_dependency](#)
- struct [future_dependency](#)< future1< future_kind_mapped< FuArg, Fn >, T... > >
- struct [future_dependency](#)< future1< future_kind_result > >
- struct [future_dependency](#)< future1< future_kind_result, T... > >
- struct [future_dependency](#)< future1< future_kind_shref< HeaderOps >, T... > >
- struct [future_dependency](#)< future1< future_kind_when_all< Arg... >, T... > >
- struct [future_dependency_shref_base](#)
- struct [future_dependency_shref_base](#)< false >
- struct [future_dependency_shref_base](#)< true >
- struct [future_dependency_when_all_arg](#)
- struct [future_dependency_when_all_base](#)
- struct [future_dependency_when_all_base](#)< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... > >
- struct [future_from_tuple](#)
- struct [future_from_tuple](#)< Kind, std::tuple< T... > >
- struct [future_header](#)
- struct [future_header_dependent](#)
- struct [future_header_ops_general](#)
- struct [future_header_ops_result](#)
- struct [future_header_ops_result_ready](#)
- struct [future_header_result](#)
- struct [future_header_result](#)<>
- struct [future_impl_mapped](#)
- struct [future_impl_result](#)

- struct [future_impl_result<>](#)
- struct [future_impl_shref](#)
- struct [future_impl_when_all](#)
- struct [future_impl_when_all< std::tuple< FuArg... >, T... >](#)
- struct [future_kind_mapped](#)
- struct [future_kind_result](#)
- struct [future_kind_shref](#)
- struct [future_kind_when_all](#)
- struct [future_then](#)
- struct [future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, false >](#)
- struct [future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, true >](#)
- struct [future_then_pure](#)
- struct [future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, false >](#)
- struct [future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, true >](#)
- struct [future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, true, fnret_trivial >](#)
- struct [global_ptr_ctor_internal](#)
- struct [is_future_cx](#)
- struct [is_future_cx< future_cx< ordinal > >](#)
- struct [make_future](#)
- struct [make_future_](#)
- struct [make_future_< false, T... >](#)
- struct [make_future_< true, T... >](#)
- struct [make_index_sequence](#)
- struct [make_index_sequence< 0, s... >](#)
- struct [os_env_parse](#)
- struct [os_env_parse< std::string >](#)
- struct [packing_tuple_each](#)
- struct [packing_tuple_each< n, n, T... >](#)
- struct [promise_like](#)
- struct [promise_like< future1< Kind, T... > >](#)
- struct [rget_byval](#)
- struct [rget_futures_of](#)
- struct [rget_futures_of< rget_byref, R, future_cx< 0 > >](#)
- struct [rget_futures_of< rget_byval< T >, R, future_cx< 0 > >](#)
- struct [rget_state_operxn](#)
- struct [rget_state_operxn< rget_byref, future_cx< ordinal > >](#)
- struct [rget_state_operxn< rget_byref, nil_cx >](#)
- struct [rget_state_operxn< rget_byref, promise_cx<> >](#)
- struct [rget_state_operxn< rget_byval< T >, future_cx< ordinal > >](#)
- struct [rget_state_operxn< rget_byval< T >, nil_cx >](#)
- struct [rget_state_operxn< rget_byval< T >, promise_cx< T > >](#)
- struct [rget_state_remote](#)
- struct [rget_state_remote< nil_cx >](#)
- struct [rget_state_remote< rpc_cx< Fn > >](#)
- struct [rget_states](#)
- struct [rpc_recipient_after](#)
- struct [rpc_return](#)
- struct [rput_byval](#)
- struct [rput_futures_of](#)
- struct [rput_futures_of< future_cx< S_ord >, R, future_cx< O_ord > >](#)
- struct [rput_futures_of< future_cx< S_ord >, R, O >](#)
- struct [rput_futures_of< S, R, future_cx< O_ord > >](#)
- struct [rput_state_here](#)
- struct [rput_state_here< future_cx< ordinal > >](#)
- struct [rput_state_here< nil_cx >](#)

- struct [rput_state_here](#)< [promise_cx](#)<> >
- struct [rput_state_remote](#)
- struct [rput_state_remote](#)< [nil_cx](#) >
- struct [rput_state_remote](#)< [rpc_cx](#)< [Fn](#) > >
- struct [rput_states](#)
- struct [rput_states](#)< [rput_byref](#), [S](#), [R](#), [O](#) >
- struct [rput_states](#)< [rput_byval](#)< [T](#) >, [S](#), [R](#), [O](#) >
- struct [tuple_get_or_void](#)
- struct [tuple_get_or_void](#)< [i](#), [TupRef](#), [false](#) >
- struct [tuple_rvals_get](#)
- struct [tuple_rvals_get](#)< [Tup](#) &&, [i](#), [Ti](#) & >
- struct [tuple_rvals_get](#)< [Tup](#) &&, [i](#), [Ti](#) && >
- struct [tuple_rvals_get](#)< [Tup](#) &&, [i](#), [Ti](#) >
- struct [tuple_rvals_get](#)< [Tup](#) &, [i](#), [Ti](#) & >
- struct [tuple_rvals_get](#)< [Tup](#) &, [i](#), [Ti](#) && >
- struct [tuple_rvals_get](#)< [Tup](#) &, [i](#), [Ti](#) >
- struct [tuple_rvals_get](#)< [Tup](#) const &, [i](#), [Ti](#) & >
- struct [tuple_rvals_get](#)< [Tup](#) const &, [i](#), [Ti](#) && >
- struct [tuple_rvals_get](#)< [Tup](#) const &, [i](#), [Ti](#) >

Typedefs

- template<typename [App](#) >
using [apply_futured_as_future_return_t](#) = typename [apply_futured_as_future](#)< [App](#) >::return_type
- template<typename [Kind](#) , typename [Tup](#) >
using [future_from_tuple_t](#) = typename [future_from_tuple](#)< [Kind](#), [Tup](#) >::type
- template<typename [Fu](#) >
using [promise_like_t](#) = typename [promise_like](#)< [Fu](#) >::type
- template<typename ... [Arg](#)>
using [when_all_return_t](#) = [future_from_tuple_t](#)< [future_kind_when_all](#)< [Arg...](#) >, [decltype](#)([std::tuple](#)_←
[cat](#)([std::declval](#)< typename [Arg](#)::results_type >()...)) >

Functions

- template<bool [throws](#), typename [T](#) , typename ... [Args](#)>
[global_ptr](#)< [T](#) > [new_](#) ([Args](#) &&...args)
- template<bool [throws](#), typename [T](#) >
[global_ptr](#)< [T](#) > [new_array](#) ([std::size_t](#) n)
- template<typename ... [P](#), int ... [heads](#), int [tail](#)>
auto [bind_last](#) ([std::tuple](#)< [P...](#) > [parms](#), [upcxx::index_sequence](#)< [heads...](#) >, [std::integral_constant](#)< [int](#),
[tail](#) >) -> [decltype](#)([detail::bind](#)< typename [std::decay](#)< typename [std::tuple_element](#)< [tail](#), [std::tuple](#)< [P...](#)
>>::type >::type >()([std::move](#)([std::get](#)< [tail](#) >(parms)), [std::move](#)([std::get](#)< [heads](#) >(parms))...))
- template<typename [T](#) >
void [broadcast_receive](#) ([T](#) const &value, [inrank_t](#) rank_ub, [dist_id](#)< [promise](#)< [T](#) >> id)
- template<typename [Fn](#) >
[future_command_executor](#) ([parcel_reader](#) &r)
- template<typename [T](#) >
[promise](#)< [dist_object](#)< [T](#) > & > * [dist_promise](#) ([digest](#) id)
- void [packing_funptr_basis](#) ()
- template<typename [Tup](#) , int ... [i](#)>
auto [tuple_rvals](#) ([Tup](#) &&tup, [index_sequence](#)< [i...](#) >) -> [std::tuple](#)< [decltype](#)([tuple_rvals_get](#)< [Tup](#) &&, [i](#)
>()(tup))... >
- template<typename [Fn](#) , typename [Tup](#) , int ... [i](#)>
auto [apply_tupled](#) ([Fn](#) &&fn, [Tup](#) &&args, [index_sequence](#)< [i...](#) >) -> [decltype](#)(fn([std::get](#)< [i](#) >(args)...))

Variables

- `std::unordered_map< digest, void * > dist_master_promises`
- `std::uint64_t dist_master_id_bump = 0`

5.6.1 Typedef Documentation

5.6.1.1 `apply_futured_as_future_return_t`

```
template<typename App >  
using upcxx::detail::apply_futured_as_future_return_t = typedef typename apply_futured_as_future_return_t<App>::return_type
```

Definition at line 115 of file core.hpp.

5.6.1.2 `future_from_tuple_t`

```
template<typename Kind , typename Tup >  
using upcxx::detail::future_from_tuple_t = typedef typename future_from_tuple<Kind,Tup>::type
```

Definition at line 131 of file core.hpp.

5.6.1.3 `promise_like_t`

```
template<typename Fu >  
using upcxx::detail::promise_like_t = typedef typename promise_like<Fu>::type
```

Definition at line 25 of file promise.hpp.

5.6.1.4 `when_all_return_t`

```
template<typename ... Arg>  
using upcxx::detail::when_all_return_t = typedef future_from_tuple_t< future_kind_when_all<Arg...>, decltype(std::tuple_cat( std::declval<typename Arg::results_type>()... )) >
```

Definition at line 20 of file when_all.hpp.

5.6.2 Function Documentation

5.6.2.1 apply_tupled()

```
template<typename Fn , typename Tup , int ... i>
auto upcxx::detail::apply_tupled (
    Fn && fn,
    Tup && args,
    index_sequence< i... > ) -> decltype(fn(std::get<i>(args)...))    [inline]
```

Definition at line 427 of file utility.hpp.

5.6.2.2 bind_last()

```
template<typename ... P, int ... heads, int tail>
auto upcxx::detail::bind_last (
    std::tuple< P... > parms,
    upcxx::index_sequence< heads... > ,
    std::integral_constant< int, tail > ) -> decltype( detail::bind<typename std::
::decay<typename std::tuple_element<tail, std::tuple<P...>>::type>::type>() ( std::move(std::
::get<tail>(parms)), std::move(std::get<heads>(parms))... ) )
```

Definition at line 316 of file bind.hpp.

5.6.2.3 broadcast_receive()

```
template<typename T >
void upcxx::detail::broadcast_receive (
    T const & value,
    intrank_t rank_ub,
    dist_id< promise< T >> id )
```

Definition at line 14 of file broadcast.hpp.

5.6.2.4 command_executor()

```
template<typename Fn >
future upcxx::detail::command_executor (
    parcel_reader & r )
```

Definition at line 69 of file command.hpp.

5.6.2.5 dist_promise()

```
template<typename T >
promise<dist_object<T>&>* upcxx::detail::dist_promise (
    digest id )
```

Definition at line 31 of file dist_object.hpp.

5.6.2.6 new_()

```
template<bool throws, typename T , typename ... Args>
global_ptr<T> upcxx::detail::new_ (
    Args &&... args )
```

Definition at line 58 of file allocate.hpp.

5.6.2.7 new_array()

```
template<bool throws, typename T >
global_ptr<T> upcxx::detail::new_array (
    std::size_t n )
```

Definition at line 96 of file allocate.hpp.

5.6.2.8 packing_funptr_basis()

```
void upcxx::detail::packing_funptr_basis ( )
```

Definition at line 3 of file packing.cpp.

5.6.2.9 tuple_rvals()

```
template<typename Tup , int ... i>
auto upcxx::detail::tuple_rvals (
    Tup && tup,
    index_sequence< i... > ) -> std::tuple<decltype(tuple_rvals_get<Tup&&, i>()(tup))...>
[inline]
```

Definition at line 399 of file utility.hpp.

5.6.3 Variable Documentation

5.6.3.1 `dist_master_id_bump`

```
uint64_t upcxx::detail::dist_master_id_bump = 0
```

Definition at line 8 of file `dist_object.cpp`.

5.6.3.2 `dist_master_promises`

```
unordered_map< digest, void * > upcxx::detail::dist_master_promises
```

Definition at line 6 of file `dist_object.cpp`.

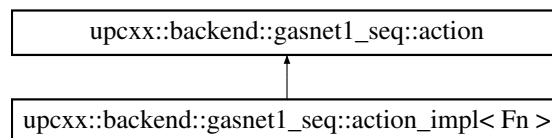
Chapter 6

Class Documentation

6.1 upcxx::backend::gasnet1_seq::action Struct Reference

```
#include <backend.hpp>
```

Inheritance diagram for upcxx::backend::gasnet1_seq::action:



Public Member Functions

- virtual void [fire_and_delete](#) ()=0

Public Attributes

- [action](#) * [next_](#)

6.1.1 Detailed Description

Definition at line 18 of file backend.hpp.

6.1.2 Member Function Documentation

6.1.2.1 fire_and_delete()

```
virtual void upcxx::backend::gasnet1_seq::action::fire_and_delete ( ) [pure virtual]
```

Implemented in [upcxx::backend::gasnet1_seq::action_impl< Fn >](#).

6.1.3 Member Data Documentation

6.1.3.1 next_

```
action* upcxx::backend::gasnet1_seq::action::next_
```

Definition at line 19 of file backend.hpp.

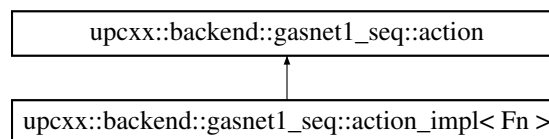
The documentation for this struct was generated from the following file:

- [backend/gasnet1_seq/backend.hpp](#)

6.2 upcxx::backend::gasnet1_seq::action_impl< Fn > Struct Template Reference

```
#include <backend.hpp>
```

Inheritance diagram for upcxx::backend::gasnet1_seq::action_impl< Fn >:



Public Member Functions

- [action_impl](#) (Fn fn)
- void [fire_and_delete](#) ()

Public Attributes

- Fn fn

6.2.1 Detailed Description

```
template<typename Fn>
struct upcxx::backend::gasnet1_seq::action_impl< Fn >
```

Definition at line 75 of file backend.hpp.

6.2.2 Constructor & Destructor Documentation

6.2.2.1 action_impl()

```
template<typename Fn >  
upcxx::backend::gasnet1_seq::action_impl< Fn >::action_impl (   
    Fn fn ) [inline]
```

Definition at line 77 of file backend.hpp.

6.2.3 Member Function Documentation

6.2.3.1 fire_and_delete()

```
template<typename Fn >  
void upcxx::backend::gasnet1_seq::action_impl< Fn >::fire_and_delete ( ) [inline], [virtual]
```

Implements `upcxx::backend::gasnet1_seq::action`.

Definition at line 79 of file backend.hpp.

6.2.4 Member Data Documentation

6.2.4.1 fn

```
template<typename Fn >  
Fn upcxx::backend::gasnet1_seq::action_impl< Fn >::fn
```

Definition at line 76 of file backend.hpp.

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

- [backend/gasnet1_seq/backend.hpp](#)

6.3 upcxx::add_lref_if_nonref< T > Struct Template Reference

```
#include <utility.hpp>
```

Public Types

- using `type` = T &

6.3.1 Detailed Description

```
template<typename T>  
struct upcxx::add_lref_if_nonref< T >
```

Definition at line 210 of file utility.hpp.

6.3.2 Member Typedef Documentation

6.3.2.1 type

```
template<typename T >  
using upcxx::add_lref_if_nonref< T >::type = T&
```

Definition at line 210 of file utility.hpp.

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

- [utility.hpp](#)

6.4 upcxx::add_lref_if_nonref< T & > Struct Template Reference

```
#include <utility.hpp>
```

Public Types

- using [type](#) = T &

6.4.1 Detailed Description

```
template<typename T>  
struct upcxx::add_lref_if_nonref< T & >
```

Definition at line 213 of file utility.hpp.

6.4.2 Member Typedef Documentation

6.4.2.1 type

```
template<typename T >  
using upcxx::add_lref_if_nonref< T & >::type = T&
```

Definition at line 213 of file utility.hpp.

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

- [utility.hpp](#)

6.5 upcxx::add_lref_if_nonref< T && > Struct Template Reference

```
#include <utility.hpp>
```

Public Types

- using `type` = T &&

6.5.1 Detailed Description

```
template<typename T>  
struct upcxx::add_lref_if_nonref< T && >
```

Definition at line 216 of file utility.hpp.

6.5.2 Member Typedef Documentation

6.5.2.1 type

```
template<typename T >  
using upcxx::add_lref_if_nonref< T && >::type = T&&
```

Definition at line 216 of file utility.hpp.

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

- [utility.hpp](#)

6.6 upcxx::detail::allreduce_state< T, Op > Struct Template Reference

```
#include <allreduce.hpp>
```

Public Member Functions

- void `contributed` (`dist_object`< `allreduce_state` > &`this_obj`)
- void `broadcast` (`dist_object`< `allreduce_state` > &`this_obj`, T const &`value`, `inrank_t` `rank_ub`)

Public Attributes

- int `incoming`
- Op `op`
- T `accum`
- `promise`< T > `answer`

6.6.1 Detailed Description

```
template<typename T, typename Op>
struct upcxx::detail::allreduce_state< T, Op >
```

Definition at line 11 of file `allreduce.hpp`.

6.6.2 Member Function Documentation

6.6.2.1 broadcast()

```
template<typename T , typename Op >
void upcxx::detail::allreduce_state< T, Op >::broadcast (
    dist_object< allreduce_state< T, Op > > & this_obj,
    T const & value,
    inrank_t rank_ub )
```

Definition at line 95 of file `allreduce.hpp`.

6.6.2.2 contributed()

```
template<typename T , typename Op >
void upcxx::detail::allreduce_state< T, Op >::contributed (
    dist_object< allreduce_state< T, Op > > & this_obj )
```

Definition at line 68 of file `allreduce.hpp`.

6.6.3 Member Data Documentation

6.6.3.1 accum

```
template<typename T , typename Op >
T upcxx::detail::allreduce_state< T, Op >::accum
```

Definition at line 14 of file allreduce.hpp.

6.6.3.2 answer

```
template<typename T , typename Op >
promise<T> upcxx::detail::allreduce_state< T, Op >::answer
```

Definition at line 15 of file allreduce.hpp.

6.6.3.3 incoming

```
template<typename T , typename Op >
int upcxx::detail::allreduce_state< T, Op >::incoming
```

Definition at line 12 of file allreduce.hpp.

6.6.3.4 op

```
template<typename T , typename Op >
Op upcxx::detail::allreduce_state< T, Op >::op
```

Definition at line 13 of file allreduce.hpp.

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

- [allreduce.hpp](#)

6.7 upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator< Arg > Struct Template Reference

```
#include <bind.hpp>
```

Public Member Functions

- template<int ... ai>
auto [apply](#)_(upcxx::index_sequence< ai... >, typename [binding](#)< Fn >::off_wire_type &fn, typename [binding](#)< B >::off_wire_type &...b) -> decltype(fn(b..., std::get< ai >(a)...))
- auto [operator](#)() (typename [binding](#)< Fn >::off_wire_type &fn, typename [binding](#)< B >::off_wire_type &...b) -> decltype([apply](#)_(upcxx::make_index_sequence< sizeof...(Arg)>

Public Attributes

- `std::tuple< Arg... >` `a`
- `fn`
- `b`

6.7.1 Detailed Description

```
template<typename Fn, typename ... B>
template<typename ... Arg>
struct upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator< Arg >
```

Definition at line 128 of file bind.hpp.

6.7.2 Member Function Documentation

6.7.2.1 apply_()

```
template<typename Fn , typename ... B>
template<typename ... Arg>
template<int ... ai>
auto upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator< Arg
>::apply_ (
    upcxx::index_sequence< ai... > ,
    typename binding< Fn >::off_wire_type & fn,
    typename binding< B >::off_wire_type &... b ) -> decltype( fn(b..., std::get<ai>(a)...))
[inline]
```

Definition at line 132 of file bind.hpp.

6.7.2.2 operator>()

```
template<typename Fn , typename ... B>
template<typename ... Arg>
auto upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator< Arg
>::operator() (
    typename binding< Fn >::off_wire_type & fn,
    typename binding< B >::off_wire_type &... b ) -> decltype( apply_(upcxx::make←
_index_sequence<sizeof...(Arg)> [inline]
```

Definition at line 141 of file bind.hpp.

6.7.3 Member Data Documentation

6.7.3.1 a

```
template<typename Fn , typename ... B>
template<typename ... Arg>
std::tuple<Arg...> upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >←
::applicator< Arg >::a
```

Definition at line 129 of file bind.hpp.

6.7.3.2 b

```
template<typename Fn , typename ... B>
template<typename ... Arg>
upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator< Arg >::b
```

Initial value:

```
{
    return apply_(upcxx::make_index_sequence<sizeof...(Arg)>{},
        fn, b...)
```

Definition at line 147 of file bind.hpp.

6.7.3.3 fn

```
template<typename Fn , typename ... B>
template<typename ... Arg>
upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator< Arg >::fn
```

Definition at line 146 of file bind.hpp.

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

- [bind.hpp](#)

6.8 upcxx::detail::apply_futured_as_future< Fn(future1< Kind, T... >)> Struct Template Reference

```
#include <apply.hpp>
```

Public Types

- using [tupled_impl](#) = [apply_tupled_as_future](#)< Fn, std::tuple< typename [upcxx::add_lref_if_nonref](#)< T >←
::type... > >
- using [return_type](#) = typename [tupled_impl](#)::return_type

Public Member Functions

- `template<typename Fn1 >`
`return_type operator() (Fn1 &&fn, future1< Kind, T... > const &arg)`
- `template<typename Fn1 >`
`return_type operator() (Fn1 &&fn, future_dependency< future1< Kind, T... >> const &arg)`

6.8.1 Detailed Description

```
template<typename Fn, typename Kind, typename ... T>
struct upcxx::detail::apply_futured_as_future< Fn(future1< Kind, T... >)>
```

Definition at line 74 of file apply.hpp.

6.8.2 Member Typedef Documentation

6.8.2.1 return_type

```
template<typename Fn , typename Kind , typename ... T>
using upcxx::detail::apply_futured_as_future< Fn(future1< Kind, T... >)>::return_type =
typename tupled_impl::return_type
```

Definition at line 80 of file apply.hpp.

6.8.2.2 tupled_impl

```
template<typename Fn , typename Kind , typename ... T>
using upcxx::detail::apply_futured_as_future< Fn(future1< Kind, T... >)>::tupled_impl =
apply_tupled_as_future< Fn, std::tuple<typename upcxx::add_lref_if_nonref<T>::type...> >
```

Definition at line 78 of file apply.hpp.

6.8.3 Member Function Documentation

6.8.3.1 operator() [1/2]

```
template<typename Fn , typename Kind , typename ... T>
template<typename Fn1 >
return_type upcxx::detail::apply_futured_as_future< Fn(future1< Kind, T... >)>::operator() (
    Fn1 && fn,
    future1< Kind, T... > const & arg ) [inline]
```

Definition at line 83 of file apply.hpp.

6.8.3.2 operator() [2/2]

```

template<typename Fn , typename Kind , typename ... T>
template<typename Fn1 >
return_type upcxx::detail::apply_futured_as_future< Fn(future1< Kind, T... >)>::operator() (
    Fn1 && fn,
    future_dependency< future1< Kind, T... >> const & arg ) [inline]

```

Definition at line 91 of file apply.hpp.

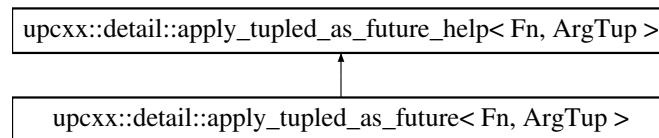
The documentation for this struct was generated from the following file:

- [future/apply.hpp](#)

6.9 upcxx::detail::apply_tupled_as_future< Fn, ArgTup > Struct Template Reference

```
#include <apply.hpp>
```

Inheritance diagram for upcxx::detail::apply_tupled_as_future< Fn, ArgTup >:



6.9.1 Detailed Description

```

template<typename Fn, typename ArgTup>
struct upcxx::detail::apply_tupled_as_future< Fn, ArgTup >

```

Definition at line 69 of file apply.hpp.

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

- [future/apply.hpp](#)

6.10 upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, ArgTupDecayed > Struct Template Reference

```
#include <apply.hpp>
```

6.10.1 Detailed Description

```
template<typename Fn, typename ArgTup, typename ArgTupDecayed = typename std::decay<ArgTup>::type>
struct upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, ArgTupDecayed >
```

Definition at line 56 of file apply.hpp.

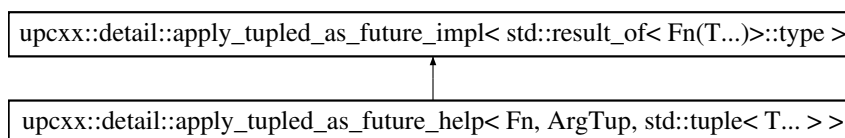
The documentation for this struct was generated from the following file:

- [future/apply.hpp](#)

6.11 upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, std::tuple< T... > > Struct Template Reference

```
#include <apply.hpp>
```

Inheritance diagram for upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, std::tuple< T... > >:



Additional Inherited Members

6.11.1 Detailed Description

```
template<typename Fn, typename ArgTup, typename ... T>
struct upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, std::tuple< T... > >
```

Definition at line 59 of file apply.hpp.

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

- [future/apply.hpp](#)

6.12 upcxx::detail::apply_tupled_as_future_impl< Return > Struct Template Reference

```
#include <apply.hpp>
```

Public Member Functions

- `template<typename Fn, typename ArgTup >`
[return_type operator\(\)](#) (Fn &&fn, ArgTup &&argtup)

Public Attributes

- `decltype(upcxx::make_future< Return >(std::declval< Return >()))` typedef `return_type`

6.12.1 Detailed Description

```
template<typename Return>
struct upcxx::detail::apply_tupled_as_future_impl< Return >
```

Definition at line 13 of file `apply.hpp`.

6.12.2 Member Function Documentation

6.12.2.1 `operator()`

```
template<typename Return>
template<typename Fn , typename ArgTup >
return_type upcxx::detail::apply_tupled_as_future_impl< Return >::operator() (
    Fn && fn,
    ArgTup && argtup ) [inline]
```

Definition at line 33 of file `apply.hpp`.

6.12.3 Member Data Documentation

6.12.3.1 `return_type`

```
template<typename Return>
decltype(upcxx::make_future<Return>(std::declval<Return>())) typedef upcxx::detail::apply_↵
tupled_as_future_impl< Return >::return_type
```

Definition at line 30 of file `apply.hpp`.

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

- `future/apply.hpp`

6.13 `upcxx::detail::apply_tupled_as_future_impl< future1< Kind, T... > >` Struct Template Reference

```
#include <apply.hpp>
```

Public Types

- typedef [future1](#)< Kind, T... > [return_type](#)

Public Member Functions

- template<typename Fn , typename ArgTup >
[return_type operator\(\)](#) (Fn &&fn, ArgTup &&argtup)

6.13.1 Detailed Description

```
template<typename Kind, typename ... T>
struct upcxx::detail::apply_tupled_as_future_impl< future1< Kind, T... > >
```

Definition at line 42 of file apply.hpp.

6.13.2 Member Typedef Documentation

6.13.2.1 return_type

```
template<typename Kind , typename ... T>
typedef future1<Kind,T...> upcxx::detail::apply\_tupled\_as\_future\_impl< future1< Kind, T...
> >::return_type
```

Definition at line 43 of file apply.hpp.

6.13.3 Member Function Documentation

6.13.3.1 operator>()

```
template<typename Kind , typename ... T>
template<typename Fn , typename ArgTup >
return\_type upcxx::detail::apply\_tupled\_as\_future\_impl< future1< Kind, T... > >::operator()
(
    Fn && fn,
    ArgTup && argtup ) [inline]
```

Definition at line 46 of file apply.hpp.

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

- [future/apply.hpp](#)

6.14 upcxx::detail::apply_tupled_as_future_impl< void > Struct Template Reference

```
#include <apply.hpp>
```

Public Member Functions

- `template<typename Fn , typename ArgTup > return_type operator() (Fn &&fn, ArgTup &&argtup)`

Public Attributes

- `decltype(upcxx::make_future<>()) typedef return_type`

6.14.1 Detailed Description

```
template<>
struct upcxx::detail::apply_tupled_as_future_impl< void >
```

Definition at line 17 of file apply.hpp.

6.14.2 Member Function Documentation

6.14.2.1 operator()

```
template<typename Fn , typename ArgTup >
return_type upcxx::detail::apply_tupled_as_future_impl< void >::operator() (
    Fn && fn,
    ArgTup && argtup ) [inline]
```

Definition at line 21 of file apply.hpp.

6.14.3 Member Data Documentation

6.14.3.1 return_type

```
decltype(upcxx::make_future<>()) typedef upcxx::detail::apply_tupled_as_future_impl< void >↔
::return_type
```

Definition at line 18 of file apply.hpp.

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

- [future/apply.hpp](#)

6.15 `upcxx::detail::bind`< `FnDecayed` > Struct Template Reference

```
#include <bind.hpp>
```

Public Member Functions

- `template<typename Fn , typename ... B>`
`bound_function`< `Fn` &&, `B` &&... > `operator()` (`Fn` &&`fn`, `B` &&...`b`) const

6.15.1 Detailed Description

```
template<typename FnDecayed>
struct upcxx::detail::bind< FnDecayed >
```

Definition at line 261 of file `bind.hpp`.

6.15.2 Member Function Documentation

6.15.2.1 `operator()`

```
template<typename FnDecayed >
template<typename Fn , typename ... B>
bound_function<Fn&&, B&&...> upcxx::detail::bind< FnDecayed >::operator() (
    Fn && fn,
    B &&... b ) const [inline]
```

Definition at line 263 of file `bind.hpp`.

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

- [bind.hpp](#)

6.16 `upcxx::detail::bind`< `bound_function`< `Fn0`, `B0`... > > Struct Template Reference

```
#include <bind.hpp>
```

Public Member Functions

- `template<typename Bf , typename ... B1>`
`bound_function`< `Fn0`, `B0`..., `B1` &&... > `operator()` (`Bf` &&`bf`, `B1` &&...`b1`) const

6.16.1 Detailed Description

```
template<typename Fn0, typename ... B0>
struct upcxx::detail::bind< bound_function< Fn0, B0... > >
```

Definition at line 275 of file bind.hpp.

6.16.2 Member Function Documentation

6.16.2.1 operator>()

```
template<typename Fn0 , typename ... B0>
template<typename Bf , typename ... B1>
bound_function<Fn0, B0..., B1&&...> upcxx::detail::bind< bound_function< Fn0, B0... > >↔
::operator() (
    Bf && bf,
    B1 &&... b1 ) const [inline]
```

Definition at line 277 of file bind.hpp.

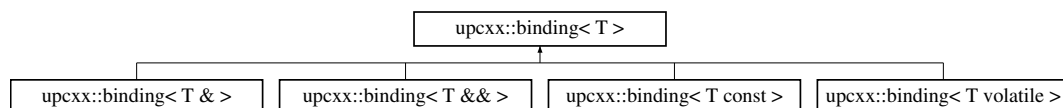
The documentation for this struct was generated from the following file:

- [bind.hpp](#)

6.17 upcxx::binding< T > Struct Template Reference

```
#include <bind.hpp>
```

Inheritance diagram for upcxx::binding< T >:



Public Types

- using [on_wire_type](#) = T
- using [off_wire_type](#) = T

Static Public Member Functions

- static T [on_wire](#) (T x)
- static auto [off_wire](#) (T x) -> decltype([make_future](#)(std::move(x)))

Static Public Attributes

- static constexpr bool [is_trivial](#) = true

6.17.1 Detailed Description

```
template<typename T>  
struct upcxx::binding< T >
```

Definition at line 31 of file bind.hpp.

6.17.2 Member Typedef Documentation

6.17.2.1 off_wire_type

```
template<typename T>  
using upcxx::binding< T >::off_wire_type = T
```

Definition at line 33 of file bind.hpp.

6.17.2.2 on_wire_type

```
template<typename T>  
using upcxx::binding< T >::on_wire_type = T
```

Definition at line 32 of file bind.hpp.

6.17.3 Member Function Documentation

6.17.3.1 off_wire()

```
template<typename T>  
static auto upcxx::binding< T >::off_wire (  
    T x ) -> decltype(make_future(std::move(x)))    [inline], [static]
```

Definition at line 39 of file bind.hpp.

6.17.3.2 on_wire()

```
template<typename T>
static T upcxx::binding< T >::on_wire (
    T x ) [inline], [static]
```

Definition at line 35 of file bind.hpp.

6.17.4 Member Data Documentation

6.17.4.1 is_trivial

```
template<typename T>
constexpr bool upcxx::binding< T >::is_trivial = true [static]
```

Definition at line 46 of file bind.hpp.

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

- [bind.hpp](#)

6.18 upcxx::binding< dist_object< T > & > Struct Template Reference

```
#include <dist_object.hpp>
```

Public Types

- using [on_wire_type](#) = [dist_id](#)< T >
- using [off_wire_type](#) = [dist_object](#)< T > &

Static Public Member Functions

- static [dist_id](#)< T > [on_wire](#) ([dist_object](#)< T > &o)
- static [future](#)< [dist_object](#)< T > & > [off_wire](#) ([dist_id](#)< T > id)

6.18.1 Detailed Description

```
template<typename T>
struct upcxx::binding< dist_object< T > & >
```

Definition at line 153 of file dist_object.hpp.

6.18.2 Member Typedef Documentation

6.18.2.1 off_wire_type

```
template<typename T >  
using upcxx::binding< dist_object< T > & >::off_wire_type = dist_object<T>&
```

Definition at line 155 of file dist_object.hpp.

6.18.2.2 on_wire_type

```
template<typename T >  
using upcxx::binding< dist_object< T > & >::on_wire_type = dist_id<T>
```

Definition at line 154 of file dist_object.hpp.

6.18.3 Member Function Documentation

6.18.3.1 off_wire()

```
template<typename T >  
static future<dist_object<T>&> upcxx::binding< dist_object< T > & >::off_wire (  
    dist_id< T > id ) [inline], [static]
```

Definition at line 161 of file dist_object.hpp.

6.18.3.2 on_wire()

```
template<typename T >  
static dist_id<T> upcxx::binding< dist_object< T > & >::on_wire (  
    dist_object< T > & o ) [inline], [static]
```

Definition at line 157 of file dist_object.hpp.

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

- [dist_object.hpp](#)

6.19 upcxx::binding< dist_object< T > && > Struct Template Reference

```
#include <dist_object.hpp>
```

6.19.1 Detailed Description

```
template<typename T>  
struct upcxx::binding< dist_object< T > && >
```

Definition at line 167 of file dist_object.hpp.

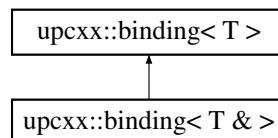
The documentation for this struct was generated from the following file:

- [dist_object.hpp](#)

6.20 upcxx::binding< T & > Struct Template Reference

```
#include <bind.hpp>
```

Inheritance diagram for upcxx::binding< T & >:



Additional Inherited Members

6.20.1 Detailed Description

```
template<typename T>  
struct upcxx::binding< T & >
```

Definition at line 60 of file bind.hpp.

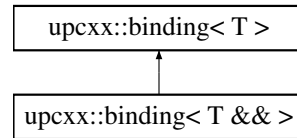
The documentation for this struct was generated from the following file:

- [bind.hpp](#)

6.21 `upcxx::binding< T && >` Struct Template Reference

```
#include <bind.hpp>
```

Inheritance diagram for `upcxx::binding< T && >`:



Additional Inherited Members

6.21.1 Detailed Description

```
template<typename T>  
struct upcxx::binding< T && >
```

Definition at line 62 of file `bind.hpp`.

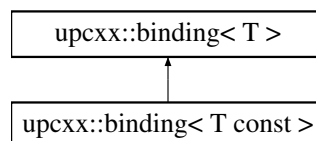
The documentation for this struct was generated from the following file:

- [bind.hpp](#)

6.22 `upcxx::binding< T const >` Struct Template Reference

```
#include <bind.hpp>
```

Inheritance diagram for `upcxx::binding< T const >`:



Additional Inherited Members

6.22.1 Detailed Description

```
template<typename T>  
struct upcxx::binding< T const >
```

Definition at line 64 of file `bind.hpp`.

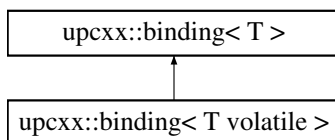
The documentation for this struct was generated from the following file:

- [bind.hpp](#)

6.23 upcxx::binding< T volatile > Struct Template Reference

```
#include <bind.hpp>
```

Inheritance diagram for upcxx::binding< T volatile >:



Additional Inherited Members

6.23.1 Detailed Description

```
template<typename T>  
struct upcxx::binding< T volatile >
```

Definition at line 66 of file bind.hpp.

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

- [bind.hpp](#)

6.24 upcxx::binding_is_trivial< T, trivial > Struct Template Reference

```
#include <bind.hpp>
```

Static Public Attributes

- static constexpr bool [value](#) = trivial::value

6.24.1 Detailed Description

```
template<typename T, typename trivial = std::false_type>  
struct upcxx::binding_is_trivial< T, trivial >
```

Definition at line 50 of file bind.hpp.

6.24.2 Member Data Documentation

6.24.2.1 value

```
template<typename T , typename trivial = std::false_type>
constexpr bool upcxx::binding\_is\_trivial< T, trivial >::value = trivial::value [static]
```

Definition at line 51 of file bind.hpp.

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

- [bind.hpp](#)

6.25 [upcxx::binding_is_trivial](#)< T, [std::integral_constant](#)< bool, [binding](#)< T >::is_↔ trivial > > Struct Template Reference

```
#include <bind.hpp>
```

Static Public Attributes

- static constexpr bool [value](#) = [binding](#)<T>::is_trivial

6.25.1 Detailed Description

```
template<typename T>
struct upcxx::binding\_is\_trivial< T, std::integral\_constant< bool, binding< T >::is_trivial > >
```

Definition at line 54 of file bind.hpp.

6.25.2 Member Data Documentation

6.25.2.1 value

```
template<typename T >
constexpr bool upcxx::binding\_is\_trivial< T, std::integral\_constant< bool, binding< T >::is_↔  
_trivial > >::value = binding<T>::is_trivial [static]
```

Definition at line 55 of file bind.hpp.

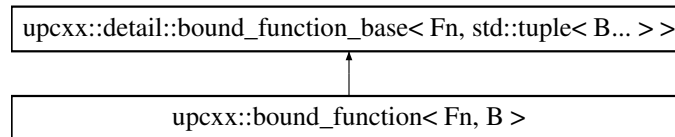
The documentation for this struct was generated from the following file:

- [bind.hpp](#)

6.26 upcxx::bound_function< Fn, B > Struct Template Reference

```
#include <bind.hpp>
```

Inheritance diagram for upcxx::bound_function< Fn, B >:



Public Types

- using `base_type` = `detail::bound_function_base< Fn, std::tuple< B... > >`

Public Member Functions

- `bound_function` (typename `binding`< Fn >::on_wire_type &&fn, std::tuple< typename `binding`< B >::on_wire_type... > &&b)
- template<typename ... Arg>
auto `operator()` (Arg &&...a) const -> decltype(base_type::apply_(*this, upcxx::make_index_sequence< sizeof...(B)>(), std::forward< Arg >(a)...))
- template<typename ... Arg>
auto `operator()` (Arg &&...a) -> decltype(base_type::apply_(*this, upcxx::make_index_sequence< sizeof...(B)>(), std::forward< Arg >(a)...))

6.26.1 Detailed Description

```
template<typename Fn, typename ... B>
struct upcxx::bound_function< Fn, B >
```

Definition at line 179 of file bind.hpp.

6.26.2 Member Typedef Documentation

6.26.2.1 base_type

```
template<typename Fn, typename ... B>
using upcxx::bound_function< Fn, B >::base_type = detail::bound_function_base<Fn, std::tuple<B...> >
```

Definition at line 182 of file bind.hpp.

6.26.3 Constructor & Destructor Documentation

6.26.3.1 bound_function()

```
template<typename Fn, typename ... B>
upcxx::bound_function< Fn, B >::bound_function (
    typename binding< Fn >::on_wire_type && fn,
    std::tuple< typename binding< B >::on_wire_type... > && b ) [inline]
```

Definition at line 184 of file bind.hpp.

6.26.4 Member Function Documentation

6.26.4.1 operator>() [1/2]

```
template<typename Fn, typename ... B>
template<typename ... Arg>
auto upcxx::bound_function< Fn, B >::operator() (
    Arg &&... a ) const -> decltype( base_type::apply_(*this, upcxx::make_index_↵
sequence<sizeof...(B)>(), std::forward<Arg>(a)... ) ) [inline]
```

Definition at line 192 of file bind.hpp.

6.26.4.2 operator>() [2/2]

```
template<typename Fn, typename ... B>
template<typename ... Arg>
auto upcxx::bound_function< Fn, B >::operator() (
    Arg &&... a ) -> decltype( base_type::apply_(*this, upcxx::make_index_sequence<sizeof...(B)>(),
std::forward<Arg>(a)... ) ) [inline]
```

Definition at line 206 of file bind.hpp.

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

- [bind.hpp](#)

6.27 upcxx::detail::bound_function_base< Fn, BndTup, all_trivial > Struct Template Reference

```
#include <bind.hpp>
```

6.27.1 Detailed Description

```
template<typename Fn, typename BndTup, bool all_trivial = binding_is_trivial<Fn>::value && upcxx::trait_forall_↵
tupled<binding_is_trivial, BndTup>::value>
struct upcxx::detail::bound_function_base< Fn, BndTup, all_trivial >
```

Definition at line 90 of file `bind.hpp`.

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

- [bind.hpp](#)

6.28 `upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >` Struct Template Reference

```
#include <bind.hpp>
```

Classes

- struct [applicator](#)

Static Public Member Functions

- `template<typename Me, int ... bi, typename ... Arg>`
`static auto apply_(Me &&me, upcxx::index_sequence< bi... > b_seq, Arg &&...a) -> decltype(upcxx::when_all(binding< Fn >::off_wire(me.fn_), binding< B >::off_wire(std::get< bi >(const_cast< std::tuple< typename binding< B >::on_wire_type... > &>(me.b_))...)...) >> std::declval< applicator< Arg &&... >>())`

Public Attributes

- `binding< Fn >::on_wire_type fn_`
- `std::tuple< typename binding< B >::on_wire_type... > b_`

6.28.1 Detailed Description

```
template<typename Fn, typename ... B>
struct upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >
```

Definition at line 120 of file `bind.hpp`.

6.28.2 Member Function Documentation

6.28.2.1 apply_()

```
template<typename Fn , typename ... B>
template<typename Me , int ... bi, typename ... Arg>
static auto upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::apply_ (
    Me && me,
    upcxx::index_sequence< bi... > b_seq,
    Arg &&... a ) -> decltype( upcxx::when_all( binding<Fn>::off_wire(me.fn_),
binding<B>::off_wire( std::get<bi>(const_cast<std::tuple<typename binding<B>::on_wire_↔
type...&>(me.b_)) )... ) >> std::declval<applicator<Arg&&...>>() ) [inline], [static]
```

Definition at line 153 of file bind.hpp.

6.28.3 Member Data Documentation

6.28.3.1 b_

```
template<typename Fn , typename ... B>
std::tuple<typename binding<B>::on_wire_type...> upcxx::detail::bound_function_base< Fn,
std::tuple< B... >, false >::b_
```

Definition at line 125 of file bind.hpp.

6.28.3.2 fn_

```
template<typename Fn , typename ... B>
binding<Fn>::on_wire_type upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false
>::fn_
```

Definition at line 124 of file bind.hpp.

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

- [bind.hpp](#)

6.29 upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true > Struct Template Reference

```
#include <bind.hpp>
```

Static Public Member Functions

- template<typename Me , int ... bi, typename ... Arg>
static auto apply_ (Me &&me, upcxx::index_sequence< bi... > b_seq, Arg &&...a) -> decltype(me.fn_↔
(std::get< bi >(const_cast< std::tuple< typename binding< B >::on_wire_type... > &>(me.b_))...), std↔
::forward< Arg >(a)...))

Public Attributes

- `binding`< Fn >::on_wire_type `fn_`
- `std::tuple`< typename `binding`< B >::on_wire_type... > `b_`

6.29.1 Detailed Description

```
template<typename Fn, typename ... B>
struct upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true >
```

Definition at line 93 of file `bind.hpp`.

6.29.2 Member Function Documentation

6.29.2.1 `apply_()`

```
template<typename Fn , typename ... B>
template<typename Me , int ... bi, typename ... Arg>
static auto upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true >::apply_ (
    Me && me,
    upcxx::index_sequence< bi... > b_seq,
    Arg &&... a ) -> decltype( me.fn_( std::get<bi>(const_cast<std::tuple<typename
binding<B>::on_wire_type...>&(me.b_))..., std::forward<Arg>(a)... ) ) ) [inline], [static]
```

Definition at line 101 of file `bind.hpp`.

6.29.3 Member Data Documentation

6.29.3.1 `b_`

```
template<typename Fn , typename ... B>
std::tuple<typename binding<B>::on_wire_type...> upcxx::detail::bound_function_base< Fn,
std::tuple< B... >, true >::b_
```

Definition at line 98 of file `bind.hpp`.

6.29.3.2 fn_

```
template<typename Fn , typename ... B>
binding<Fn>::on_wire_type upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true
>::fn_
```

Definition at line 97 of file bind.hpp.

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

- [bind.hpp](#)

6.30 upcxx::commanding< Fn > Struct Template Reference

```
#include <command.hpp>
```

Static Public Member Functions

- static void [size_ubound](#) ([parcel_layout](#) &ub, Fn const &x)
- static void [pack](#) ([parcel_writer](#) &w, Fn const &x)
- static auto [execute](#) ([parcel_reader](#) &r) -> decltype([make_future](#)())

6.30.1 Detailed Description

```
template<typename Fn>
struct upcxx::commanding< Fn >
```

Definition at line 35 of file command.hpp.

6.30.2 Member Function Documentation

6.30.2.1 execute()

```
template<typename Fn >
static auto upcxx::commanding< Fn >::execute (
    parcel\_reader & r ) -> decltype(make\_future()) [inline], [static]
```

Definition at line 44 of file command.hpp.

6.30.2.2 pack()

```
template<typename Fn >
static void upcxx::commanding< Fn >::pack (
    parcel_writer & w,
    Fn const & x ) [inline], [static]
```

Definition at line 40 of file command.hpp.

6.30.2.3 size_ubound()

```
template<typename Fn >
static void upcxx::commanding< Fn >::size_ubound (
    parcel_layout & ub,
    Fn const & x ) [inline], [static]
```

Definition at line 36 of file command.hpp.

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

- [command.hpp](#)

6.31 upcxx::completions< SourceCx, RemoteCx, OpxnCx > Struct Template Reference

```
#include <completion.hpp>
```

Public Attributes

- SourceCx [source](#)
- RemoteCx [remote](#)
- OpxnCx [opern](#)

Static Public Attributes

- static constexpr int [future_n](#)

6.31.1 Detailed Description

```
template<typename SourceCx, typename RemoteCx, typename OpxnCx>
struct upcxx::completions< SourceCx, RemoteCx, OpxnCx >
```

Definition at line 72 of file completion.hpp.

6.31.2 Member Data Documentation

6.31.2.1 future_n

```
template<typename SourceCx, typename RemoteCx, typename OpxnCx>
constexpr int upcxx::completions< SourceCx, RemoteCx, OpxnCx >::future_n [static]
```

Initial value:

```
=
    (detail::is_future_cx<SourceCx>::value ? 1 : 0) +
    (detail::is_future_cx<OpxnCx>::value ? 1 : 0)
```

Definition at line 77 of file completion.hpp.

6.31.2.2 operxn

```
template<typename SourceCx, typename RemoteCx, typename OpxnCx>
OpxnCx upcxx::completions< SourceCx, RemoteCx, OpxnCx >::operxn
```

Definition at line 75 of file completion.hpp.

6.31.2.3 remote

```
template<typename SourceCx, typename RemoteCx, typename OpxnCx>
RemoteCx upcxx::completions< SourceCx, RemoteCx, OpxnCx >::remote
```

Definition at line 74 of file completion.hpp.

6.31.2.4 source

```
template<typename SourceCx, typename RemoteCx, typename OpxnCx>
SourceCx upcxx::completions< SourceCx, RemoteCx, OpxnCx >::source
```

Definition at line 73 of file completion.hpp.

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

- [completion.hpp](#)

6.32 upcxx::constant_function< T > Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- [constant_function](#) (T value)
- [template<typename ... Arg>
T operator\(\)](#) (Arg &&...args) const

Public Attributes

- [T value_](#)

6.32.1 Detailed Description

```
template<typename T>  
struct upcxx::constant_function< T >
```

Definition at line 44 of file utility.hpp.

6.32.2 Constructor & Destructor Documentation

6.32.2.1 constant_function()

```
template<typename T >  
upcxx::constant_function< T >::constant_function (  
    T value ) [inline]
```

Definition at line 46 of file utility.hpp.

6.32.3 Member Function Documentation

6.32.3.1 operator()

```
template<typename T >  
template<typename ... Arg>  
T upcxx::constant_function< T >::operator() (  
    Arg &&... args ) const [inline]
```

Definition at line 49 of file utility.hpp.

6.32.4 Member Data Documentation

6.32.4.1 value_

```
template<typename T >
T upcxx::constant_function< T >::value_
```

Definition at line 45 of file utility.hpp.

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

- [utility.hpp](#)

6.33 upcxx::decay_tupled< Tup > Struct Template Reference

```
#include <utility.hpp>
```

6.33.1 Detailed Description

```
template<typename Tup>
struct upcxx::decay_tupled< Tup >
```

Definition at line 221 of file utility.hpp.

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

- [utility.hpp](#)

6.34 upcxx::decay_tupled< std::tuple< T... > > Struct Template Reference

```
#include <utility.hpp>
```

Public Types

- typedef std::tuple< typename std::decay< T >::type... > [type](#)

6.34.1 Detailed Description

```
template<typename ... T>
struct upcxx::decay_tupled< std::tuple< T... > >
```

Definition at line 223 of file utility.hpp.

6.34.2 Member Typedef Documentation

6.34.2.1 type

```
template<typename ... T>
typedef std::tuple<typename std::decay<T>::type...> upcxx::decay_tupled< std::tuple< T... >
>::type
```

Definition at line 224 of file utility.hpp.

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

- [utility.hpp](#)

6.35 upcxx::detail::future_header::dependency_link Struct Reference

```
#include <core.hpp>
```

Public Member Functions

- void [unlink](#) ()

Public Attributes

- [future_header](#) * [dep](#)
- [future_header_dependent](#) * [suc](#)
- [dependency_link](#) * [sucs_next](#)

6.35.1 Detailed Description

Definition at line 187 of file core.hpp.

6.35.2 Member Function Documentation

6.35.2.1 unlink()

```
void future_header::dependency_link::unlink ( )
```

Definition at line 31 of file core.cpp.

6.35.3 Member Data Documentation

6.35.3.1 dep

[future_header*](#) upcxx::detail::future_header::dependency_link::dep

Definition at line 189 of file core.hpp.

6.35.3.2 suc

[future_header_dependent*](#) upcxx::detail::future_header::dependency_link::suc

Definition at line 192 of file core.hpp.

6.35.3.3 sucs_next

[dependency_link*](#) upcxx::detail::future_header::dependency_link::sucs_next

Definition at line 195 of file core.hpp.

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

- [future/core.hpp](#)
- [future/core.cpp](#)

6.36 upcxx::digest Struct Reference

```
#include <digest.hpp>
```

Public Member Functions

- [digest eat](#) (std::uint64_t x0, std::uint64_t x1=0) const
- [digest eat](#) ([digest](#) that) const

Static Public Member Functions

- static constexpr [digest zero](#) ()

Public Attributes

- `std::uint64_t w0`
- `std::uint64_t w1`

Friends

- `bool operator== (digest a, digest b)`
- `bool operator!= (digest a, digest b)`
- `bool operator< (digest a, digest b)`
- `bool operator<= (digest a, digest b)`
- `bool operator> (digest a, digest b)`
- `bool operator>= (digest a, digest b)`

6.36.1 Detailed Description

Definition at line 9 of file `digest.hpp`.

6.36.2 Member Function Documentation

6.36.2.1 `eat()` [1/2]

```
digest upcxx::digest::eat (  
    std::uint64_t x0,  
    std::uint64_t x1 = 0 ) const
```

6.36.2.2 `eat()` [2/2]

```
digest upcxx::digest::eat (  
    digest that ) const [inline]
```

Definition at line 18 of file `digest.hpp`.

6.36.2.3 `zero()`

```
static constexpr digest upcxx::digest::zero ( ) [inline], [static]
```

Definition at line 12 of file `digest.hpp`.

6.36.3 Friends And Related Function Documentation

6.36.3.1 operator!=

```
bool operator!= (
    digest a,
    digest b ) [friend]
```

Definition at line 23 of file digest.hpp.

6.36.3.2 operator<

```
bool operator< (
    digest a,
    digest b ) [friend]
```

Definition at line 25 of file digest.hpp.

6.36.3.3 operator<=

```
bool operator<= (
    digest a,
    digest b ) [friend]
```

Definition at line 26 of file digest.hpp.

6.36.3.4 operator==

```
bool operator== (
    digest a,
    digest b ) [friend]
```

Definition at line 22 of file digest.hpp.

6.36.3.5 operator>

```
bool operator> (
    digest a,
    digest b ) [friend]
```

Definition at line 27 of file digest.hpp.

6.36.3.6 operator>=

```
bool operator>= (
    digest a,
    digest b ) [friend]
```

Definition at line 28 of file digest.hpp.

6.36.4 Member Data Documentation

6.36.4.1 w0

```
std::uint64_t upcxx::digest::w0
```

Definition at line 10 of file digest.hpp.

6.36.4.2 w1

```
std::uint64_t upcxx::digest::w1
```

Definition at line 10 of file digest.hpp.

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

- [digest.hpp](#)

6.37 upcxx::detail::disjoin_cx< A, B, B_ord_bump > Struct Template Reference

```
#include <completion.hpp>
```

6.37.1 Detailed Description

```
template<typename A, typename B, int B_ord_bump>
struct upcxx::detail::disjoin_cx< A, B, B_ord_bump >
```

Definition at line 28 of file completion.hpp.

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

- [completion.hpp](#)

6.38 `upcxx::detail::disjoin_cx< A, nil_cx, B_ord_bump >` Struct Template Reference

```
#include <completion.hpp>
```

Public Types

- using `type` = A

Public Member Functions

- A && `operator()` (A &&a, `nil_cx` &&b)

6.38.1 Detailed Description

```
template<typename A, int B_ord_bump>
struct upcxx::detail::disjoin_cx< A, nil_cx, B_ord_bump >
```

Definition at line 31 of file `completion.hpp`.

6.38.2 Member Typedef Documentation

6.38.2.1 `type`

```
template<typename A , int B_ord_bump>
using upcxx::detail::disjoin_cx< A, nil_cx, B_ord_bump >::type = A
```

Definition at line 32 of file `completion.hpp`.

6.38.3 Member Function Documentation

6.38.3.1 `operator()`

```
template<typename A , int B_ord_bump>
A&& upcxx::detail::disjoin_cx< A, nil_cx, B_ord_bump >::operator() (
    A && a,
    nil_cx && b ) [inline]
```

Definition at line 34 of file `completion.hpp`.

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

- [completion.hpp](#)

6.39 `upcxx::detail::disjoin_cx< nil_cx, B, B_ord_bump >` Struct Template Reference

```
#include <completion.hpp>
```

Public Types

- using `type` = B

Public Member Functions

- B && `operator()` (`nil_cx` &&a, B &&b)

6.39.1 Detailed Description

```
template<typename B, int B_ord_bump>  
struct upcxx::detail::disjoin_cx< nil_cx, B, B_ord_bump >
```

Definition at line 39 of file `completion.hpp`.

6.39.2 Member Typedef Documentation

6.39.2.1 `type`

```
template<typename B , int B_ord_bump>  
using upcxx::detail::disjoin_cx< nil_cx, B, B_ord_bump >::type = B
```

Definition at line 40 of file `completion.hpp`.

6.39.3 Member Function Documentation

6.39.3.1 `operator()`

```
template<typename B , int B_ord_bump>  
B&& upcxx::detail::disjoin_cx< nil_cx, B, B_ord_bump >::operator() (  
    nil_cx && a,  
    B && b ) [inline]
```

Definition at line 42 of file `completion.hpp`.

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

- [completion.hpp](#)

6.40 `upcxx::detail::disjoin_cx< nil_cx, future_cx< B_ord_old >, B_ord_bump >` Struct Template Reference

```
#include <completion.hpp>
```

Public Types

- using `type` = `future_cx< B_ord_old+B_ord_bump >`

Public Member Functions

- `future_cx< B_ord_old+B_ord_bump > operator() (nil_cx &&a, future_cx< B_ord_old > &&b)`

6.40.1 Detailed Description

```
template<int B_ord_old, int B_ord_bump>
struct upcxx::detail::disjoin_cx< nil_cx, future_cx< B_ord_old >, B_ord_bump >
```

Definition at line 47 of file completion.hpp.

6.40.2 Member Typedef Documentation

6.40.2.1 `type`

```
template<int B_ord_old, int B_ord_bump>
using upcxx::detail::disjoin_cx< nil_cx, future_cx< B_ord_old >, B_ord_bump >::type = future_cx<B_ord_old + B_ord_bump>
```

Definition at line 48 of file completion.hpp.

6.40.3 Member Function Documentation

6.40.3.1 `operator()()`

```
template<int B_ord_old, int B_ord_bump>
future_cx<B_ord_old + B_ord_bump> upcxx::detail::disjoin_cx< nil_cx, future_cx< B_ord_old >,
B_ord_bump >::operator() (
    nil_cx && a,
    future_cx< B_ord_old > && b ) [inline]
```

Definition at line 50 of file completion.hpp.

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

- [completion.hpp](#)

6.41 `upcxx::detail::disjoin_cx< nil_cx, nil_cx, B_ord0 >` Struct Template Reference

```
#include <completion.hpp>
```

Public Types

- using `type` = `nil_cx`

Public Member Functions

- `nil_cx operator()` (`nil_cx &&a`, `nil_cx &&b`)

6.41.1 Detailed Description

```
template<int B_ord0>
struct upcxx::detail::disjoin_cx< nil_cx, nil_cx, B_ord0 >
```

Definition at line 55 of file `completion.hpp`.

6.41.2 Member Typedef Documentation

6.41.2.1 `type`

```
template<int B_ord0>
using upcxx::detail::disjoin_cx< nil_cx, nil_cx, B_ord0 >::type = nil_cx
```

Definition at line 56 of file `completion.hpp`.

6.41.3 Member Function Documentation

6.41.3.1 `operator()`

```
template<int B_ord0>
nil_cx upcxx::detail::disjoin_cx< nil_cx, nil_cx, B_ord0 >::operator() (
    nil_cx && a,
    nil_cx && b ) [inline]
```

Definition at line 58 of file `completion.hpp`.

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

- [completion.hpp](#)

6.42 upcxx::dist_id< T > Struct Template Reference

```
#include <dist_object.hpp>
```

Public Member Functions

- [dist_object](#)< T > & [here](#) () const
- [future](#)< [dist_object](#)< T > & > [when_here](#) () const

Public Attributes

- [digest](#) [dig_](#)

6.42.1 Detailed Description

```
template<typename T>  
struct upcxx::dist_id< T >
```

Definition at line 15 of file `dist_object.hpp`.

6.42.2 Member Function Documentation

6.42.2.1 here()

```
template<typename T>  
dist\_object<T>& upcxx::dist\_id< T >::here ( ) const [inline]
```

Definition at line 56 of file `dist_object.hpp`.

6.42.2.2 when_here()

```
template<typename T>  
future<dist\_object<T>&& > upcxx::dist\_id< T >::when\_here ( ) const [inline]
```

Definition at line 60 of file `dist_object.hpp`.

6.42.3 Member Data Documentation

6.42.3.1 `dig_`

```
template<typename T>
digest upcxx::dist_id< T >::dig_
```

Definition at line 53 of file `dist_object.hpp`.

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

- [dist_object.hpp](#)

6.43 `upcxx::dist_object< T >` Class Template Reference

```
#include <dist_object.hpp>
```

Public Member Functions

- [dist_object](#) (T value)
- [dist_object](#) ([dist_object](#) const &)=delete
- [dist_object](#) ([dist_object](#) &&that)
- [~dist_object](#) ()
- T * [operator->](#) () const
- T & [operator*](#) () const
- [dist_id](#)< T > [id](#) () const
- [future](#)< T > [fetch](#) (inrank_t rank)

6.43.1 Detailed Description

```
template<typename T>
class upcxx::dist_object< T >
```

Definition at line 18 of file `dist_object.hpp`.

6.43.2 Constructor & Destructor Documentation

6.43.2.1 `dist_object()` [1/3]

```
template<typename T>
upcxx::dist_object< T >::dist_object (
    T value ) [inline]
```

Definition at line 101 of file `dist_object.hpp`.

6.43.2.2 `dist_object()` [2/3]

```
template<typename T>
upcxx::dist_object< T >::dist_object (
    dist_object< T > const & ) [delete]
```

6.43.2.3 `dist_object()` [3/3]

```
template<typename T>
upcxx::dist_object< T >::dist_object (
    dist_object< T > && that ) [inline]
```

Definition at line 113 of file `dist_object.hpp`.

6.43.2.4 `~dist_object()`

```
template<typename T>
upcxx::dist_object< T >::~dist_object ( ) [inline]
```

Definition at line 129 of file `dist_object.hpp`.

6.43.3 Member Function Documentation

6.43.3.1 `fetch()`

```
template<typename T>
future<T> upcxx::dist_object< T >::fetch (
    intrank_t rank ) [inline]
```

Definition at line 142 of file `dist_object.hpp`.

6.43.3.2 `id()`

```
template<typename T>
dist_id<T> upcxx::dist_object< T >::id ( ) const [inline]
```

Definition at line 140 of file `dist_object.hpp`.

6.43.3.3 operator*()

```
template<typename T>
T& upcxx::dist_object< T >::operator* ( ) const [inline]
```

Definition at line 138 of file dist_object.hpp.

6.43.3.4 operator->()

```
template<typename T>
T* upcxx::dist_object< T >::operator-> ( ) const [inline]
```

Definition at line 137 of file dist_object.hpp.

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

- [dist_object.hpp](#)

6.44 upcxx::fast_hasher Struct Reference

```
#include <reflection.hpp>
```

Public Member Functions

- [fast_hasher](#) (std::size_t salt)
- void [operator\(\)](#) (std::size_t x)
- std::size_t [result](#) () const

Public Attributes

- std::size_t [s](#)

6.44.1 Detailed Description

Definition at line 61 of file reflection.hpp.

6.44.2 Constructor & Destructor Documentation

6.44.2.1 fast_hasher()

```
upcxx::fast_hasher::fast_hasher (
    std::size_t salt ) [inline]
```

Definition at line 64 of file reflection.hpp.

6.44.3 Member Function Documentation

6.44.3.1 operator>()

```
void upcxx::fast_hasher::operator() (
    std::size_t x ) [inline]
```

Utilities derived from Boost, subject to the following license:

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Definition at line 68 of file reflection.hpp.

6.44.3.2 result()

```
std::size_t upcxx::fast_hasher::result ( ) const [inline]
```

Definition at line 98 of file reflection.hpp.

6.44.4 Member Data Documentation

6.44.4.1 s

```
std::size_t upcxx::fast_hasher::s
```

Definition at line 62 of file reflection.hpp.

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

- [reflection.hpp](#)

6.45 upcxx::fast_hashing< T > Struct Template Reference

```
#include <reflection.hpp>
```

Public Member Functions

- `std::size_t operator() (const T &x, std::size_t salt=0) const`

6.45.1 Detailed Description

```
template<typename T>  
struct upcxx::fast_hashing< T >
```

Definition at line 123 of file reflection.hpp.

6.45.2 Member Function Documentation

6.45.2.1 operator()()

```
template<typename T >  
std::size_t upcxx::fast_hashing< T >::operator() (  
    const T & x,  
    std::size_t salt = 0 ) const [inline]
```

Definition at line 124 of file reflection.hpp.

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

- [reflection.hpp](#)

6.46 upcxx::function_ref< Sig > Class Template Reference

```
#include <utility.hpp>
```

6.46.1 Detailed Description

```
template<typename Sig>
class upcxx::function_ref< Sig >
```

Definition at line 74 of file utility.hpp.

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

- [utility.hpp](#)

6.47 upcxx::function_ref< Ret(Arg...)> Class Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- [function_ref](#) ()
- [function_ref](#) (Fn &&fn)
- [function_ref](#) (const [function_ref](#) &)=default
- [function_ref](#) & [operator=](#) (const [function_ref](#) &)=default
- [function_ref](#) ([function_ref](#) &&)=default
- [function_ref](#) & [operator=](#) ([function_ref](#) &&)=default
- [Ret operator\(\)](#) (Arg ...arg) const

6.47.1 Detailed Description

```
template<typename Ret, typename ... Arg>
class upcxx::function_ref< Ret(Arg...)>
```

Definition at line 77 of file utility.hpp.

6.47.2 Constructor & Destructor Documentation

6.47.2.1 [function_ref\(\)](#) [1/4]

```
template<typename Ret , typename ... Arg>
upcxx::function_ref< Ret (Arg...)>::function_ref ( ) [inline]
```

Definition at line 92 of file utility.hpp.

6.47.2.2 function_ref() [2/4]

```
template<typename Ret , typename ... Arg>
template<typename Fn >
upcxx::function_ref< Ret (Arg...)>::function_ref (
    Fn && fn ) [inline]
```

Definition at line 97 of file utility.hpp.

6.47.2.3 function_ref() [3/4]

```
template<typename Ret , typename ... Arg>
upcxx::function_ref< Ret (Arg...)>::function_ref (
    const function_ref< Ret (Arg...)> & ) [default]
```

6.47.2.4 function_ref() [4/4]

```
template<typename Ret , typename ... Arg>
upcxx::function_ref< Ret (Arg...)>::function_ref (
    function_ref< Ret (Arg...)> && ) [default]
```

6.47.3 Member Function Documentation**6.47.3.1** operator>()

```
template<typename Ret , typename ... Arg>
Ret upcxx::function_ref< Ret (Arg...)>::operator() (
    Arg ... arg ) const [inline]
```

Definition at line 106 of file utility.hpp.

6.47.3.2 operator=() [1/2]

```
template<typename Ret , typename ... Arg>
function_ref& upcxx::function_ref< Ret (Arg...)>::operator= (
    const function_ref< Ret (Arg...)> & ) [default]
```

6.47.3.3 operator=() [2/2]

```
template<typename Ret , typename ... Arg>
function_ref& upcxx::function_ref< Ret (Arg...)>::operator= (
    function_ref< Ret (Arg...)> && ) [default]
```

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

- [utility.hpp](#)

6.48 upcxx::future1< Kind, T > Struct Template Reference

```
#include <core.hpp>
```

Public Types

- typedef Kind [kind_type](#)
- typedef std::tuple< T... > [results_type](#)
- typedef Kind::template with_types< T... > [impl_type](#)

Public Member Functions

- [future1](#) ()=default
- [~future1](#) ()=default
- [future1](#) ([impl_type](#) impl)
- template<typename impl_type1 > [future1](#) (impl_type1 impl)
- [future1](#) ([future1](#) const &)=default
- template<typename Kind1 > [future1](#) ([future1](#)< Kind1, T... > const &that)
- [future1](#) ([future1](#) &&)=default
- template<typename Kind1 > [future1](#) ([future1](#)< Kind1, T... > &&that)
- [future1](#) & [operator=](#) ([future1](#) const &)=default
- template<typename Kind1 > [future1](#) & [operator=](#) ([future1](#)< Kind1, T... > const &that)
- [future1](#) & [operator=](#) ([future1](#) &&)=default
- template<typename Kind1 > [future1](#) & [operator=](#) ([future1](#)< Kind1, T... > &&that)
- bool [ready](#) () const
- template<int i = 0> upcxx::tuple_element_or_void< i, [results_type](#) >::type [result](#) () const
- [results_type](#) [results](#) () const
- template<int i = 0> auto [result_moved](#) () -> decltype(upcxx::get_or_void< i >(impl_.template result_rvals()))
- auto [results_moved](#) () -> decltype(impl_.template result_rvals())
- template<typename Fn > auto [then](#) (Fn &&fn) -> decltype(detail::future_then< [future1](#)< Kind, T... >, typename std::decay< Fn >::type >)(*this, std::forward< Fn >(fn))
- template<typename Fn > auto [then_pure](#) (Fn &&pure_fn) -> decltype(detail::future_then_pure< [future1](#)< Kind, T... >, typename std::decay< Fn >::type >)(*this, std::forward< Fn >(pure_fn))

Public Attributes

- [impl_type impl_](#)

6.48.1 Detailed Description

```
template<typename Kind, typename ... T>
struct upcxx::future1< Kind, T >
```

Definition at line 74 of file `core.hpp`.

6.48.2 Member Typedef Documentation

6.48.2.1 `impl_type`

```
template<typename Kind, typename ... T>
typedef Kind::template with_types<T...> upcxx::future1< Kind, T >::impl\_type
```

Definition at line 53 of file `future1.hpp`.

6.48.2.2 `kind_type`

```
template<typename Kind, typename ... T>
typedef Kind upcxx::future1< Kind, T >::kind\_type
```

Definition at line 51 of file `future1.hpp`.

6.48.2.3 `results_type`

```
template<typename Kind, typename ... T>
typedef std::tuple<T...> upcxx::future1< Kind, T >::results\_type
```

Definition at line 52 of file `future1.hpp`.

6.48.3 Constructor & Destructor Documentation

6.48.3.1 future1() [1/7]

```
template<typename Kind, typename ... T>
upcxx::future1< Kind, T >::future1 ( ) [default]
```

6.48.3.2 ~future1()

```
template<typename Kind, typename ... T>
upcxx::future1< Kind, T >::~~future1 ( ) [default]
```

6.48.3.3 future1() [2/7]

```
template<typename Kind, typename ... T>
upcxx::future1< Kind, T >::future1 (
    impl_type impl ) [inline]
```

Definition at line 61 of file future1.hpp.

6.48.3.4 future1() [3/7]

```
template<typename Kind, typename ... T>
template<typename impl_type1 >
upcxx::future1< Kind, T >::future1 (
    impl_type1 impl ) [inline]
```

Definition at line 63 of file future1.hpp.

6.48.3.5 future1() [4/7]

```
template<typename Kind, typename ... T>
upcxx::future1< Kind, T >::future1 (
    future1< Kind, T > const & ) [default]
```

6.48.3.6 future1() [5/7]

```
template<typename Kind, typename ... T>
template<typename Kind1 >
upcxx::future1< Kind, T >::future1 (
    future1< Kind1, T... > const & that ) [inline]
```

Definition at line 67 of file future1.hpp.

6.48.3.7 `future1()` [6/7]

```
template<typename Kind, typename ... T>
upcxx::future1< Kind, T >::future1 (
    future1< Kind, T > && ) [default]
```

6.48.3.8 `future1()` [7/7]

```
template<typename Kind, typename ... T>
template<typename Kind1 >
upcxx::future1< Kind, T >::future1 (
    future1< Kind1, T... > && that ) [inline]
```

Definition at line 71 of file `future1.hpp`.

6.48.4 Member Function Documentation

6.48.4.1 `operator=()` [1/4]

```
template<typename Kind, typename ... T>
future1& upcxx::future1< Kind, T >::operator= (
    future1< Kind, T > const & ) [default]
```

6.48.4.2 `operator=()` [2/4]

```
template<typename Kind, typename ... T>
template<typename Kind1 >
future1& upcxx::future1< Kind, T >::operator= (
    future1< Kind1, T... > const & that ) [inline]
```

Definition at line 75 of file `future1.hpp`.

6.48.4.3 `operator=()` [3/4]

```
template<typename Kind, typename ... T>
future1& upcxx::future1< Kind, T >::operator= (
    future1< Kind, T > && ) [default]
```

6.48.4.4 operator=() [4/4]

```
template<typename Kind, typename ... T>
template<typename Kind1 >
future1& upcxx::future1< Kind, T >::operator= (
    future1< Kind1, T... > && that ) [inline]
```

Definition at line 82 of file future1.hpp.

6.48.4.5 ready()

```
template<typename Kind, typename ... T>
bool upcxx::future1< Kind, T >::ready ( ) const [inline]
```

Definition at line 87 of file future1.hpp.

6.48.4.6 result()

```
template<typename Kind, typename ... T>
template<int i = 0>
upcxx::tuple_element_or_void<i, results_type>::type upcxx::future1< Kind, T >::result ( )
const [inline]
```

Definition at line 92 of file future1.hpp.

6.48.4.7 result_moved()

```
template<typename Kind, typename ... T>
template<int i = 0>
auto upcxx::future1< Kind, T >::result_moved ( ) -> decltype(upcxx::get_or_void<i>(impl_.←
template result_rvals())) [inline]
```

Definition at line 102 of file future1.hpp.

6.48.4.8 results()

```
template<typename Kind, typename ... T>
results_type upcxx::future1< Kind, T >::results ( ) const [inline]
```

Definition at line 97 of file future1.hpp.

6.48.4.9 results_moved()

```
template<typename Kind, typename ... T>
auto upcxx::future1< Kind, T >::results_moved ( ) -> decltype(impl_.template result_rvals())
[inline]
```

Definition at line 107 of file future1.hpp.

6.48.4.10 then()

```
template<typename Kind, typename ... T>
template<typename Fn >
auto upcxx::future1< Kind, T >::then (
    Fn && fn ) -> decltype( detail::future_then<future1<Kind,T...>, typename std::
::decay<Fn>::type>() ( *this, std::forward<Fn>(fn) ) ) [inline]
```

Definition at line 113 of file future1.hpp.

6.48.4.11 then_pure()

```
template<typename Kind, typename ... T>
template<typename Fn >
auto upcxx::future1< Kind, T >::then_pure (
    Fn && pure_fn ) -> decltype( detail::future_then_pure<future1<Kind,T...>, typename
std::decay<Fn>::type>() ( *this, std::forward<Fn>(pure_fn) ) ) [inline]
```

Definition at line 127 of file future1.hpp.

6.48.5 Member Data Documentation

6.48.5.1 impl_

```
template<typename Kind, typename ... T>
impl_type upcxx::future1< Kind, T >::impl_
```

Definition at line 55 of file future1.hpp.

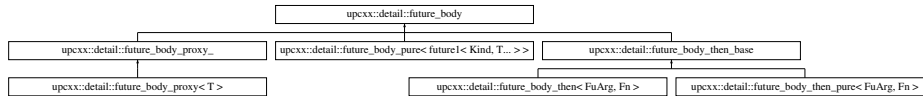
The documentation for this struct was generated from the following files:

- [future/core.hpp](#)
- [future/future1.hpp](#)

6.49 upcxx::detail::future_body Struct Reference

```
#include <core.hpp>
```

Inheritance diagram for upcxx::detail::future_body:



Public Member Functions

- [future_body](#) (void *storage)
- virtual void [destruct_early](#) ()
- virtual void [leave_active](#) ([future_header_dependent](#) *owner_hdr)=0

Public Attributes

- void * [storage_](#)

6.49.1 Detailed Description

Definition at line 329 of file core.hpp.

6.49.2 Constructor & Destructor Documentation

6.49.2.1 future_body()

```
upcxx::detail::future_body::future_body (
    void * storage ) [inline]
```

Definition at line 333 of file core.hpp.

6.49.3 Member Function Documentation

6.49.3.1 destruct_early()

```
void future_body::destruct_early ( ) [virtual]
```

Reimplemented in [upcxx::detail::future_body_proxy< T >](#), [upcxx::detail::future_body_then_pure< FuArg, Fn >](#), and [upcxx::detail::future_body_pure< future1< Kind, T... > >](#).

Definition at line 270 of file core.cpp.

6.49.3.2 leave_active()

```
virtual void upcxx::detail::future_body::leave_active (
    future_header_dependent * owner_hdr ) [pure virtual]
```

Implemented in [upcxx::detail::future_body_proxy_](#), [upcxx::detail::future_body_then_pure< FuArg, Fn >](#), [upcxx::detail::future_body_then< FuArg, Fn >](#), and [upcxx::detail::future_body_pure< future1< Kind, T... > >](#).

6.49.4 Member Data Documentation

6.49.4.1 storage_

```
void* upcxx::detail::future_body::storage_
```

Definition at line 331 of file core.hpp.

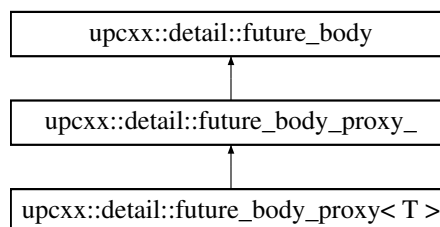
The documentation for this struct was generated from the following files:

- [future/core.hpp](#)
- [future/core.cpp](#)

6.50 upcxx::detail::future_body_proxy< T > Struct Template Reference

```
#include <core.hpp>
```

Inheritance diagram for upcxx::detail::future_body_proxy< T >:



Public Member Functions

- [future_body_proxy](#) (void *storage)
- void [destruct_early](#) ()

Additional Inherited Members

6.50.1 Detailed Description

```
template<typename ... T>
struct upcxx::detail::future_body_proxy< T >
```

Definition at line 23 of file core.hpp.

6.50.2 Constructor & Destructor Documentation

6.50.2.1 future_body_proxy()

```
template<typename ... T>
upcxx::detail::future_body_proxy< T >::future_body_proxy (
    void * storage ) [inline]
```

Definition at line 357 of file core.hpp.

6.50.3 Member Function Documentation

6.50.3.1 destruct_early()

```
template<typename ... T>
void upcxx::detail::future_body_proxy< T >::destruct_early ( ) [inline], [virtual]
```

Reimplemented from [upcxx::detail::future_body](#).

Definition at line 359 of file core.hpp.

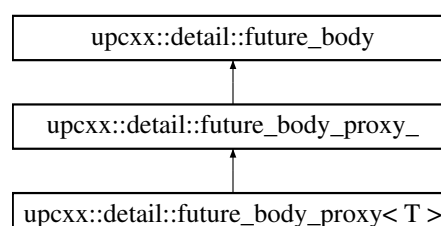
The documentation for this struct was generated from the following file:

- [future/core.hpp](#)

6.51 upcxx::detail::future_body_proxy_ Struct Reference

```
#include <core.hpp>
```

Inheritance diagram for upcxx::detail::future_body_proxy_:



Public Member Functions

- [future_body_proxy_](#) (void *storage)
- void [leave_active](#) ([future_header_dependent](#) *owner_hdr)

Public Attributes

- [future_header::dependency_link](#) link_

6.51.1 Detailed Description

Definition at line 346 of file core.hpp.

6.51.2 Constructor & Destructor Documentation

6.51.2.1 future_body_proxy_()

```
upcxx::detail::future_body_proxy_::future_body_proxy_ (  
    void * storage ) [inline]
```

Definition at line 350 of file core.hpp.

6.51.3 Member Function Documentation

6.51.3.1 leave_active()

```
void future_body_proxy_::leave_active (  
    future\_header\_dependent * owner_hdr ) [virtual]
```

Implements [upcxx::detail::future_body](#).

Definition at line 250 of file core.cpp.

6.51.4 Member Data Documentation

6.51.4.1 link_

`future_header::dependency_link` `upcxx::detail::future_body_proxy_::link_`

Definition at line 348 of file `core.hpp`.

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

- `future/core.hpp`
- `future/core.cpp`

6.52 `upcxx::detail::future_body_pure< FuArg >` Struct Template Reference

```
#include <core.hpp>
```

6.52.1 Detailed Description

```
template<typename FuArg>
struct upcxx::detail::future_body_pure< FuArg >
```

Definition at line 25 of file `core.hpp`.

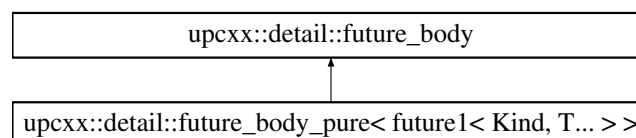
The documentation for this struct was generated from the following file:

- `future/core.hpp`

6.53 `upcxx::detail::future_body_pure< future1< Kind, T... > >` Struct Template Reference

```
#include <body_pure.hpp>
```

Inheritance diagram for `upcxx::detail::future_body_pure< future1< Kind, T... > >`:



Public Member Functions

- `future_body_pure` (void *storage, `future_header_dependent` *suc_hdr, `future1< Kind, T... >` arg)
- void `destruct_early` ()
- void `leave_active` (`future_header_dependent` *hdr)

Public Attributes

- `future_dependency< future1< Kind, T... > >` `dep_`

6.53.1 Detailed Description

```
template<typename Kind, typename ... T>
struct upcxx::detail::future_body_pure< future1< Kind, T... > >
```

Definition at line 12 of file `body_pure.hpp`.

6.53.2 Constructor & Destructor Documentation

6.53.2.1 `future_body_pure()`

```
template<typename Kind , typename ... T>
upcxx::detail::future_body_pure< future1< Kind, T... > >::future_body_pure (
    void * storage,
    future_header_dependent * suc_hdr,
    future1< Kind, T... > arg ) [inline]
```

Definition at line 16 of file `body_pure.hpp`.

6.53.3 Member Function Documentation

6.53.3.1 `destruct_early()`

```
template<typename Kind , typename ... T>
void upcxx::detail::future_body_pure< future1< Kind, T... > >::destruct_early ( ) [inline],
[virtual]
```

Reimplemented from `upcxx::detail::future_body`.

Definition at line 25 of file `body_pure.hpp`.

6.53.3.2 `leave_active()`

```
template<typename Kind , typename ... T>
void upcxx::detail::future_body_pure< future1< Kind, T... > >::leave_active (
    future_header_dependent * hdr ) [inline], [virtual]
```

Implements `upcxx::detail::future_body`.

Definition at line 30 of file `body_pure.hpp`.

6.53.4 Member Data Documentation

6.53.4.1 dep_

```
template<typename Kind , typename ... T>
future_dependency<future1<Kind, T...> > upcxx::detail::future_body_pure< future1< Kind, T...
> >::dep_
```

Definition at line 13 of file body_pure.hpp.

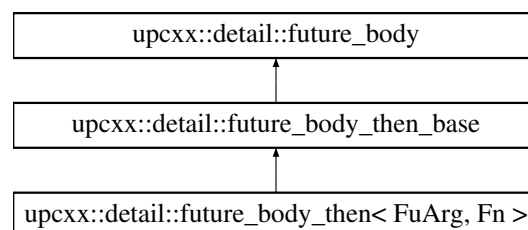
The documentation for this struct was generated from the following file:

- [future/body_pure.hpp](#)

6.54 upcxx::detail::future_body_then< FuArg, Fn > Struct Template Reference

```
#include <core.hpp>
```

Inheritance diagram for upcxx::detail::future_body_then< FuArg, Fn >:



Public Member Functions

- [future_body_then](#) (void *storage, [future_header_dependent](#) *hdr, FuArg arg, Fn fn)
- void [leave_active](#) ([future_header_dependent](#) *hdr)

Public Attributes

- [future_dependency](#)< FuArg > [dep_](#)
- Fn [fn_](#)

6.54.1 Detailed Description

```
template<typename FuArg, typename Fn>
struct upcxx::detail::future_body_then< FuArg, Fn >
```

Definition at line 27 of file core.hpp.

6.54.2 Constructor & Destructor Documentation

6.54.2.1 future_body_then()

```
template<typename FuArg , typename Fn >
upcxx::detail::future_body_then< FuArg, Fn >::future_body_then (
    void * storage,
    future_header_dependent * hdr,
    FuArg arg,
    Fn fn ) [inline]
```

Definition at line 59 of file then.hpp.

6.54.3 Member Function Documentation

6.54.3.1 leave_active()

```
template<typename FuArg , typename Fn >
void upcxx::detail::future_body_then< FuArg, Fn >::leave_active (
    future_header_dependent * hdr ) [inline], [virtual]
```

Implements [upcxx::detail::future_body](#).

Definition at line 71 of file then.hpp.

6.54.4 Member Data Documentation

6.54.4.1 dep_

```
template<typename FuArg , typename Fn >
future_dependency<FuArg> upcxx::detail::future_body_then< FuArg, Fn >::dep_
```

Definition at line 56 of file then.hpp.

6.54.4.2 fn_

```
template<typename FuArg , typename Fn >
Fn upcxx::detail::future_body_then< FuArg, Fn >::fn_
```

Definition at line 57 of file then.hpp.

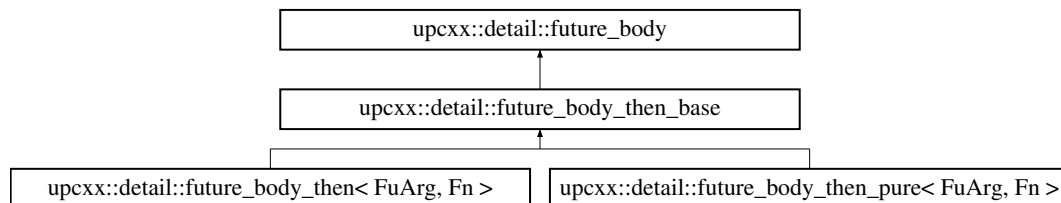
The documentation for this struct was generated from the following files:

- [future/core.hpp](#)
- [future/then.hpp](#)

6.55 upcxx::detail::future_body_then_base Struct Reference

```
#include <then.hpp>
```

Inheritance diagram for upcxx::detail::future_body_then_base:



Public Member Functions

- [future_body_then_base](#) (void *storage)
- [template<typename Kind , typename ... T> void leave_active_into_proxy](#) (bool pure, [future_header_dependent](#) *hdr, void *storage, [future1](#)< Kind, T...> proxied)

Additional Inherited Members

6.55.1 Detailed Description

Definition at line 17 of file then.hpp.

6.55.2 Constructor & Destructor Documentation

6.55.2.1 future_body_then_base()

```
upcxx::detail::future_body_then_base::future_body_then_base (
    void * storage ) [inline]
```

Definition at line 18 of file then.hpp.

6.55.3 Member Function Documentation

6.55.3.1 leave_active_into_proxy()

```
template<typename Kind , typename ... T>
void upcxx::detail::future_body_then_base::leave_active_into_proxy (
    bool pure,
    future_header_dependent * hdr,
    void * storage,
    future1< Kind, T... > proxied ) [inline]
```

Definition at line 22 of file then.hpp.

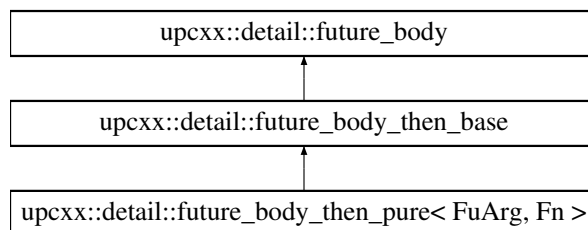
The documentation for this struct was generated from the following file:

- [future/then.hpp](#)

6.56 upcxx::detail::future_body_then_pure< FuArg, Fn > Struct Template Reference

```
#include <then.hpp>
```

Inheritance diagram for upcxx::detail::future_body_then_pure< FuArg, Fn >:



Public Member Functions

- [future_body_then_pure](#) (void *storage, [future_header_dependent](#) *hdr, FuArg arg, Fn fn)
- void [destruct_early](#) ()
- void [leave_active](#) ([future_header_dependent](#) *hdr)

Public Attributes

- [future_dependency](#)< FuArg > [dep_](#)
- Fn [fn_](#)

6.56.1 Detailed Description

```
template<typename FuArg, typename Fn>
struct upcxx::detail::future_body_then_pure< FuArg, Fn >
```

Definition at line 88 of file then.hpp.

6.56.2 Constructor & Destructor Documentation

6.56.2.1 future_body_then_pure()

```
template<typename FuArg, typename Fn>
upcxx::detail::future_body_then_pure< FuArg, Fn >::future_body_then_pure (
    void * storage,
    future_header_dependent * hdr,
    FuArg arg,
    Fn fn ) [inline]
```

Definition at line 92 of file then.hpp.

6.56.3 Member Function Documentation

6.56.3.1 destruct_early()

```
template<typename FuArg, typename Fn>
void upcxx::detail::future_body_then_pure< FuArg, Fn >::destruct_early ( ) [inline], [virtual]
```

Reimplemented from [upcxx::detail::future_body](#).

Definition at line 102 of file then.hpp.

6.56.3.2 leave_active()

```
template<typename FuArg, typename Fn>
void upcxx::detail::future_body_then_pure< FuArg, Fn >::leave_active (
    future_header_dependent * hdr ) [inline], [virtual]
```

Implements [upcxx::detail::future_body](#).

Definition at line 107 of file then.hpp.

6.56.4 Member Data Documentation

6.56.4.1 `dep_`

```
template<typename FuArg, typename Fn>  
future_dependency<FuArg> upcxx::detail::future_body_then_pure< FuArg, Fn >::dep_
```

Definition at line 89 of file `then.hpp`.

6.56.4.2 `fn_`

```
template<typename FuArg, typename Fn>  
Fn upcxx::detail::future_body_then_pure< FuArg, Fn >::fn_
```

Definition at line 90 of file `then.hpp`.

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

- [future/then.hpp](#)

6.57 `upcxx::future_cx< ordinal >` Struct Template Reference

```
#include <completion.hpp>
```

6.57.1 Detailed Description

```
template<int ordinal>  
struct upcxx::future_cx< ordinal >
```

Definition at line 17 of file `completion.hpp`.

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

- [completion.hpp](#)

6.58 `upcxx::detail::future_dependency< FuArg >` Struct Template Reference

```
#include <core.hpp>
```

6.58.1 Detailed Description

```
template<typename FuArg>
struct upcxx::detail::future_dependency< FuArg >
```

Definition at line 18 of file core.hpp.

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

- [future/core.hpp](#)

6.59 upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > > Struct Template Reference

```
#include <impl_mapped.hpp>
```

Public Types

- typedef [future_impl_mapped< FuArg, Fn, T... >::result_lrefs_function](#) [result_lrefs_function](#)

Public Member Functions

- [future_dependency](#) ([future_header_dependent](#) *suc_hdr, [future1< future_kind_mapped< FuArg, Fn >, T... > arg](#))
- void [cleanup_early](#) ()
- void [cleanup_ready](#) ()
- [result_lrefs_function](#) [result_lrefs_getter](#) () const
- [future_header](#) * [cleanup_ready_get_header](#) ()

Public Attributes

- [future_dependency< FuArg > dep_](#)
- [Fn fn_](#)

6.59.1 Detailed Description

```
template<typename FuArg, typename Fn, typename ... T>
struct upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >
```

Definition at line 115 of file impl_mapped.hpp.

6.59.2 Member Typedef Documentation

6.59.2.1 result_lrefs_function

```
template<typename FuArg , typename Fn , typename ... T>
typedef future_impl_mapped<FuArg,Fn,T...>::result_lrefs_function upcxx::detail::future_↵
dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >::result_lrefs_function
```

Definition at line 133 of file impl_mapped.hpp.

6.59.3 Constructor & Destructor Documentation

6.59.3.1 future_dependency()

```
template<typename FuArg , typename Fn , typename ... T>
upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >↵
::future_dependency (
    future_header_dependent * suc_hdr,
    future1< future_kind_mapped< FuArg, Fn >, T... > arg ) [inline]
```

Definition at line 122 of file impl_mapped.hpp.

6.59.4 Member Function Documentation

6.59.4.1 cleanup_early()

```
template<typename FuArg , typename Fn , typename ... T>
void upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >↵
::cleanup_early ( ) [inline]
```

Definition at line 130 of file impl_mapped.hpp.

6.59.4.2 cleanup_ready()

```
template<typename FuArg , typename Fn , typename ... T>
void upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >↵
::cleanup_ready ( ) [inline]
```

Definition at line 131 of file impl_mapped.hpp.

6.59.4.3 cleanup_ready_get_header()

```
template<typename FuArg , typename Fn , typename ... T>
future_header* upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >,
T... > >::cleanup_ready_get_header ( ) [inline]
```

Definition at line 141 of file impl_mapped.hpp.

6.59.4.4 result_lrefs_getter()

```
template<typename FuArg , typename Fn , typename ... T>
result_lrefs_function upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg,
Fn >, T... > >::result_lrefs_getter ( ) const [inline]
```

Definition at line 135 of file impl_mapped.hpp.

6.59.5 Member Data Documentation

6.59.5.1 dep_

```
template<typename FuArg , typename Fn , typename ... T>
future_dependency<FuArg> upcxx::detail::future_dependency< future1< future_kind_mapped< Fu←
Arg, Fn >, T... > >::dep_
```

Definition at line 118 of file impl_mapped.hpp.

6.59.5.2 fn_

```
template<typename FuArg , typename Fn , typename ... T>
Fn upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >↔
::fn_
```

Definition at line 119 of file impl_mapped.hpp.

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

- [future/impl_mapped.hpp](#)

6.60 upcxx::detail::future_dependency< future1< future_kind_result > > Struct Template Reference

```
#include <impl_result.hpp>
```

Public Member Functions

- `future_dependency` (`future_header_dependent *suc_hdr, future1< future_kind_result > arg`)
- `void cleanup_early` ()
- `upcxx::constant_function< std::tuple<> > result_lrefs_getter` () const
- `void cleanup_ready` ()
- `future_header * cleanup_ready_get_header` ()

6.60.1 Detailed Description

```
template<>
struct upcxx::detail::future_dependency< future1< future_kind_result > >
```

Definition at line 103 of file `impl_result.hpp`.

6.60.2 Constructor & Destructor Documentation

6.60.2.1 `future_dependency()`

```
upcxx::detail::future_dependency< future1< future_kind_result > >::future_dependency (
    future_header_dependent * suc_hdr,
    future1< future_kind_result > arg ) [inline]
```

Definition at line 106 of file `impl_result.hpp`.

6.60.3 Member Function Documentation

6.60.3.1 `cleanup_early()`

```
void upcxx::detail::future_dependency< future1< future_kind_result > >::cleanup_early ( )
[inline]
```

Definition at line 112 of file `impl_result.hpp`.

6.60.3.2 `cleanup_ready()`

```
void upcxx::detail::future_dependency< future1< future_kind_result > >::cleanup_ready ( )
[inline]
```

Definition at line 118 of file `impl_result.hpp`.

6.60.3.3 cleanup_ready_get_header()

```
future_header* upcxx::detail::future_dependency< future1< future_kind_result > >::cleanup_↔
ready_get_header ( ) [inline]
```

Definition at line 120 of file impl_result.hpp.

6.60.3.4 result_lrefs_getter()

```
upcxx::constant_function<std::tuple<> > upcxx::detail::future_dependency< future1< future_↔
kind_result > >::result_lrefs_getter ( ) const [inline]
```

Definition at line 114 of file impl_result.hpp.

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

- [future/impl_result.hpp](#)

6.61 upcxx::detail::future_dependency< future1< future_kind_result, T... > > Struct Template Reference

```
#include <impl_result.hpp>
```

Public Member Functions

- [future_dependency](#) ([future_header_dependent](#) *suc_hdr, [future1](#)< [future_kind_result](#), T... > arg)
- void [cleanup_early](#) ()
- [upcxx::constant_function](#)< [std::tuple](#)< T &... > > [result_lrefs_getter](#) () const
- void [cleanup_ready](#) ()
- [future_header](#) * [cleanup_ready_get_header](#) ()

Public Attributes

- [std::tuple](#)< T... > [results_](#)

6.61.1 Detailed Description

```
template<typename ... T>
struct upcxx::detail::future_dependency< future1< future_kind_result, T... > >
```

Definition at line 72 of file impl_result.hpp.

6.61.2 Constructor & Destructor Documentation

6.61.2.1 future_dependency()

```
template<typename ... T>
upcxx::detail::future_dependency< future1< future_kind_result, T... > >::future_dependency (
    future_header_dependent * suc_hdr,
    future1< future_kind_result, T... > arg ) [inline]
```

Definition at line 77 of file impl_result.hpp.

6.61.3 Member Function Documentation

6.61.3.1 cleanup_early()

```
template<typename ... T>
void upcxx::detail::future_dependency< future1< future_kind_result, T... > >::cleanup_early
( ) [inline]
```

Definition at line 84 of file impl_result.hpp.

6.61.3.2 cleanup_ready()

```
template<typename ... T>
void upcxx::detail::future_dependency< future1< future_kind_result, T... > >::cleanup_ready
( ) [inline]
```

Definition at line 92 of file impl_result.hpp.

6.61.3.3 cleanup_ready_get_header()

```
template<typename ... T>
future_header* upcxx::detail::future_dependency< future1< future_kind_result, T... > >↵
::cleanup_ready_get_header ( ) [inline]
```

Definition at line 94 of file impl_result.hpp.

6.61.3.4 result_lrefs_getter()

```
template<typename ... T>
upcxx::constant_function<std::tuple<T&...> > upcxx::detail::future_dependency< future1<
future_kind_result, T... > >::result_lrefs_getter ( ) const [inline]
```

Definition at line 86 of file impl_result.hpp.

6.61.4 Member Data Documentation

6.61.4.1 results_

```
template<typename ... T>
std::tuple<T...> upcxx::detail::future_dependency< future1< future_kind_result, T... > >↔
::results_
```

Definition at line 75 of file impl_result.hpp.

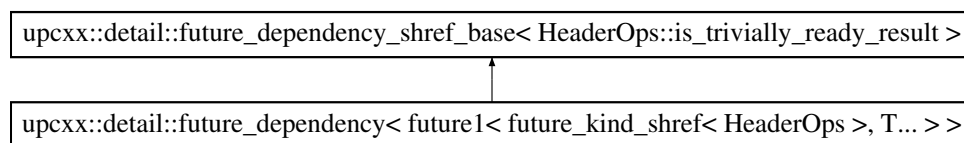
The documentation for this struct was generated from the following file:

- [future/impl_result.hpp](#)

6.62 upcxx::detail::future_dependency< future1< future_kind_shref< HeaderOps >, T... > > Struct Template Reference

```
#include <impl_shref.hpp>
```

Inheritance diagram for upcxx::detail::future_dependency< future1< future_kind_shref< HeaderOps >, T... > >:



Public Member Functions

- [future_dependency](#) ([future_header_dependent](#) *suc_hdr, [future1](#)< [future_kind_shref](#)< HeaderOps >, T... > arg)

6.62.1 Detailed Description

```
template<typename HeaderOps, typename ... T>
struct upcxx::detail::future_dependency< future1< future_kind_shref< HeaderOps >, T... > >
```

Definition at line 227 of file impl_shref.hpp.

6.62.2 Constructor & Destructor Documentation

6.62.2.1 future_dependency()

```
template<typename HeaderOps , typename ... T>
upcxx::detail::future_dependency< future1< future_kind_shref< HeaderOps >, T... > >::future←
_dependency (
    future_header_dependent * suc_hdr,
    future1< future_kind_shref< HeaderOps >, T... > arg ) [inline]
```

Definition at line 232 of file impl_shref.hpp.

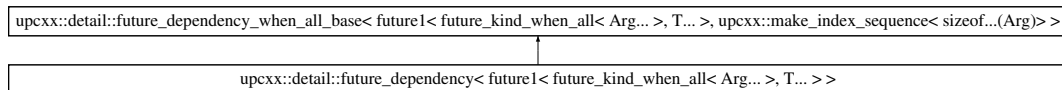
The documentation for this struct was generated from the following file:

- [future/impl_shref.hpp](#)

6.63 upcxx::detail::future_dependency< future1< future_kind_when_all< Arg... >, T... > > Struct Template Reference

```
#include <impl_when_all.hpp>
```

Inheritance diagram for upcxx::detail::future_dependency< future1< future_kind_when_all< Arg... >, T... > >:



Public Member Functions

- [future_dependency](#) ([future_header_dependent](#) *suc_hdr, [future1](#)< [future_kind_when_all](#)< Arg... >, T... > arg)

6.63.1 Detailed Description

```
template<typename ... Arg, typename ... T>
struct upcxx::detail::future_dependency< future1< future_kind_when_all< Arg... >, T... > >
```

Definition at line 200 of file impl_when_all.hpp.

6.63.2 Constructor & Destructor Documentation

6.63.2.1 future_dependency()

```
template<typename ... Arg, typename ... T>
upcxx::detail::future_dependency< future1< future_kind_when_all< Arg... >, T... > >↵
::future_dependency (
    future_header_dependent * suc_hdr,
    future1< future_kind_when_all< Arg... >, T... > arg ) [inline]
```

Definition at line 208 of file impl_when_all.hpp.

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

- [future/impl_when_all.hpp](#)

6.64 upcxx::detail::future_dependency_shref_base< is_trivially_ready_result > Struct Template Reference

```
#include <impl_shref.hpp>
```

6.64.1 Detailed Description

```
template<bool is_trivially_ready_result>
struct upcxx::detail::future_dependency_shref_base< is_trivially_ready_result >
```

Definition at line 173 of file impl_shref.hpp.

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

- [future/impl_shref.hpp](#)

6.65 upcxx::detail::future_dependency_shref_base< false > Struct Template Reference

```
#include <impl_shref.hpp>
```

Public Member Functions

- [future_dependency_shref_base](#) ([future_header_dependent](#) *suc_hdr, [future_header](#) *arg_hdr)
- [future_dependency_shref_base](#) ([future_dependency_shref_base](#) const &)=delete
- [future_dependency_shref_base](#) ([future_dependency_shref_base](#) &&)=delete
- void [unlink_](#) ()
- [future_header](#) * [header_](#) () const

Public Attributes

- [future_header::dependency_link](#) [link_](#)

6.65.1 Detailed Description

```
template<>
struct upcxx::detail::future_dependency_shref_base< false >
```

Definition at line 176 of file impl_shref.hpp.

6.65.2 Constructor & Destructor Documentation

6.65.2.1 future_dependency_shref_base() [1/3]

```
upcxx::detail::future_dependency_shref_base< false >::future_dependency_shref_base (
    future_header_dependent * suc_hdr,
    future_header * arg_hdr ) [inline]
```

Definition at line 179 of file impl_shref.hpp.

6.65.2.2 future_dependency_shref_base() [2/3]

```
upcxx::detail::future_dependency_shref_base< false >::future_dependency_shref_base (
    future_dependency_shref_base< false > const & ) [delete]
```

6.65.2.3 future_dependency_shref_base() [3/3]

```
upcxx::detail::future_dependency_shref_base< false >::future_dependency_shref_base (
    future_dependency_shref_base< false > && ) [delete]
```

6.65.3 Member Function Documentation

6.65.3.1 header_()

```
future_header* upcxx::detail::future_dependency_shref_base< false >::header_ ( ) const [inline]
```

Definition at line 207 of file impl_shref.hpp.

6.65.3.2 `unlink_()`

```
void upcxx::detail::future_dependency_shref_base< false >::unlink_ ( ) [inline]
```

Definition at line 205 of file `impl_shref.hpp`.

6.65.4 Member Data Documentation

6.65.4.1 `link_`

```
future_header::dependency_link upcxx::detail::future_dependency_shref_base< false >::link_
```

Definition at line 177 of file `impl_shref.hpp`.

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

- [future/impl_shref.hpp](#)

6.66 `upcxx::detail::future_dependency_shref_base< true >` Struct Template Reference

```
#include <impl_shref.hpp>
```

Public Member Functions

- `future_dependency_shref_base` (`future_header_dependent *suc_hdr`, `future_header *arg_hdr`)
- void `unlink_()`
- `future_header * header_()` const

Public Attributes

- `future_header * hdr_`

6.66.1 Detailed Description

```
template<>
struct upcxx::detail::future_dependency_shref_base< true >
```

Definition at line 211 of file `impl_shref.hpp`.

6.66.2 Constructor & Destructor Documentation

6.66.2.1 future_dependency_shref_base()

```
upcxx::detail::future_dependency_shref_base< true >::future_dependency_shref_base (
    future_header_dependent * suc_hdr,
    future_header * arg_hdr ) [inline]
```

Definition at line 214 of file impl_shref.hpp.

6.66.3 Member Function Documentation

6.66.3.1 header_()

```
future_header* upcxx::detail::future_dependency_shref_base< true >::header_ ( ) const [inline]
```

Definition at line 223 of file impl_shref.hpp.

6.66.3.2 unlink_()

```
void upcxx::detail::future_dependency_shref_base< true >::unlink_ ( ) [inline]
```

Definition at line 221 of file impl_shref.hpp.

6.66.4 Member Data Documentation

6.66.4.1 hdr_

```
future_header* upcxx::detail::future_dependency_shref_base< true >::hdr_
```

Definition at line 212 of file impl_shref.hpp.

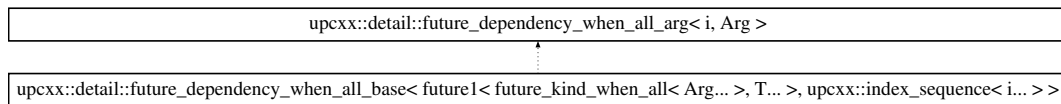
The documentation for this struct was generated from the following file:

- [future/impl_shref.hpp](#)

6.67 `upcxx::detail::future_dependency_when_all_arg< i, Arg >` Struct Template Reference

```
#include <impl_when_all.hpp>
```

Inheritance diagram for `upcxx::detail::future_dependency_when_all_arg< i, Arg >`:



Public Member Functions

- `future_dependency_when_all_arg` (`future_header_dependent *suc_hdr`, `Arg arg`)

Public Attributes

- `future_dependency< Arg > dep_`

6.67.1 Detailed Description

```
template<int i, typename Arg>
struct upcxx::detail::future_dependency_when_all_arg< i, Arg >
```

Definition at line 121 of file `impl_when_all.hpp`.

6.67.2 Constructor & Destructor Documentation

6.67.2.1 `future_dependency_when_all_arg()`

```
template<int i, typename Arg >
upcxx::detail::future_dependency_when_all_arg< i, Arg >::future_dependency_when_all_arg (
    future_header_dependent * suc_hdr,
    Arg arg ) [inline]
```

Definition at line 124 of file `impl_when_all.hpp`.

6.67.3 Member Data Documentation

6.67.3.1 `dep_`

```
template<int i, typename Arg >
future_dependency<Arg> upcxx::detail::future_dependency_when_all_arg< i, Arg >::dep_
```

Definition at line 122 of file `impl_when_all.hpp`.

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

- [future/impl_when_all.hpp](#)

6.68 `upcxx::detail::future_dependency_when_all_base< AllArg, lxSeq >` Struct Template Reference

```
#include <impl_when_all.hpp>
```

6.68.1 Detailed Description

```
template<typename AllArg, typename lxSeq>
struct upcxx::detail::future_dependency_when_all_base< AllArg, lxSeq >
```

Definition at line 133 of file `impl_when_all.hpp`.

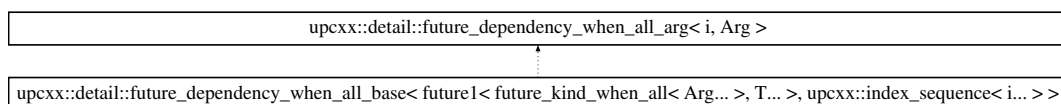
The documentation for this struct was generated from the following file:

- [future/impl_when_all.hpp](#)

6.69 `upcxx::detail::future_dependency_when_all_base< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... > >` Struct Template Reference

```
#include <impl_when_all.hpp>
```

Inheritance diagram for `upcxx::detail::future_dependency_when_all_base< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... > >`:



Public Types

- typedef `future_dependency_when_all_base< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... > >` `this_t`

Public Member Functions

- [future_dependency_when_all_base](#) ([future_header_dependent](#) *[suc_hdr](#), [future1](#)< [future_kind_when_all](#)< [Arg...](#) >, [T...](#) > [all_args](#))
- [result_lrefs_function](#) [result_lrefs_getter](#) () const

6.69.1 Detailed Description

```
template<typename ... Arg, typename ... T, int ... i>
struct upcxx::detail::future_dependency_when_all_base< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_↵
sequence< i... > >
```

Definition at line 136 of file `impl_when_all.hpp`.

6.69.2 Member Typedef Documentation

6.69.2.1 this_t

```
template<typename ... Arg, typename ... T, int ... i>
typedef future_dependency_when_all_base< future1<future_kind_when_all<Arg...>, T...>, upcxx↵
::index_sequence<i...> > upcxx::detail::future_dependency_when_all_base< future1< future_↵
kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... > >::this_t
```

Definition at line 146 of file `impl_when_all.hpp`.

6.69.3 Constructor & Destructor Documentation

6.69.3.1 future_dependency_when_all_base()

```
template<typename ... Arg, typename ... T, int ... i>
upcxx::detail::future_dependency_when_all_base< future1< future_kind_when_all< Arg... >,
T... >, upcxx::index_sequence< i... > >::future_dependency_when_all_base (
    future_header_dependent * suc_hdr,
    future1< future_kind_when_all< Arg... >, T... > all_args ) [inline]
```

Definition at line 148 of file `impl_when_all.hpp`.

6.69.4 Member Function Documentation

6.69.4.1 `result_lrefs_getter()`

```
template<typename ... Arg, typename ... T, int ... i>
result_lrefs_function upcxx::detail::future_dependency_when_all_base< future1< future_kind←
_when_all< Arg... >, T... >, upcxx::index_sequence< i... > >::result_lrefs_getter ( )
const [inline]
```

Definition at line 194 of file `impl_when_all.hpp`.

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

- [future/impl_when_all.hpp](#)

6.70 `upcxx::detail::future_from_tuple< Kind, Tup >` Struct Template Reference

```
#include <core.hpp>
```

6.70.1 Detailed Description

```
template<typename Kind, typename Tup>
struct upcxx::detail::future_from_tuple< Kind, Tup >
```

Definition at line 124 of file `core.hpp`.

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

- [future/core.hpp](#)

6.71 `upcxx::detail::future_from_tuple< Kind, std::tuple< T... > >` Struct Template Reference

```
#include <core.hpp>
```

Public Types

- using `type` = `future1< Kind, T... >`

6.71.1 Detailed Description

```
template<typename Kind, typename ... T>
struct upcxx::detail::future_from_tuple< Kind, std::tuple< T... > >
```

Definition at line 126 of file `core.hpp`.

6.71.2 Member Typedef Documentation

6.71.2.1 type

```
template<typename Kind , typename ... T>
using upcxx::detail::future_from_tuple< Kind, std::tuple< T... > >::type = future1<Kind,
T...>
```

Definition at line 127 of file core.hpp.

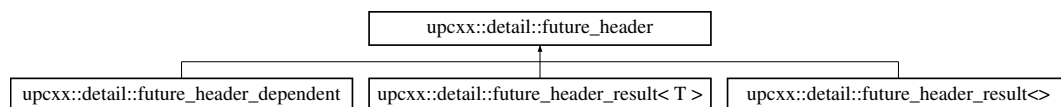
The documentation for this struct was generated from the following file:

- [future/core.hpp](#)

6.72 upcxx::detail::future_header Struct Reference

```
#include <core.hpp>
```

Inheritance diagram for upcxx::detail::future_header:



Classes

- struct [dependency_link](#)

Public Member Functions

- void [enter_ready](#) ([future_header](#) *result)
- void [refs_add](#) (int n)
- int [refs_drop](#) (int n)

Static Public Member Functions

- static [future_header](#) * [drop_for_result](#) ([future_header](#) *a)
- static [future_header](#) * [drop_for_proxied](#) ([future_header](#) *a)

Public Attributes

- int `ref_n_`
- int `status_`
- `dependency_link * succs_head_`
- union {
 - `future_header * result_`
 - `future_body * body_`

Static Public Attributes

- static constexpr int `status_active` = 0
- static constexpr int `status_ready` = -1
- static constexpr int `status_proxying` = -2
- static constexpr int `status_proxying_active` = -3
- static `future_header the_nil`

6.72.1 Detailed Description

Definition at line 158 of file core.hpp.

6.72.2 Member Function Documentation

6.72.2.1 drop_for_proxied()

```
future_header * upcxx::detail::future_header::drop_for_proxied (  
    future_header * a ) [inline], [static]
```

Definition at line 593 of file core.hpp.

6.72.2.2 drop_for_result()

```
future_header * upcxx::detail::future_header::drop_for_result (  
    future_header * a ) [inline], [static]
```

Definition at line 564 of file core.hpp.

6.72.2.3 enter_ready()

```
void future_header::enter_ready (
    future_header * result )
```

Definition at line 73 of file core.cpp.

6.72.2.4 refs_add()

```
void upcxx::detail::future_header::refs_add (
    int n ) [inline]
```

Definition at line 223 of file core.hpp.

6.72.2.5 refs_drop()

```
int upcxx::detail::future_header::refs_drop (
    int n ) [inline]
```

Definition at line 228 of file core.hpp.

6.72.3 Member Data Documentation

6.72.3.1 "@2

```
union { ... }
```

6.72.3.2 body_

```
future_body* upcxx::detail::future_header::body_
```

Definition at line 211 of file core.hpp.

6.72.3.3 ref_n_

```
int upcxx::detail::future_header::ref_n_
```

Definition at line 160 of file core.hpp.

6.72.3.4 result_

`future_header*` upcxx::detail::future_header::result_

Definition at line 207 of file core.hpp.

6.72.3.5 status_

`int` upcxx::detail::future_header::status_

Definition at line 185 of file core.hpp.

6.72.3.6 status_active

`constexpr int` upcxx::detail::future_header::status_active = 0 [static]

Definition at line 171 of file core.hpp.

6.72.3.7 status_proxying

`constexpr int` upcxx::detail::future_header::status_proxying = -2 [static]

Definition at line 179 of file core.hpp.

6.72.3.8 status_proxying_active

`constexpr int` upcxx::detail::future_header::status_proxying_active = -3 [static]

Definition at line 183 of file core.hpp.

6.72.3.9 status_ready

`constexpr int` upcxx::detail::future_header::status_ready = -1 [static]

Definition at line 175 of file core.hpp.

6.72.3.10 `sucs_head_`

`dependency_link*` `upcxx::detail::future_header::sucs_head_`

Definition at line 201 of file `core.hpp`.

6.72.3.11 `the_nil`

`future_header` `future_header::the_nil` [static]

Initial value:

```
= {
  -1,
  future_header::status_active + 666,
  nullptr,
  {nullptr}
}
```

Definition at line 220 of file `core.hpp`.

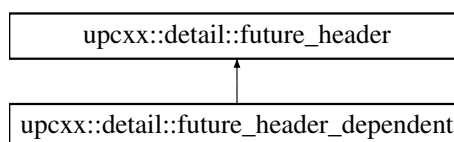
The documentation for this struct was generated from the following files:

- [future/core.hpp](#)
- [future/core.cpp](#)

6.73 `upcxx::detail::future_header_dependent` Struct Reference

```
#include <core.hpp>
```

Inheritance diagram for `upcxx::detail::future_header_dependent`:



Public Member Functions

- [future_header_dependent](#) ()
- void [entered_active](#) ()
- void [enter_proxying](#) ([future_body_proxy_](#) *body, [future_header](#) *proxied)
- void [refs_add](#) (int n)
- int [refs_drop](#) (int n)

Public Attributes

- [future_header_dependent](#) * [active_next_](#)

Additional Inherited Members

6.73.1 Detailed Description

Definition at line 253 of file core.hpp.

6.73.2 Constructor & Destructor Documentation

6.73.2.1 future_header_dependent()

```
upcxx::detail::future_header_dependent::future_header_dependent ( ) [inline]
```

Definition at line 257 of file core.hpp.

6.73.3 Member Function Documentation

6.73.3.1 enter_proxying()

```
void future_header_dependent::enter_proxying (
    future_body_proxy_ * body,
    future_header * proxied )
```

Definition at line 150 of file core.cpp.

6.73.3.2 entered_active()

```
void future_header_dependent::entered_active ( )
```

Definition at line 40 of file core.cpp.

6.73.3.3 refs_add()

```
void upcxx::detail::future_header_dependent::refs_add (
    int n ) [inline]
```

Definition at line 273 of file core.hpp.

6.73.3.4 refs_drop()

```
int upcxx::detail::future_header_dependent::refs_drop (
    int n ) [inline]
```

Definition at line 276 of file core.hpp.

6.73.4 Member Data Documentation

6.73.4.1 active_next_

```
future_header_dependent* upcxx::detail::future_header_dependent::active_next_
```

Definition at line 255 of file core.hpp.

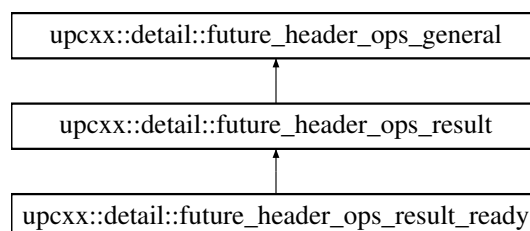
The documentation for this struct was generated from the following files:

- [future/core.hpp](#)
- [future/core.cpp](#)

6.74 upcxx::detail::future_header_ops_general Struct Reference

```
#include <core.hpp>
```

Inheritance diagram for upcxx::detail::future_header_ops_general:



Static Public Member Functions

- `template<typename ... T>`
static void `decref_header` (`future_header *hdr`)
- `template<typename ... T>`
static void `delete_header` (`future_header *hdr`)

Static Public Attributes

- static constexpr bool `is_trivially_ready_result` = false

6.74.1 Detailed Description

Definition at line 286 of file core.hpp.

6.74.2 Member Function Documentation

6.74.2.1 `decref_header()`

```
template<typename ... T>
void upcxx::detail::future_header_ops_general::decref_header (
    future_header * hdr ) [inline], [static]
```

Definition at line 526 of file core.hpp.

6.74.2.2 `delete_header()`

```
template<typename ... T>
void upcxx::detail::future_header_ops_general::delete_header (
    future_header * hdr ) [static]
```

Definition at line 490 of file core.hpp.

6.74.3 Member Data Documentation

6.74.3.1 is_trivially_ready_result

```
constexpr bool upcxx::detail::future_header_ops_general::is_trivially_ready_result = false
[static]
```

Definition at line 287 of file core.hpp.

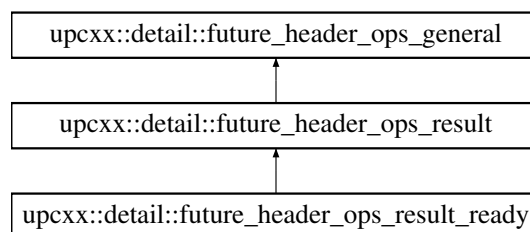
The documentation for this struct was generated from the following file:

- [future/core.hpp](#)

6.75 upcxx::detail::future_header_ops_result Struct Reference

```
#include <core.hpp>
```

Inheritance diagram for upcxx::detail::future_header_ops_result:



Static Public Member Functions

- `template<typename ... T>`
static void [decref_header](#) (`future_header *hdr`)
- `template<typename ... T>`
static void [delete_header](#) (`future_header *hdr`)

Static Public Attributes

- static constexpr bool [is_trivially_ready_result](#) = false

6.75.1 Detailed Description

Definition at line 300 of file core.hpp.

6.75.2 Member Function Documentation

6.75.2.1 decref_header()

```
template<typename ... T>
void upcxx::detail::future_header_ops_result::decref_header (
    future_header * hdr ) [static]
```

Definition at line 541 of file core.hpp.

6.75.2.2 delete_header()

```
template<typename ... T>
void upcxx::detail::future_header_ops_result::delete_header (
    future_header * hdr ) [static]
```

Definition at line 536 of file core.hpp.

6.75.3 Member Data Documentation

6.75.3.1 is_trivially_ready_result

```
constexpr bool upcxx::detail::future_header_ops_result::is_trivially_ready_result = false
[static]
```

Definition at line 301 of file core.hpp.

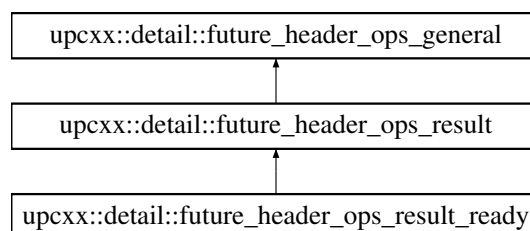
The documentation for this struct was generated from the following file:

- [future/core.hpp](#)

6.76 upcxx::detail::future_header_ops_result_ready Struct Reference

```
#include <core.hpp>
```

Inheritance diagram for upcxx::detail::future_header_ops_result_ready:



Static Public Member Functions

- `template<typename ... T>`
static void `decref_header` (`future_header *hdr`)
- `template<typename ... T>`
static void `delete_header` (`future_header *hdr`)

Static Public Attributes

- static constexpr bool `is_trivially_ready_result` = true

6.76.1 Detailed Description

Definition at line 314 of file `core.hpp`.

6.76.2 Member Function Documentation

6.76.2.1 `decref_header()`

```
template<typename ... T>
void upcxx::detail::future_header_ops_result_ready::decref_header (
    future_header * hdr ) [static]
```

Definition at line 556 of file `core.hpp`.

6.76.2.2 `delete_header()`

```
template<typename ... T>
void upcxx::detail::future_header_ops_result_ready::delete_header (
    future_header * hdr ) [static]
```

Definition at line 551 of file `core.hpp`.

6.76.3 Member Data Documentation

6.76.3.1 is_trivially_ready_result

```
constexpr bool upcxx::detail::future_header_ops_result_ready::is_trivially_ready_result = true
[static]
```

Definition at line 315 of file core.hpp.

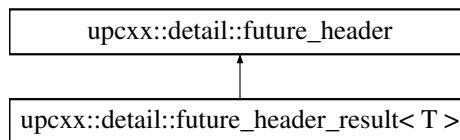
The documentation for this struct was generated from the following file:

- [future/core.hpp](#)

6.77 upcxx::detail::future_header_result< T > Struct Template Reference

```
#include <core.hpp>
```

Inheritance diagram for upcxx::detail::future_header_result< T >:



Public Member Functions

- [future_header_result](#) ()
- [template<typename ... U> future_header_result](#) (bool not_ready, std::tuple< U... > values)
- [void readify](#) ()
- [template<typename ... U> void construct_results](#) (U &&...values)
- [template<typename ... U> void construct_results](#) (std::tuple< U... > &&values)
- [void delete_me_ready](#) ()
- [void delete_me](#) ()

Static Public Member Functions

- [static std::tuple< T... > & results_of](#) ([future_header](#) *hdr)

Public Attributes

- [union](#) {
 std::tuple< T... > [results_](#)
 };

Static Public Attributes

- static constexpr int `status_results_yes` = `status_active` + 1
- static constexpr int `status_results_no` = `status_active` + 2

6.77.1 Detailed Description

```
template<typename ... T>
struct upcxx::detail::future_header_result< T >
```

Definition at line 14 of file core.hpp.

6.77.2 Constructor & Destructor Documentation

6.77.2.1 future_header_result() [1/2]

```
template<typename ... T>
upcxx::detail::future_header_result< T >::future_header_result ( ) [inline]
```

Definition at line 377 of file core.hpp.

6.77.2.2 future_header_result() [2/2]

```
template<typename ... T>
template<typename ... U>
upcxx::detail::future_header_result< T >::future_header_result (
    bool not_ready,
    std::tuple< U... > values ) [inline]
```

Definition at line 387 of file core.hpp.

6.77.3 Member Function Documentation

6.77.3.1 construct_results() [1/2]

```
template<typename ... T>
template<typename ... U>
void upcxx::detail::future_header_result< T >::construct_results (
    U &&... values ) [inline]
```

Definition at line 414 of file core.hpp.

6.77.3.2 construct_results() [2/2]

```
template<typename ... T>
template<typename ... U>
void upcxx::detail::future_header_result< T >::construct_results (
    std::tuple< U... > && values ) [inline]
```

Definition at line 421 of file core.hpp.

6.77.3.3 delete_me()

```
template<typename ... T>
void upcxx::detail::future_header_result< T >::delete_me ( ) [inline]
```

Definition at line 433 of file core.hpp.

6.77.3.4 delete_me_ready()

```
template<typename ... T>
void upcxx::detail::future_header_result< T >::delete_me_ready ( ) [inline]
```

Definition at line 427 of file core.hpp.

6.77.3.5 readify()

```
template<typename ... T>
void upcxx::detail::future_header_result< T >::readify ( ) [inline]
```

Definition at line 408 of file core.hpp.

6.77.3.6 results_of()

```
template<typename ... T>
static std::tuple<T...>& upcxx::detail::future_header_result< T >::results_of (
    future_header * hdr ) [inline], [static]
```

Definition at line 404 of file core.hpp.

6.77.4 Member Data Documentation

6.77.4.1 "@4

```
union { ... }
```

6.77.4.2 results_

```
template<typename ... T>
std::tuple<T...> upcxx::detail::future_header_result< T >::results_
```

Definition at line 375 of file core.hpp.

6.77.4.3 status_results_no

```
template<typename ... T>
constexpr int upcxx::detail::future_header_result< T >::status_results_no = status_active + 2
[static]
```

Definition at line 373 of file core.hpp.

6.77.4.4 status_results_yes

```
template<typename ... T>
constexpr int upcxx::detail::future_header_result< T >::status_results_yes = status_active + 1
[static]
```

Definition at line 372 of file core.hpp.

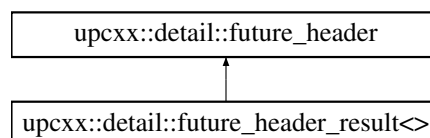
The documentation for this struct was generated from the following file:

- [future/core.hpp](#)

6.78 upcxx::detail::future_header_result<> Struct Template Reference

```
#include <core.hpp>
```

Inheritance diagram for upcxx::detail::future_header_result<>:



Public Types

- enum { `status_not_ready` = `status_active` + 1 }

Public Member Functions

- `future_header_result` ()
- `future_header_result` (bool `not_ready`, std::tuple<>)
- void `readify` ()
- void `construct_results` ()
- void `construct_results` (std::tuple<>)
- void `delete_me_ready` ()
- void `delete_me` ()

Static Public Member Functions

- static std::tuple `results_of` (`future_header` *hdr)

Static Public Attributes

- static `future_header` `the_always`

Additional Inherited Members

6.78.1 Detailed Description

```
template<>
struct upcxx::detail::future_header_result<>
```

Definition at line 444 of file core.hpp.

6.78.2 Member Enumeration Documentation

6.78.2.1 anonymous enum

anonymous enum

Enumerator

<code>status_not_ready</code>	
-------------------------------	--

Definition at line 447 of file core.hpp.

6.78.3 Constructor & Destructor Documentation

6.78.3.1 `future_header_result()` [1/2]

```
upcxx::detail::future_header_result<>::future_header_result ( ) [inline]
```

Definition at line 451 of file core.hpp.

6.78.3.2 `future_header_result()` [2/2]

```
upcxx::detail::future_header_result<>::future_header_result (
    bool not_ready,
    std::tuple<> ) [inline]
```

Definition at line 460 of file core.hpp.

6.78.4 Member Function Documentation

6.78.4.1 `construct_results()` [1/2]

```
void upcxx::detail::future_header_result<>::construct_results ( ) [inline]
```

Definition at line 478 of file core.hpp.

6.78.4.2 `construct_results()` [2/2]

```
void upcxx::detail::future_header_result<>::construct_results (
    std::tuple<> ) [inline]
```

Definition at line 479 of file core.hpp.

6.78.4.3 `delete_me()`

```
void upcxx::detail::future_header_result<>::delete_me ( ) [inline]
```

Definition at line 482 of file core.hpp.

6.78.4.4 `delete_me_ready()`

```
void upcxx::detail::future_header_result<>::delete_me_ready ( ) [inline]
```

Definition at line 481 of file `core.hpp`.

6.78.4.5 `readify()`

```
void upcxx::detail::future_header_result<>::readify ( ) [inline]
```

Definition at line 473 of file `core.hpp`.

6.78.4.6 `results_of()`

```
static std::tuple upcxx::detail::future_header_result<>::results_of (
    future_header * hdr ) [inline], [static]
```

Definition at line 469 of file `core.hpp`.

6.78.5 Member Data Documentation

6.78.5.1 `the_always`

```
future_header upcxx::detail::future_header_result<>::the_always [static]
```

Definition at line 445 of file `core.hpp`.

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

- [future/core.hpp](#)

6.79 `upcxx::detail::future_impl_mapped< FuArg, Fn, T >` Struct Template Reference

```
#include <core.hpp>
```

Classes

- struct [result_lrefs_function](#)

Public Types

- typedef [future_header_ops_general](#) [header_ops](#)

Public Member Functions

- [future_impl_mapped](#) (FuArg arg, Fn fn)
- bool [ready](#) () const
- [result_lrefs_function](#) [result_lrefs_getter](#) () const
- std::tuple< T... > [result_rvals](#) ()
- [future_header](#) * [steal_header](#) ()

Public Attributes

- FuArg [arg_](#)
- Fn [fn_](#)

6.79.1 Detailed Description

```
template<typename FuArg, typename Fn, typename ... T>
struct upcxx::detail::future_impl_mapped< FuArg, Fn, T >
```

Definition at line 42 of file core.hpp.

6.79.2 Member Typedef Documentation

6.79.2.1 header_ops

```
template<typename FuArg , typename Fn , typename ... T>
typedef future\_header\_ops\_general upcxx::detail::future\_impl\_mapped< FuArg, Fn, T >::header\_ops
```

Definition at line 90 of file impl_mapped.hpp.

6.79.3 Constructor & Destructor Documentation

6.79.3.1 future_impl_mapped()

```
template<typename FuArg , typename Fn , typename ... T>
upcxx::detail::future\_impl\_mapped< FuArg, Fn, T >::future\_impl\_mapped (
    FuArg arg,
    Fn fn ) [inline]
```

Definition at line 29 of file impl_mapped.hpp.

6.79.4 Member Function Documentation

6.79.4.1 ready()

```
template<typename FuArg , typename Fn , typename ... T>
bool upcxx::detail::future_impl_mapped< FuArg, Fn, T >::ready ( ) const [inline]
```

Definition at line 34 of file impl_mapped.hpp.

6.79.4.2 result_lrefs_getter()

```
template<typename FuArg , typename Fn , typename ... T>
result_lrefs_function upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_getter (
) const [inline]
```

Definition at line 76 of file impl_mapped.hpp.

6.79.4.3 result_rvals()

```
template<typename FuArg , typename Fn , typename ... T>
std::tuple<T...> upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_rvals ( ) [inline]
```

Definition at line 82 of file impl_mapped.hpp.

6.79.4.4 steal_header()

```
template<typename FuArg , typename Fn , typename ... T>
future_header* upcxx::detail::future_impl_mapped< FuArg, Fn, T >::steal_header ( ) [inline]
```

Definition at line 92 of file impl_mapped.hpp.

6.79.5 Member Data Documentation

6.79.5.1 `arg_`

```
template<typename FuArg , typename Fn , typename ... T>
FuArg upcxx::detail::future\_impl\_mapped< FuArg, Fn, T >::arg_
```

Definition at line 25 of file `impl_mapped.hpp`.

6.79.5.2 `fn_`

```
template<typename FuArg , typename Fn , typename ... T>
Fn upcxx::detail::future\_impl\_mapped< FuArg, Fn, T >::fn_
```

Definition at line 26 of file `impl_mapped.hpp`.

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

- [future/core.hpp](#)
- [future/impl_mapped.hpp](#)

6.80 `upcxx::detail::future_impl_result< T >` Struct Template Reference

```
#include <core.hpp>
```

Public Types

- typedef `future_header_ops_result_ready_header_ops`

Public Member Functions

- `future_impl_result` (T ...values)
- `bool ready` () const
- `upcxx::constant_function`< std::tuple< T &... > > `result_lrefs_getter` () const
- `auto result_rvals` () -> `decltype(upcxx::tuple_rvals(results_))`
- `future_header * steal_header` ()

Public Attributes

- `std::tuple< T... > results_`

6.80.1 Detailed Description

```
template<typename ... T>
struct upcxx::detail::future\_impl\_result< T >
```

Definition at line 38 of file `core.hpp`.

6.80.2 Member Typedef Documentation

6.80.2.1 header_ops

```
template<typename ... T>
typedef future_header_ops_result_ready upcxx::detail::future_impl_result< T >::header_ops
```

Definition at line 43 of file impl_result.hpp.

6.80.3 Constructor & Destructor Documentation

6.80.3.1 future_impl_result()

```
template<typename ... T>
upcxx::detail::future_impl_result< T >::future_impl_result (
    T ... values ) [inline]
```

Definition at line 26 of file impl_result.hpp.

6.80.4 Member Function Documentation

6.80.4.1 ready()

```
template<typename ... T>
bool upcxx::detail::future_impl_result< T >::ready ( ) const [inline]
```

Definition at line 30 of file impl_result.hpp.

6.80.4.2 result_lrefs_getter()

```
template<typename ... T>
upcxx::constant_function<std::tuple<T&...> > upcxx::detail::future_impl_result< T >::result←
_lrefs_getter ( ) const [inline]
```

Definition at line 32 of file impl_result.hpp.

6.80.4.3 result_rvals()

```
template<typename ... T>
auto upcxx::detail::future_impl_result< T >::result_rvals ( ) -> decltype(upcxx::tuple_↵
rvals(results_)) [inline]
```

Definition at line 38 of file impl_result.hpp.

6.80.4.4 steal_header()

```
template<typename ... T>
future_header* upcxx::detail::future_impl_result< T >::steal_header ( ) [inline]
```

Definition at line 45 of file impl_result.hpp.

6.80.5 Member Data Documentation

6.80.5.1 results_

```
template<typename ... T>
std::tuple<T...> upcxx::detail::future_impl_result< T >::results_
```

Definition at line 23 of file impl_result.hpp.

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

- [future/core.hpp](#)
- [future/impl_result.hpp](#)

6.81 upcxx::detail::future_impl_result<> Struct Template Reference

```
#include <impl_result.hpp>
```

Public Types

- typedef [future_header_ops_result_ready_header_ops](#)

Public Member Functions

- bool [ready](#) () const
- [upcxx::constant_function](#)< std::tuple<> > [result_lrefs_getter](#) () const
- std::tuple [result_rvals](#) ()
- [future_header](#) * [steal_header](#) ()

6.81.1 Detailed Description

```
template<>
struct upcxx::detail::future_impl_result<>
```

Definition at line 51 of file impl_result.hpp.

6.81.2 Member Typedef Documentation

6.81.2.1 header_ops

```
typedef future_header_ops_result_ready upcxx::detail::future_impl_result<>::header_ops
```

Definition at line 60 of file impl_result.hpp.

6.81.3 Member Function Documentation

6.81.3.1 ready()

```
bool upcxx::detail::future_impl_result<>::ready ( ) const [inline]
```

Definition at line 52 of file impl_result.hpp.

6.81.3.2 result_lrefs_getter()

```
upcxx::constant_function<std::tuple<> > upcxx::detail::future_impl_result<>::result_lrefs_↵
getter ( ) const [inline]
```

Definition at line 54 of file impl_result.hpp.

6.81.3.3 result_rvals()

```
std::tuple upcxx::detail::future_impl_result<>::result_rvals ( ) [inline]
```

Definition at line 58 of file impl_result.hpp.

6.81.3.4 steal_header()

```
future_header* upcxx::detail::future_impl_result<>::steal_header ( ) [inline]
```

Definition at line 62 of file impl_result.hpp.

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

- [future/impl_result.hpp](#)

6.82 upcxx::detail::future_impl_shref< HeaderOps, T > Struct Template Reference

```
#include <core.hpp>
```

Public Types

- typedef HeaderOps [header_ops](#)

Public Member Functions

- [future_impl_shref](#) ()
- template<typename Header >
[future_impl_shref](#) (Header *hdr)
- [future_impl_shref](#) (const [future_impl_shref](#) &that)
- template<typename Ops1 >
[future_impl_shref](#) (const [future_impl_shref](#)< Ops1, T... > &that, typename std::enable_if< std::is_base_of< HeaderOps, Ops1 >::value >::type !=0)
- [future_impl_shref](#) & [operator=](#) (const [future_impl_shref](#) &that)
- template<typename HeaderOps1 >
std::enable_if< std::is_base_of< HeaderOps, HeaderOps1 >::value, [future_impl_shref](#) &>::type [operator=](#) (const [future_impl_shref](#)< HeaderOps1, T... > &that)
- [future_impl_shref](#) ([future_impl_shref](#) &&that)
- template<typename Ops1 >
[future_impl_shref](#) ([future_impl_shref](#)< Ops1, T... > &&that, typename std::enable_if< std::is_base_of< HeaderOps, Ops1 >::value >::type !=0)
- [future_impl_shref](#) & [operator=](#) ([future_impl_shref](#) &&that)
- template<typename Impl >
[future_impl_shref](#) (Impl &&that)
- [~future_impl_shref](#) ()
- bool [ready](#) () const
- [upcxx::constant_function](#)< std::tuple< T &... > > [result_lrefs_getter](#) () const
- auto [result_rvals](#) () -> decltype([upcxx::tuple_rvals](#)([future_header_result](#)< T... >::results_of(hdr_->result_)))
- [future_header](#) * [steal_header](#) ()

Public Attributes

- [future_header](#) * [hdr_](#)

6.82.1 Detailed Description

```
template<typename HeaderOps, typename ... T>  
struct upcxx::detail::future_impl_shref< HeaderOps, T >
```

Definition at line 36 of file core.hpp.

6.82.2 Member Typedef Documentation

6.82.2.1 header_ops

```
template<typename HeaderOps, typename ... T>  
typedef HeaderOps upcxx::detail::future_impl_shref< HeaderOps, T >::header_ops
```

Definition at line 159 of file impl_shref.hpp.

6.82.3 Constructor & Destructor Documentation

6.82.3.1 future_impl_shref() [1/7]

```
template<typename HeaderOps, typename ... T>  
upcxx::detail::future_impl_shref< HeaderOps, T >::future_impl_shref ( ) [inline]
```

Definition at line 27 of file impl_shref.hpp.

6.82.3.2 future_impl_shref() [2/7]

```
template<typename HeaderOps, typename ... T>  
template<typename Header >  
upcxx::detail::future_impl_shref< HeaderOps, T >::future_impl_shref (   
    Header * hdr ) [inline]
```

Definition at line 32 of file impl_shref.hpp.

6.82.3.3 future_impl_shref() [3/7]

```
template<typename HeaderOps, typename ... T>
upcxx::detail::future_impl_shref< HeaderOps, T >::future_impl_shref (
    const future_impl_shref< HeaderOps, T > & that ) [inline]
```

Definition at line 36 of file impl_shref.hpp.

6.82.3.4 future_impl_shref() [4/7]

```
template<typename HeaderOps, typename ... T>
template<typename Ops1 >
upcxx::detail::future_impl_shref< HeaderOps, T >::future_impl_shref (
    const future_impl_shref< Ops1, T... > & that,
    typename std::enable_if< std::is_base_of< HeaderOps, Ops1 >::value >::type * =
0 ) [inline]
```

Definition at line 41 of file impl_shref.hpp.

6.82.3.5 future_impl_shref() [5/7]

```
template<typename HeaderOps, typename ... T>
upcxx::detail::future_impl_shref< HeaderOps, T >::future_impl_shref (
    future_impl_shref< HeaderOps, T > && that ) [inline]
```

Definition at line 102 of file impl_shref.hpp.

6.82.3.6 future_impl_shref() [6/7]

```
template<typename HeaderOps, typename ... T>
template<typename Ops1 >
upcxx::detail::future_impl_shref< HeaderOps, T >::future_impl_shref (
    future_impl_shref< Ops1, T... > && that,
    typename std::enable_if< std::is_base_of< HeaderOps, Ops1 >::value >::type * =
0 ) [inline]
```

Definition at line 107 of file impl_shref.hpp.

6.82.3.7 future_impl_shref() [7/7]

```
template<typename HeaderOps, typename ... T>
template<typename Impl >
upcxx::detail::future_impl_shref< HeaderOps, T >::future_impl_shref (
    Impl && that ) [inline]
```

Definition at line 124 of file impl_shref.hpp.

6.82.3.8 ~future_impl_shref()

```
template<typename HeaderOps, typename ... T>
upcxx::detail::future_impl_shref< HeaderOps, T >::~~future_impl_shref ( ) [inline]
```

Definition at line 128 of file impl_shref.hpp.

6.82.4 Member Function Documentation

6.82.4.1 operator=() [1/3]

```
template<typename HeaderOps, typename ... T>
future_impl_shref& upcxx::detail::future_impl_shref< HeaderOps, T >::operator= (
    const future_impl_shref< HeaderOps, T > & that ) [inline]
```

Definition at line 49 of file impl_shref.hpp.

6.82.4.2 operator=() [2/3]

```
template<typename HeaderOps, typename ... T>
template<typename HeaderOps1 >
std::enable_if< std::is_base_of<HeaderOps,HeaderOps1>::value, future_impl_shref& >::type
upcxx::detail::future_impl_shref< HeaderOps, T >::operator= (
    const future_impl_shref< HeaderOps1, T... > & that ) [inline]
```

Definition at line 78 of file impl_shref.hpp.

6.82.4.3 operator=() [3/3]

```
template<typename HeaderOps, typename ... T>
future_impl_shref& upcxx::detail::future_impl_shref< HeaderOps, T >::operator= (
    future_impl_shref< HeaderOps, T > && that ) [inline]
```

Definition at line 115 of file impl_shref.hpp.

6.82.4.4 ready()

```
template<typename HeaderOps, typename ... T>
bool upcxx::detail::future_impl_shref< HeaderOps, T >::ready ( ) const [inline]
```

Definition at line 136 of file impl_shref.hpp.

6.82.4.5 result_lrefs_getter()

```
template<typename HeaderOps, typename ... T>
upcxx::constant_function<std::tuple<T&...> > upcxx::detail::future_impl_shref< HeaderOps, T
>::result_lrefs_getter ( ) const [inline]
```

Definition at line 140 of file impl_shref.hpp.

6.82.4.6 result_rvals()

```
template<typename HeaderOps, typename ... T>
auto upcxx::detail::future_impl_shref< HeaderOps, T >::result_rvals ( ) -> decltype( upcxx←
::tuple_rvals( future_header_result<T...>::results_of(hdr_->result_) ) ) [inline]
```

Definition at line 148 of file impl_shref.hpp.

6.82.4.7 steal_header()

```
template<typename HeaderOps, typename ... T>
future_header* upcxx::detail::future_impl_shref< HeaderOps, T >::steal_header ( ) [inline]
```

Definition at line 161 of file impl_shref.hpp.

6.82.5 Member Data Documentation

6.82.5.1 hdr_

```
template<typename HeaderOps, typename ... T>
future_header* upcxx::detail::future_impl_shref< HeaderOps, T >::hdr_
```

Definition at line 24 of file impl_shref.hpp.

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

- [future/core.hpp](#)
- [future/impl_shref.hpp](#)

6.83 upcxx::detail::future_impl_when_all< ArgTuple, T > Struct Template Reference

```
#include <core.hpp>
```

6.83.1 Detailed Description

```
template<typename ArgTuple, typename ... T>
struct upcxx::detail::future_impl_when_all< ArgTuple, T >
```

Definition at line 40 of file `core.hpp`.

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

- [future/core.hpp](#)

6.84 `upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >` Struct Template Reference

```
#include <impl_when_all.hpp>
```

Public Types

- typedef [future_header_ops_general_header_ops](#)

Public Member Functions

- [future_impl_when_all](#) (FuArg ...args)
- bool [ready](#) () const
- auto [result_lrefs_getter](#) () const -> decltype(this->result_lrefs_getter_(upcxx::make_index_sequence< sizeof...(FuArg)>()))
- auto [result_rvals](#) () -> decltype(this->result_rvals_(upcxx::make_index_sequence< sizeof...(FuArg)>()))
- [future_header](#) * [steal_header](#) ()

Public Attributes

- `std::tuple< FuArg... >` [args_](#)

6.84.1 Detailed Description

```
template<typename ... FuArg, typename ... T>
struct upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >
```

Definition at line 27 of file `impl_when_all.hpp`.

6.84.2 Member Typedef Documentation

6.84.2.1 header_ops

```
template<typename ... FuArg, typename ... T>
typedef future_header_ops_general upcxx::detail::future_impl_when_all< std::tuple< FuArg...
>, T... >::header_ops
```

Definition at line 99 of file impl_when_all.hpp.

6.84.3 Constructor & Destructor Documentation

6.84.3.1 future_impl_when_all()

```
template<typename ... FuArg, typename ... T>
upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >::future_impl_when_all (
    FuArg ... args ) [inline]
```

Definition at line 75 of file impl_when_all.hpp.

6.84.4 Member Function Documentation

6.84.4.1 ready()

```
template<typename ... FuArg, typename ... T>
bool upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >::ready ( ) const
[inline]
```

Definition at line 79 of file impl_when_all.hpp.

6.84.4.2 result_lrefs_getter()

```
template<typename ... FuArg, typename ... T>
auto upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >::result_lrefs_getter ( ) const -> decltype(this->result_lrefs_getter_(upcxx::make_index_sequence<sizeof...(FuArg)>())) [inline]
```

Definition at line 83 of file impl_when_all.hpp.

6.84.4.3 result_rvals()

```
template<typename ... FuArg, typename ... T>
auto upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >::result_rvals ( )
-> decltype(this->result_rvals_(upcxx::make_index_sequence<sizeof...(FuArg)>())) [inline]
```

Definition at line 88 of file impl_when_all.hpp.

6.84.4.4 steal_header()

```
template<typename ... FuArg, typename ... T>
future_header* upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >::steal←
_header ( ) [inline]
```

Definition at line 101 of file impl_when_all.hpp.

6.84.5 Member Data Documentation

6.84.5.1 args_

```
template<typename ... FuArg, typename ... T>
std::tuple<FuArg...> upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >←
::args_
```

Definition at line 28 of file impl_when_all.hpp.

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

- [future/impl_when_all.hpp](#)

6.85 upcxx::future_is_trivially_ready< FutureOrKind > Struct Template Reference

```
#include <core.hpp>
```

Static Public Attributes

- static constexpr bool [value](#) = false

6.85.1 Detailed Description

```
template<typename FutureOrKind>
struct upcxx::future_is_trivially_ready< FutureOrKind >
```

Definition at line 90 of file core.hpp.

6.85.2 Member Data Documentation

6.85.2.1 value

```
template<typename FutureOrKind>
constexpr bool upcxx::future_is_trivially_ready< FutureOrKind >::value = false [static]
```

Definition at line 91 of file core.hpp.

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

- [future/core.hpp](#)

6.86 upcxx::future_is_trivially_ready< future1< detail::future_kind_mapped< Arg, Fn >, T... > > Struct Template Reference

```
#include <impl_mapped.hpp>
```

Static Public Attributes

- static constexpr bool [value](#) = [future_is_trivially_ready<Arg>::value](#)

6.86.1 Detailed Description

```
template<typename Arg, typename Fn, typename ... T>
struct upcxx::future_is_trivially_ready< future1< detail::future_kind_mapped< Arg, Fn >, T... > >
```

Definition at line 13 of file impl_mapped.hpp.

6.86.2 Member Data Documentation

6.86.2.1 value

```
template<typename Arg , typename Fn , typename ... T>
constexpr bool upcxx::future_is_trivially_ready< future1< detail::future_kind_mapped< Arg, Fn >, T... > >::value = future_is_trivially_ready<Arg>::value [static]
```

Definition at line 16 of file impl_mapped.hpp.

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

- [future/impl_mapped.hpp](#)

6.87 `upcxx::future_is_trivially_ready< future1< detail::future_kind_result, T... > >` Struct Template Reference

```
#include <impl_result.hpp>
```

Static Public Attributes

- static constexpr bool `value` = true

6.87.1 Detailed Description

```
template<typename ... T>  
struct upcxx::future_is_trivially_ready< future1< detail::future_kind_result, T... > >
```

Definition at line 11 of file `impl_result.hpp`.

6.87.2 Member Data Documentation

6.87.2.1 `value`

```
template<typename ... T>  
constexpr bool upcxx::future_is_trivially_ready< future1< detail::future_kind_result, T... >  
>::value = true [static]
```

Definition at line 14 of file `impl_result.hpp`.

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

- [future/impl_result.hpp](#)

6.88 `upcxx::future_is_trivially_ready< future1< detail::future_kind_shref< HeaderOps >, T... > >` Struct Template Reference

```
#include <impl_shref.hpp>
```

Static Public Attributes

- static constexpr bool `value` = `HeaderOps::is_trivially_ready_result`

6.88.1 Detailed Description

```
template<typename HeaderOps, typename ... T>
struct upcxx::future_is_trivially_ready< future1< detail::future_kind_shref< HeaderOps >, T... > >
```

Definition at line 11 of file impl_shref.hpp.

6.88.2 Member Data Documentation

6.88.2.1 value

```
template<typename HeaderOps , typename ... T>
constexpr bool upcxx::future_is_trivially_ready< future1< detail::future_kind_shref< HeaderOps >, T... > >::value = HeaderOps::is_trivially_ready_result [static]
```

Definition at line 14 of file impl_shref.hpp.

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

- [future/impl_shref.hpp](#)

6.89 upcxx::future_is_trivially_ready< future1< detail::future_kind_when_all< Arg... >, T... > > Struct Template Reference

```
#include <impl_when_all.hpp>
```

Static Public Attributes

- static constexpr bool [value](#) = [upcxx::trait_forall<upcxx::future_is_trivially_ready, Arg...>::value](#)

6.89.1 Detailed Description

```
template<typename ... Arg, typename ... T>
struct upcxx::future_is_trivially_ready< future1< detail::future_kind_when_all< Arg... >, T... > >
```

Definition at line 12 of file impl_when_all.hpp.

6.89.2 Member Data Documentation

6.89.2.1 value

```
template<typename ... Arg, typename ... T>
constexpr bool upcxx::future_is_trivially_ready< future1< detail::future_kind_when_all< Arg...
>, T... > >::value = upcxx::trait_forall<upcxx::future_is_trivially_ready, Arg...>::value
[static]
```

Definition at line 15 of file `impl_when_all.hpp`.

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

- [future/impl_when_all.hpp](#)

6.90 `upcxx::detail::future_kind_mapped< FuArg, Fn >` Struct Template Reference

```
#include <core.hpp>
```

Public Types

- `template<typename ... T>`
using `with_types = future_impl_mapped< FuArg, Fn, T... >`

6.90.1 Detailed Description

```
template<typename FuArg, typename Fn>
struct upcxx::detail::future_kind_mapped< FuArg, Fn >
```

Definition at line 63 of file `core.hpp`.

6.90.2 Member Typedef Documentation

6.90.2.1 `with_types`

```
template<typename FuArg , typename Fn >
template<typename ... T>
using upcxx::detail::future_kind_mapped< FuArg, Fn >::with_types = future_impl_mapped<Fu↔
Arg,Fn,T...>
```

Definition at line 65 of file `core.hpp`.

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

- [future/core.hpp](#)

6.91 `upcxx::detail::future_kind_result` Struct Reference

```
#include <core.hpp>
```

Public Types

- `template<typename ... T>`
using `with_types` = `future_impl_result< T... >`

6.91.1 Detailed Description

Definition at line 51 of file `core.hpp`.

6.91.2 Member Typedef Documentation

6.91.2.1 `with_types`

```
template<typename ... T>
using upcxx::detail::future_kind_result::with_types = future_impl_result<T...>
```

Definition at line 53 of file `core.hpp`.

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

- `future/core.hpp`

6.92 `upcxx::detail::future_kind_shref< HeaderOps >` Struct Template Reference

```
#include <core.hpp>
```

Public Types

- `template<typename ... T>`
using `with_types` = `future_impl_shref< HeaderOps, T... >`

6.92.1 Detailed Description

```
template<typename HeaderOps>
struct upcxx::detail::future_kind_shref< HeaderOps >
```

Definition at line 46 of file `core.hpp`.

6.92.2 Member Typedef Documentation

6.92.2.1 `with_types`

```
template<typename HeaderOps>
template<typename ... T>
using upcxx::detail::future_kind_shref< HeaderOps >::with_types = future_impl_shref<HeaderOps, T...>
```

Definition at line 48 of file `core.hpp`.

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

- [future/core.hpp](#)

6.93 `upcxx::detail::future_kind_when_all< FuArg >` Struct Template Reference

```
#include <core.hpp>
```

Public Types

- `template<typename ... T>`
using `with_types` = `future_impl_when_all< std::tuple< FuArg... >, T... >`

6.93.1 Detailed Description

```
template<typename ... FuArg>
struct upcxx::detail::future_kind_when_all< FuArg >
```

Definition at line 57 of file `core.hpp`.

6.93.2 Member Typedef Documentation

6.93.2.1 `with_types`

```
template<typename ... FuArg>
template<typename ... T>
using upcxx::detail::future_kind_when_all< FuArg >::with_types = future_impl_when_all<std::tuple<FuArg...>, T...>
```

Definition at line 59 of file `core.hpp`.

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

- [future/core.hpp](#)

6.94 `upcxx::detail::future_then`< Arg, Fn, FnRet, arg_trivial > Struct Template Reference

```
#include <core.hpp>
```

6.94.1 Detailed Description

```
template<typename Arg, typename Fn, typename FnRet = apply_futured_as_future_return_t<Fn(Arg)>, bool arg_trivial =
future_is_trivially_ready<Arg>::value>
struct upcxx::detail::future_then< Arg, Fn, FnRet, arg_trivial >
```

Definition at line 144 of file core.hpp.

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

- [future/core.hpp](#)

6.95 `upcxx::detail::future_then`< Arg, Fn, future1< FnRetKind, FnRetT... >, false > Struct Template Reference

```
#include <then.hpp>
```

Public Member Functions

- `template<typename Arg1, typename Fn1 >`
[future1< future_kind_shref< future_header_ops_general >, FnRetT... >](#) `operator()` (Arg1 &&arg, Fn1 &&fn)

6.95.1 Detailed Description

```
template<typename Arg, typename Fn, typename FnRetKind, typename ... FnRetT>
struct upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, false >
```

Definition at line 127 of file then.hpp.

6.95.2 Member Function Documentation

6.95.2.1 operator()

```
template<typename Arg , typename Fn , typename FnRetKind , typename ... FnRetT>
template<typename Arg1 , typename Fn1 >
future1<future_kind_shref<future_header_ops_general>, FnRetT...> upcxx::detail::future_then<
Arg, Fn, future1< FnRetKind, FnRetT... >, false >::operator() (
    Arg1 && arg,
    Fn1 && fn ) [inline]
```

Definition at line 134 of file then.hpp.

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

- [future/then.hpp](#)

6.96 upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, true > Struct Template Reference

```
#include <then.hpp>
```

Public Member Functions

- `template<typename Arg1 , typename Fn1 >`
`future1< FnRetKind, FnRetT... > operator()` (Arg1 &&arg, Fn1 &&fn)

6.96.1 Detailed Description

```
template<typename Arg, typename Fn, typename FnRetKind, typename ... FnRetT>
struct upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, true >
```

Definition at line 161 of file then.hpp.

6.96.2 Member Function Documentation

6.96.2.1 operator()

```
template<typename Arg , typename Fn , typename FnRetKind , typename ... FnRetT>
template<typename Arg1 , typename Fn1 >
future1<FnRetKind,FnRetT...> upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, true >::operator() (
    Arg1 && arg,
    Fn1 && fn ) [inline]
```

Definition at line 168 of file then.hpp.

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

- [future/then.hpp](#)

6.97 `upcxx::detail::future_then_pure< Arg, Fn, FnRet, arg_trivial, fnret_trivial >` Struct Template Reference

```
#include <core.hpp>
```

6.97.1 Detailed Description

```
template<typename Arg, typename Fn, typename FnRet = apply_futured_as_future_return_t<Fn(Arg)>, bool arg_trivial =
future_is_trivially_ready<Arg>::value, bool fnret_trivial = future_is_trivially_ready<FnRet>::value>
struct upcxx::detail::future_then_pure< Arg, Fn, FnRet, arg_trivial, fnret_trivial >
```

Definition at line 151 of file core.hpp.

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

- [future/core.hpp](#)

6.98 `upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, false >` Struct Template Reference

```
#include <then.hpp>
```

Public Member Functions

- `template<typename Arg1, typename Fn1 > future1< future_kind_shref< future_header_ops_general >, FnRetT... > operator() (Arg1 &&arg, Fn1 &&fn)`

6.98.1 Detailed Description

```
template<typename Arg, typename Fn, typename FnRetKind, typename ... FnRetT>
struct upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, false >
```

Definition at line 212 of file then.hpp.

6.98.2 Member Function Documentation

6.98.2.1 operator>()

```
template<typename Arg , typename Fn , typename FnRetKind , typename ... FnRetT>
template<typename Arg1 , typename Fn1 >
future1<future_kind_shref<future_header_ops_general>, FnRetT...> upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, false >::operator() (
    Arg1 && arg,
    Fn1 && fn ) [inline]
```

Definition at line 220 of file then.hpp.

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

- [future/then.hpp](#)

6.99 upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, true > Struct Template Reference

```
#include <then.hpp>
```

Public Member Functions

- `template<typename Arg1 , typename Fn1 > future1< future_kind_mapped< Arg, Fn >, FnRetT... > operator() (Arg1 &&arg, Fn1 &&fn)`

6.99.1 Detailed Description

```
template<typename Arg, typename Fn, typename FnRetKind, typename ... FnRetT>
struct upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, true >
```

Definition at line 258 of file then.hpp.

6.99.2 Member Function Documentation

6.99.2.1 operator>()

```
template<typename Arg , typename Fn , typename FnRetKind , typename ... FnRetT>
template<typename Arg1 , typename Fn1 >
future1<future_kind_mapped<Arg,Fn>, FnRetT...> upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, true >::operator() (
    Arg1 && arg,
    Fn1 && fn ) [inline]
```

Definition at line 265 of file then.hpp.

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

- [future/then.hpp](#)

6.100 upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, true, fnret_trivial > Struct Template Reference

```
#include <then.hpp>
```

Public Member Functions

- `template<typename Arg1 , typename Fn1 >`
`future1< FnRetKind, FnRetT... >` `operator()` (Arg &&arg, Fn1 &&fn)

6.100.1 Detailed Description

```
template<typename Arg, typename Fn, typename FnRetKind, typename ... FnRetT, bool fnret_trivial>
struct upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, true, fnret_trivial >
```

Definition at line 243 of file `then.hpp`.

6.100.2 Member Function Documentation

6.100.2.1 operator()

```
template<typename Arg , typename Fn , typename FnRetKind , typename ... FnRetT, bool fnret_↵
trivial>
template<typename Arg1 , typename Fn1 >
future1<FnRetKind, FnRetT...> upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind,
FnRetT... >, true, fnret_trivial >::operator() (
    Arg && arg,
    Fn1 && fn ) [inline]
```

Definition at line 249 of file `then.hpp`.

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

- [future/then.hpp](#)

6.101 upcxx::global_ptr< T > Class Template Reference

```
#include <global_ptr.hpp>
```

Public Types

- using `element_type` = T

Public Member Functions

- [global_ptr](#) (std::nullptr_t nil=nullptr)
- [global_ptr](#) (T *ptr)
- [global_ptr](#) (detail::global_ptr_ctor_internal, intrank_t rank, T *raw)
- bool [is_local](#) () const
- bool [is_null](#) () const
- T * [local](#) () const
- [intrank_t](#) [where](#) () const
- [global_ptr](#) [operator+](#) (std::ptrdiff_t diff) const
- [global_ptr](#) [operator-](#) (std::ptrdiff_t diff) const
- std::ptrdiff_t [operator-](#) ([global_ptr](#) rhs) const
- [global_ptr](#) & [operator++](#) ()
- [global_ptr](#) [operator++](#) (int)
- [global_ptr](#) & [operator--](#) ()
- [global_ptr](#) [operator--](#) (int)
- bool [operator==](#) ([global_ptr](#) rhs) const
- bool [operator!=](#) ([global_ptr](#) rhs) const
- bool [operator<](#) ([global_ptr](#) rhs) const
- bool [operator<=](#) ([global_ptr](#) rhs) const
- bool [operator>](#) ([global_ptr](#) rhs) const
- bool [operator>=](#) ([global_ptr](#) rhs) const

Public Attributes

- [intrank_t](#) [rank_](#)
- T * [raw_ptr_](#)

Friends

- struct [std::less](#)< [global_ptr](#)< T > >
- struct [std::less_equal](#)< [global_ptr](#)< T > >
- struct [std::greater](#)< [global_ptr](#)< T > >
- struct [std::greater_equal](#)< [global_ptr](#)< T > >
- struct [std::hash](#)< [global_ptr](#)< T > >
- template<typename U , typename V >
[global_ptr](#)< U > [reinterpret_pointer_cast](#) ([global_ptr](#)< V > ptr)
- template<typename U >
std::ostream & [operator<<](#) (std::ostream &os, [global_ptr](#)< U > ptr)

6.101.1 Detailed Description

```
template<typename T>
class upcxx::global_ptr< T >
```

Definition at line 43 of file [global_ptr.hpp](#).

6.101.2 Member Typedef Documentation

6.101.2.1 element_type

```
template<typename T>
using upcxx::global_ptr< T >::element_type = T
```

Definition at line 48 of file global_ptr.hpp.

6.101.3 Constructor & Destructor Documentation

6.101.3.1 global_ptr() [1/3]

```
template<typename T>
upcxx::global_ptr< T >::global_ptr (
    std::nullptr_t nil = nullptr ) [inline]
```

Definition at line 51 of file global_ptr.hpp.

6.101.3.2 global_ptr() [2/3]

```
template<typename T>
upcxx::global_ptr< T >::global_ptr (
    T * ptr ) [inline], [explicit]
```

Definition at line 56 of file global_ptr.hpp.

6.101.3.3 global_ptr() [3/3]

```
template<typename T>
upcxx::global_ptr< T >::global_ptr (
    detail::global_ptr_ctor_internal ,
    intrank_t rank,
    T * raw ) [inline], [explicit]
```

Definition at line 66 of file global_ptr.hpp.

6.101.4 Member Function Documentation

6.101.4.1 is_local()

```
template<typename T>
bool upcxx::global_ptr< T >::is_local ( ) const [inline]
```

Definition at line 74 of file global_ptr.hpp.

6.101.4.2 is_null()

```
template<typename T>
bool upcxx::global_ptr< T >::is_null ( ) const [inline]
```

Definition at line 78 of file global_ptr.hpp.

6.101.4.3 local()

```
template<typename T>
T* upcxx::global_ptr< T >::local ( ) const [inline]
```

Definition at line 82 of file global_ptr.hpp.

6.101.4.4 operator!=(())

```
template<typename T>
bool upcxx::global_ptr< T >::operator!= (
    global_ptr< T > rhs ) const [inline]
```

Definition at line 128 of file global_ptr.hpp.

6.101.4.5 operator+()

```
template<typename T>
global_ptr upcxx::global_ptr< T >::operator+ (
    std::ptrdiff_t diff ) const [inline]
```

Definition at line 90 of file global_ptr.hpp.

6.101.4.6 operator++() [1/2]

```
template<typename T>
global_ptr& upcxx::global_ptr< T >::operator++ ( ) [inline]
```

Definition at line 104 of file global_ptr.hpp.

6.101.4.7 operator++() [2/2]

```
template<typename T>
global_ptr upcxx::global_ptr< T >::operator++ (
    int ) [inline]
```

Definition at line 108 of file global_ptr.hpp.

6.101.4.8 operator-() [1/2]

```
template<typename T>
global_ptr upcxx::global_ptr< T >::operator- (
    std::ptrdiff_t diff ) const [inline]
```

Definition at line 94 of file global_ptr.hpp.

6.101.4.9 operator-() [2/2]

```
template<typename T>
std::ptrdiff_t upcxx::global_ptr< T >::operator- (
    global_ptr< T > rhs ) const [inline]
```

Definition at line 98 of file global_ptr.hpp.

6.101.4.10 operator--() [1/2]

```
template<typename T>
global_ptr& upcxx::global_ptr< T >::operator-- ( ) [inline]
```

Definition at line 114 of file global_ptr.hpp.

6.101.4.11 operator--() [2/2]

```
template<typename T>
global_ptr upcxx::global_ptr< T >::operator-- (
    int ) [inline]
```

Definition at line 118 of file global_ptr.hpp.

6.101.4.12 operator<()

```
template<typename T>
bool upcxx::global_ptr< T >::operator< (
    global_ptr< T > rhs ) const [inline]
```

Definition at line 133 of file global_ptr.hpp.

6.101.4.13 operator<=()

```
template<typename T>
bool upcxx::global_ptr< T >::operator<= (
    global_ptr< T > rhs ) const [inline]
```

Definition at line 137 of file global_ptr.hpp.

6.101.4.14 operator==(())

```
template<typename T>
bool upcxx::global_ptr< T >::operator==(
    global_ptr< T > rhs ) const [inline]
```

Definition at line 124 of file global_ptr.hpp.

6.101.4.15 operator>()

```
template<typename T>
bool upcxx::global_ptr< T >::operator> (
    global_ptr< T > rhs ) const [inline]
```

Definition at line 141 of file global_ptr.hpp.

6.101.4.16 operator>=()

```
template<typename T>
bool upcxx::global_ptr< T >::operator>= (
    global_ptr< T > rhs ) const [inline]
```

Definition at line 145 of file global_ptr.hpp.

6.101.4.17 where()

```
template<typename T>
inrank_t upcxx::global_ptr< T >::where ( ) const [inline]
```

Definition at line 86 of file global_ptr.hpp.

6.101.5 Friends And Related Function Documentation

6.101.5.1 operator<<

```
template<typename T>
template<typename U >
std::ostream& operator<< (
    std::ostream & os,
    global_ptr< U > ptr ) [friend]
```

6.101.5.2 reinterpret_pointer_cast

```
template<typename T>
template<typename U , typename V >
global_ptr<U> reinterpret_pointer_cast (
    global_ptr< V > ptr ) [friend]
```

6.101.5.3 std::greater< global_ptr< T > >

```
template<typename T>
friend struct std::greater< global_ptr< T > > [friend]
```

Definition at line 152 of file global_ptr.hpp.

6.101.5.4 std::greater_equal< global_ptr< T > >

```
template<typename T>
friend struct std::greater_equal< global_ptr< T > > [friend]
```

Definition at line 153 of file global_ptr.hpp.

6.101.5.5 std::hash< global_ptr< T > >

```
template<typename T>
friend struct std::hash< global_ptr< T > > [friend]
```

Definition at line 154 of file global_ptr.hpp.

6.101.5.6 std::less< global_ptr< T > >

```
template<typename T>
friend struct std::less< global_ptr< T > > [friend]
```

Definition at line 150 of file global_ptr.hpp.

6.101.5.7 std::less_equal< global_ptr< T > >

```
template<typename T>
friend struct std::less_equal< global_ptr< T > > [friend]
```

Definition at line 151 of file global_ptr.hpp.

6.101.6 Member Data Documentation

6.101.6.1 rank_

```
template<typename T>
inrank_t upcxx::global_ptr< T >::rank_
```

Definition at line 166 of file global_ptr.hpp.

6.101.6.2 raw_ptr_

```
template<typename T>
T* upcxx::global_ptr< T >::raw_ptr_
```

Definition at line 167 of file global_ptr.hpp.

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

- [global_ptr.hpp](#)

6.102 upcxx::detail::global_ptr_ctor_internal Struct Reference

```
#include <global_ptr.hpp>
```

6.102.1 Detailed Description

Definition at line 38 of file global_ptr.hpp.

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

- [global_ptr.hpp](#)

6.103 std::greater< upcxx::global_ptr< T > > Struct Template Reference

```
#include <global_ptr.hpp>
```

Public Member Functions

- constexpr bool [operator\(\)](#) (upcxx::global_ptr< T > lhs, upcxx::global_ptr< T > rhs) const

6.103.1 Detailed Description

```
template<typename T>
struct std::greater< upcxx::global_ptr< T > >
```

Definition at line 209 of file global_ptr.hpp.

6.103.2 Member Function Documentation

6.103.2.1 operator()

```
template<typename T >
constexpr bool std::greater< upcxx::global_ptr< T > >::operator() (
    upcxx::global_ptr< T > lhs,
    upcxx::global_ptr< T > rhs ) const [inline]
```

Definition at line 210 of file global_ptr.hpp.

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

- [global_ptr.hpp](#)

6.104 std::greater_equal< upcxx::global_ptr< T > > Struct Template Reference

```
#include <global_ptr.hpp>
```

Public Member Functions

- constexpr bool [operator\(\)](#) ([upcxx::global_ptr< T > lhs](#), [upcxx::global_ptr< T > rhs](#)) const

6.104.1 Detailed Description

```
template<typename T>
struct std::greater_equal< upcxx::global_ptr< T > >
```

Definition at line 218 of file global_ptr.hpp.

6.104.2 Member Function Documentation

6.104.2.1 operator()

```
template<typename T >
constexpr bool std::greater_equal< upcxx::global_ptr< T > >::operator() (
    upcxx::global_ptr< T > lhs,
    upcxx::global_ptr< T > rhs ) const [inline]
```

Definition at line 219 of file global_ptr.hpp.

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

- [global_ptr.hpp](#)

6.105 `std::hash< upcxx::digest >` Struct Template Reference

```
#include <digest.hpp>
```

Public Member Functions

- `size_t operator() (upcxx::digest x) const`

6.105.1 Detailed Description

```
template<>  
struct std::hash< upcxx::digest >
```

Definition at line 38 of file `digest.hpp`.

6.105.2 Member Function Documentation

6.105.2.1 `operator()`

```
size_t std::hash< upcxx::digest >::operator() (  
    upcxx::digest x ) const [inline]
```

Definition at line 39 of file `digest.hpp`.

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

- [digest.hpp](#)

6.106 `std::hash< upcxx::dist_id< T > >` Struct Template Reference

```
#include <dist_object.hpp>
```

Public Member Functions

- `size_t operator() (upcxx::dist_id< T > id) const`

6.106.1 Detailed Description

```
template<typename T>  
struct std::hash< upcxx::dist_id< T > >
```

Definition at line 85 of file `dist_object.hpp`.

6.106.2 Member Function Documentation

6.106.2.1 operator()

```
template<typename T >
size_t std::hash< upcxx::dist_id< T > >::operator() (
    upcxx::dist_id< T > id ) const [inline]
```

Definition at line 86 of file dist_object.hpp.

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

- [dist_object.hpp](#)

6.107 std::hash< upcxx::global_ptr< T > > Struct Template Reference

```
#include <global_ptr.hpp>
```

Public Member Functions

- `std::size_t operator() (upcxx::global_ptr< T > gptr) const`

6.107.1 Detailed Description

```
template<typename T>
struct std::hash< upcxx::global_ptr< T > >
```

Definition at line 227 of file global_ptr.hpp.

6.107.2 Member Function Documentation

6.107.2.1 operator()

```
template<typename T >
std::size_t std::hash< upcxx::global_ptr< T > >::operator() (
    upcxx::global_ptr< T > gptr ) const [inline]
```

Utilities derived from Boost, subject to the following license:

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Definition at line 228 of file `global_ptr.hpp`.

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

- [global_ptr.hpp](#)

6.108 upcxx::hasher_reflector< Hasher > Struct Template Reference

```
#include <reflection.hpp>
```

Public Member Functions

- `template<typename T > void operator() (const T &x)`
- `template<typename T > void opaque (const T &x)`

Public Attributes

- `Hasher & h`

6.108.1 Detailed Description

```
template<typename Hasher>
struct upcxx::hasher_reflector< Hasher >
```

Definition at line 104 of file reflection.hpp.

6.108.2 Member Function Documentation

6.108.2.1 opaque()

```
template<typename Hasher >
template<typename T >
void upcxx::hasher_reflector< Hasher >::opaque (
    const T & x ) [inline]
```

Definition at line 113 of file reflection.hpp.

6.108.2.2 operator()

```
template<typename Hasher >
template<typename T >
void upcxx::hasher_reflector< Hasher >::operator() (
    const T & x ) [inline]
```

Definition at line 108 of file reflection.hpp.

6.108.3 Member Data Documentation

6.108.3.1 h

```
template<typename Hasher >
Hasher& upcxx::hasher_reflector< Hasher >::h
```

Definition at line 105 of file reflection.hpp.

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

- [reflection.hpp](#)

6.109 `upcxx::index_sequence<... >` Struct Template Reference

```
#include <utility.hpp>
```

6.109.1 Detailed Description

```
template<int...>  
struct upcxx::index_sequence<... >
```

Definition at line 188 of file `utility.hpp`.

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

- [utility.hpp](#)

6.110 `upcxx::integer_golden_ratio< T >` Struct Template Reference

```
#include <reflection.hpp>
```

Static Public Attributes

- static constexpr T `value` = `integer_golden_ratio_bits<8*sizeof(T)>::value`

6.110.1 Detailed Description

```
template<typename T>  
struct upcxx::integer_golden_ratio< T >
```

Definition at line 152 of file `reflection.hpp`.

6.110.2 Member Data Documentation

6.110.2.1 `value`

```
template<typename T >  
constexpr T upcxx::integer_golden_ratio< T >::value = integer_golden_ratio_bits<8*sizeof(T)>↔  
::value [static]
```

Definition at line 153 of file `reflection.hpp`.

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

- [reflection.hpp](#)

6.111 `upcxx::integer_golden_ratio_bits< bit_n >` Struct Template Reference

```
#include <reflection.hpp>
```

6.111.1 Detailed Description

```
template<int bit_n>  
struct upcxx::integer_golden_ratio_bits< bit_n >
```

Definition at line 142 of file `reflection.hpp`.

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

- [reflection.hpp](#)

6.112 `upcxx::integer_golden_ratio_bits< 32 >` Struct Template Reference

```
#include <reflection.hpp>
```

Static Public Attributes

- static constexpr `std::uint32_t value` = `0x9e3779b1u`

6.112.1 Detailed Description

```
template<>  
struct upcxx::integer_golden_ratio_bits< 32 >
```

Definition at line 144 of file `reflection.hpp`.

6.112.2 Member Data Documentation

6.112.2.1 `value`

```
constexpr std::uint32_t upcxx::integer_golden_ratio_bits< 32 >::value = 0x9e3779b1u [static]
```

Definition at line 145 of file `reflection.hpp`.

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

- [reflection.hpp](#)

6.113 upcxx::integer_golden_ratio_bits< 64 > Struct Template Reference

```
#include <reflection.hpp>
```

Static Public Attributes

- static constexpr std::uint64_t [value](#) = 0x9e3779b97f4a7c15u

6.113.1 Detailed Description

```
template<>
struct upcxx::integer_golden_ratio_bits< 64 >
```

Definition at line 147 of file reflection.hpp.

6.113.2 Member Data Documentation

6.113.2.1 value

```
constexpr std::uint64_t upcxx::integer_golden_ratio_bits< 64 >::value = 0x9e3779b97f4a7c15u
[static]
```

Definition at line 148 of file reflection.hpp.

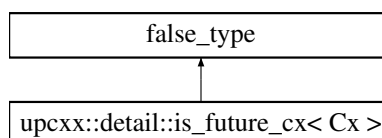
The documentation for this struct was generated from the following file:

- [reflection.hpp](#)

6.114 upcxx::detail::is_future_cx< Cx > Struct Template Reference

```
#include <completion.hpp>
```

Inheritance diagram for upcxx::detail::is_future_cx< Cx >:



6.114.1 Detailed Description

```
template<typename Cx>
struct upcxx::detail::is_future_cx< Cx >
```

Definition at line 64 of file completion.hpp.

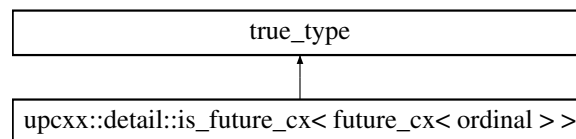
The documentation for this struct was generated from the following file:

- [completion.hpp](#)

6.115 upcxx::detail::is_future_cx< future_cx< ordinal > > Struct Template Reference

```
#include <completion.hpp>
```

Inheritance diagram for upcxx::detail::is_future_cx< future_cx< ordinal > >:



6.115.1 Detailed Description

```
template<int ordinal>
struct upcxx::detail::is_future_cx< future_cx< ordinal > >
```

Definition at line 66 of file completion.hpp.

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

- [completion.hpp](#)

6.116 std::less< upcxx::global_ptr< T > > Struct Template Reference

```
#include <global_ptr.hpp>
```

Public Member Functions

- constexpr bool [operator\(\)](#) (upcxx::global_ptr< T > lhs, upcxx::global_ptr< T > rhs) const

6.116.1 Detailed Description

```
template<typename T>
struct std::less< upcxx::global_ptr< T > >
```

Definition at line 191 of file `global_ptr.hpp`.

6.116.2 Member Function Documentation

6.116.2.1 operator()

```
template<typename T >
constexpr bool std::less< upcxx::global_ptr< T > >::operator() (
    upcxx::global_ptr< T > lhs,
    upcxx::global_ptr< T > rhs ) const [inline]
```

Definition at line 192 of file `global_ptr.hpp`.

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

- [global_ptr.hpp](#)

6.117 std::less_equal< upcxx::global_ptr< T > > Struct Template Reference

```
#include <global_ptr.hpp>
```

Public Member Functions

- constexpr bool [operator\(\)](#) ([upcxx::global_ptr< T > lhs](#), [upcxx::global_ptr< T > rhs](#)) const

6.117.1 Detailed Description

```
template<typename T>
struct std::less_equal< upcxx::global_ptr< T > >
```

Definition at line 200 of file `global_ptr.hpp`.

6.117.2 Member Function Documentation

6.117.2.1 operator()

```
template<typename T >
constexpr bool std::less_equal< upcxx::global_ptr< T > >::operator() (
    upcxx::global_ptr< T > lhs,
    upcxx::global_ptr< T > rhs ) const [inline]
```

Definition at line 201 of file global_ptr.hpp.

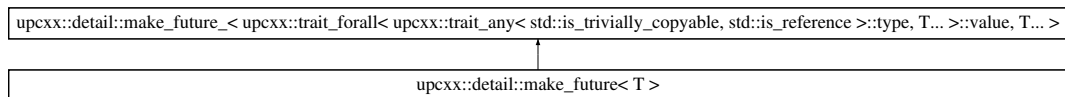
The documentation for this struct was generated from the following file:

- [global_ptr.hpp](#)

6.118 upcxx::detail::make_future< T > Struct Template Reference

```
#include <make_future.hpp>
```

Inheritance diagram for upcxx::detail::make_future< T >:



6.118.1 Detailed Description

```
template<typename ... T>
struct upcxx::detail::make_future< T >
```

Definition at line 40 of file make_future.hpp.

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

- [future/make_future.hpp](#)

6.119 upcxx::detail::make_future_< trivial, T > Struct Template Reference

```
#include <make_future.hpp>
```

6.119.1 Detailed Description

```
template<bool trivial, typename ... T>
struct upcxx::detail::make_future_< trivial, T >
```

Definition at line 13 of file make_future.hpp.

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

- [future/make_future.hpp](#)

6.120 `upcxx::detail::make_future_< false, T... >` Struct Template Reference

```
#include <make_future.hpp>
```

Public Member Functions

- [future1](#)`< future_kind_shref< future_header_ops_result_ready >, T... > operator()` (T ...values)

6.120.1 Detailed Description

```
template<typename ... T>
struct upcxx::detail::make_future_< false, T... >
```

Definition at line 27 of file `make_future.hpp`.

6.120.2 Member Function Documentation

6.120.2.1 `operator()`

```
template<typename ... T>
future1<future_kind_shref<future_header_ops_result_ready>,T...> upcxx::detail::make_future_<
false, T... >::operator() (
    T ... values ) [inline]
```

Definition at line 29 of file `make_future.hpp`.

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

- [future/make_future.hpp](#)

6.121 `upcxx::detail::make_future_< true, T... >` Struct Template Reference

```
#include <make_future.hpp>
```

Public Member Functions

- [future1](#)`< future_kind_result, T... > operator()` (T ...values)

6.121.1 Detailed Description

```
template<typename ... T>
struct upcxx::detail::make_future_< true, T... >
```

Definition at line 18 of file `make_future.hpp`.

6.121.2 Member Function Documentation

6.121.2.1 `operator()`

```
template<typename ... T>
future1<future_kind_result, T...> upcxx::detail::make_future_< true, T... >::operator() (
    T ... values ) [inline]
```

Definition at line 19 of file `make_future.hpp`.

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

- [future/make_future.hpp](#)

6.122 `upcxx::detail::make_index_sequence<n, s>` Struct Template Reference

```
#include <utility.hpp>
```

6.122.1 Detailed Description

```
template<int n, int ... s>
struct upcxx::detail::make_index_sequence<n, s >
```

Definition at line 194 of file `utility.hpp`.

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

- [utility.hpp](#)

6.123 `upcxx::detail::make_index_sequence<0, s...>` Struct Template Reference

```
#include <utility.hpp>
```

Public Types

- typedef [index_sequence](#)< s... > [type](#)

6.123.1 Detailed Description

```
template<int ... s>
struct upcxx::detail::make_index_sequence< 0, s... >
```

Definition at line 197 of file utility.hpp.

6.123.2 Member Typedef Documentation

6.123.2.1 type

```
template<int ... s>
typedef index\_sequence<s...> upcxx::detail::make\_index\_sequence< 0, s... >::type
```

Definition at line 198 of file utility.hpp.

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

- [utility.hpp](#)

6.124 mallinfo Struct Reference

```
#include <dl_malloc.h>
```

Public Attributes

- [MALLINFO_FIELD_TYPE](#) arena
- [MALLINFO_FIELD_TYPE](#) ordblks
- [MALLINFO_FIELD_TYPE](#) smblks
- [MALLINFO_FIELD_TYPE](#) hblks
- [MALLINFO_FIELD_TYPE](#) hblkhd
- [MALLINFO_FIELD_TYPE](#) usmblks
- [MALLINFO_FIELD_TYPE](#) fsmblks
- [MALLINFO_FIELD_TYPE](#) uordblks
- [MALLINFO_FIELD_TYPE](#) fordblks
- [MALLINFO_FIELD_TYPE](#) keepcost

6.124.1 Detailed Description

Definition at line 768 of file dl_malloc.c.

6.124.2 Member Data Documentation

6.124.2.1 arena

`MALLINFO_FIELD_TYPE` mallinfo::arena

Definition at line 769 of file dl_malloc.c.

6.124.2.2 fordblks

`MALLINFO_FIELD_TYPE` mallinfo::fordblks

Definition at line 777 of file dl_malloc.c.

6.124.2.3 fsmblks

`MALLINFO_FIELD_TYPE` mallinfo::fsmblks

Definition at line 775 of file dl_malloc.c.

6.124.2.4 hblkhd

`MALLINFO_FIELD_TYPE` mallinfo::hblkhd

Definition at line 773 of file dl_malloc.c.

6.124.2.5 hblks

`MALLINFO_FIELD_TYPE` mallinfo::hblks

Definition at line 772 of file dl_malloc.c.

6.124.2.6 keepcost

`MALLINFO_FIELD_TYPE` `mallinfo::keepcost`

Definition at line 778 of file `dl_malloc.c`.

6.124.2.7 ordblks

`MALLINFO_FIELD_TYPE` `mallinfo::ordblks`

Definition at line 770 of file `dl_malloc.c`.

6.124.2.8 smlbks

`MALLINFO_FIELD_TYPE` `mallinfo::smlbks`

Definition at line 771 of file `dl_malloc.c`.

6.124.2.9 uordblks

`MALLINFO_FIELD_TYPE` `mallinfo::uordblks`

Definition at line 776 of file `dl_malloc.c`.

6.124.2.10 usmlbks

`MALLINFO_FIELD_TYPE` `mallinfo::usmlbks`

Definition at line 774 of file `dl_malloc.c`.

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

- [dl_malloc.c](#)
- [dl_malloc.h](#)

6.125 malloc_chunk Struct Reference

Public Attributes

- `size_t` `prev_foot`
- `size_t` `head`
- `struct malloc_chunk *` `fd`
- `struct malloc_chunk *` `bk`

6.125.1 Detailed Description

Definition at line 2187 of file dl_malloc.c.

6.125.2 Member Data Documentation

6.125.2.1 bk

```
struct malloc_chunk* malloc_chunk::bk
```

Definition at line 2191 of file dl_malloc.c.

6.125.2.2 fd

```
struct malloc_chunk* malloc_chunk::fd
```

Definition at line 2190 of file dl_malloc.c.

6.125.2.3 head

```
size_t malloc_chunk::head
```

Definition at line 2189 of file dl_malloc.c.

6.125.2.4 prev_foot

```
size_t malloc_chunk::prev_foot
```

Definition at line 2188 of file dl_malloc.c.

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

- [dl_malloc.c](#)

6.126 malloc_params Struct Reference

Public Attributes

- `size_t` [magic](#)
- `size_t` [page_size](#)
- `size_t` [granularity](#)
- `size_t` [mmap_threshold](#)
- `size_t` [trim_threshold](#)
- `flag_t` [default_mflags](#)

6.126.1 Detailed Description

Definition at line 2618 of file dl_malloc.c.

6.126.2 Member Data Documentation

6.126.2.1 default_mflags

`flag_t` malloc_params::default_mflags

Definition at line 2624 of file dl_malloc.c.

6.126.2.2 granularity

`size_t` malloc_params::granularity

Definition at line 2621 of file dl_malloc.c.

6.126.2.3 magic

`size_t` malloc_params::magic

Definition at line 2619 of file dl_malloc.c.

6.126.2.4 mmap_threshold

```
size_t malloc_params::mmap_threshold
```

Definition at line 2622 of file dl_malloc.c.

6.126.2.5 page_size

```
size_t malloc_params::page_size
```

Definition at line 2620 of file dl_malloc.c.

6.126.2.6 trim_threshold

```
size_t malloc_params::trim_threshold
```

Definition at line 2623 of file dl_malloc.c.

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

- [dl_malloc.c](#)

6.127 malloc_segment Struct Reference

Public Attributes

- char * [base](#)
- size_t [size](#)
- struct [malloc_segment](#) * [next](#)
- flag_t [sflags](#)

6.127.1 Detailed Description

Definition at line 2472 of file dl_malloc.c.

6.127.2 Member Data Documentation

6.127.2.1 base

```
char* malloc_segment::base
```

Definition at line 2473 of file dl_malloc.c.

6.127.2.2 next

```
struct malloc_segment* malloc_segment::next
```

Definition at line 2475 of file dl_malloc.c.

6.127.2.3 sflags

```
flag_t malloc_segment::sflags
```

Definition at line 2476 of file dl_malloc.c.

6.127.2.4 size

```
size_t malloc_segment::size
```

Definition at line 2474 of file dl_malloc.c.

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

- [dl_malloc.c](#)

6.128 malloc_state Struct Reference

Public Attributes

- [binmap_t](#) smallmap
- [binmap_t](#) treemap
- [size_t](#) dsize
- [size_t](#) topsize
- [char *](#) least_addr
- [mchunkptr](#) dv
- [mchunkptr](#) top
- [size_t](#) trim_check
- [size_t](#) release_checks
- [size_t](#) magic
- [mchunkptr](#) smallbins [(NSMALLBINS+1) *2]
- [tbinptr](#) treebins [NTREEBINS]
- [size_t](#) footprint
- [size_t](#) max_footprint
- [size_t](#) footprint_limit
- [flag_t](#) mflags
- [MLOCK_T](#) mutex
- [msegment](#) seg
- [void *](#) extp
- [size_t](#) exts

6.128.1 Detailed Description

Definition at line 2582 of file dl_malloc.c.

6.128.2 Member Data Documentation

6.128.2.1 dv

`mchunkptr malloc_state::dv`

Definition at line 2588 of file dl_malloc.c.

6.128.2.2 dsize

`size_t malloc_state::dsize`

Definition at line 2585 of file dl_malloc.c.

6.128.2.3 extp

`void* malloc_state::extp`

Definition at line 2603 of file dl_malloc.c.

6.128.2.4 exts

`size_t malloc_state::exts`

Definition at line 2604 of file dl_malloc.c.

6.128.2.5 footprint

`size_t malloc_state::footprint`

Definition at line 2595 of file dl_malloc.c.

6.128.2.6 footprint_limit

```
size_t malloc_state::footprint_limit
```

Definition at line 2597 of file dl_malloc.c.

6.128.2.7 least_addr

```
char* malloc_state::least_addr
```

Definition at line 2587 of file dl_malloc.c.

6.128.2.8 magic

```
size_t malloc_state::magic
```

Definition at line 2592 of file dl_malloc.c.

6.128.2.9 max_footprint

```
size_t malloc_state::max_footprint
```

Definition at line 2596 of file dl_malloc.c.

6.128.2.10 mflags

```
flag_t malloc_state::mflags
```

Definition at line 2598 of file dl_malloc.c.

6.128.2.11 mutex

```
MLOCK_T malloc_state::mutex
```

Definition at line 2600 of file dl_malloc.c.

6.128.2.12 release_checks

size_t malloc_state::release_checks

Definition at line 2591 of file dl_malloc.c.

6.128.2.13 seg

msegment malloc_state::seg

Definition at line 2602 of file dl_malloc.c.

6.128.2.14 smallbins

mchunkptr malloc_state::smallbins[(NSMALLBINS+1) *2]

Definition at line 2593 of file dl_malloc.c.

6.128.2.15 smallmap

binmap_t malloc_state::smallmap

Definition at line 2583 of file dl_malloc.c.

6.128.2.16 top

mchunkptr malloc_state::top

Definition at line 2589 of file dl_malloc.c.

6.128.2.17 toplevel

size_t malloc_state::toplevel

Definition at line 2586 of file dl_malloc.c.

6.128.2.18 treebins

`tbinp` `malloc_state::treebins` [`N``TREEBINS`]

Definition at line 2594 of file `dl_malloc.c`.

6.128.2.19 treemap

`binmap_t` `malloc_state::treemap`

Definition at line 2584 of file `dl_malloc.c`.

6.128.2.20 trim_check

`size_t` `malloc_state::trim_check`

Definition at line 2590 of file `dl_malloc.c`.

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

- [dl_malloc.c](#)

6.129 malloc_tree_chunk Struct Reference

Public Attributes

- `size_t` `prev_foot`
- `size_t` `head`
- `struct malloc_tree_chunk *` `fd`
- `struct malloc_tree_chunk *` `bk`
- `struct malloc_tree_chunk *` `child` [2]
- `struct malloc_tree_chunk *` `parent`
- `bindex_t` `index`

6.129.1 Detailed Description

Definition at line 2396 of file `dl_malloc.c`.

6.129.2 Member Data Documentation

6.129.2.1 bk

```
struct malloc_tree_chunk* malloc_tree_chunk::bk
```

Definition at line 2401 of file dl_malloc.c.

6.129.2.2 child

```
struct malloc_tree_chunk* malloc_tree_chunk::child[2]
```

Definition at line 2403 of file dl_malloc.c.

6.129.2.3 fd

```
struct malloc_tree_chunk* malloc_tree_chunk::fd
```

Definition at line 2400 of file dl_malloc.c.

6.129.2.4 head

```
size_t malloc_tree_chunk::head
```

Definition at line 2399 of file dl_malloc.c.

6.129.2.5 index

```
bindex_t malloc_tree_chunk::index
```

Definition at line 2405 of file dl_malloc.c.

6.129.2.6 parent

```
struct malloc_tree_chunk* malloc_tree_chunk::parent
```

Definition at line 2404 of file dl_malloc.c.

6.129.2.7 prev_foot

```
size_t malloc_tree_chunk::prev_foot
```

Definition at line 2398 of file dl_malloc.c.

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

- [dl_malloc.c](#)

6.130 upcxx::mod2n_hashing< T > Struct Template Reference

```
#include <reflection.hpp>
```

Public Member Functions

- `std::size_t operator() (const T &x, int bit_n) const`

6.130.1 Detailed Description

```
template<typename T>  
struct upcxx::mod2n_hashing< T >
```

Definition at line 157 of file reflection.hpp.

6.130.2 Member Function Documentation

6.130.2.1 operator()

```
template<typename T >  
std::size_t upcxx::mod2n_hashing< T >::operator() (  
    const T & x,  
    int bit_n ) const [inline]
```

Definition at line 158 of file reflection.hpp.

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

- [reflection.hpp](#)

6.131 upcxx::nil_cx Struct Reference

```
#include <completion.hpp>
```

6.131.1 Detailed Description

Definition at line 8 of file `completion.hpp`.

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

- [completion.hpp](#)

6.132 `upcxx::nop_function< Sig >` Struct Template Reference

```
#include <utility.hpp>
```

6.132.1 Detailed Description

```
template<typename Sig>  
struct upcxx::nop_function< Sig >
```

Definition at line 19 of file `utility.hpp`.

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

- [utility.hpp](#)

6.133 `upcxx::nop_function< Ret(Arg...)>` Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- `template<typename ... Arg1>`
`Ret operator() (Arg1 &&...a) const`

6.133.1 Detailed Description

```
template<typename Ret, typename ... Arg>  
struct upcxx::nop_function< Ret(Arg...)>
```

Definition at line 22 of file `utility.hpp`.

6.133.2 Member Function Documentation

6.133.2.1 operator()

```
template<typename Ret , typename ... Arg>
template<typename ... Arg1>
Ret upcxx::nop_function< Ret (Arg...)>::operator() (
    Arg1 &&... a ) const [inline]
```

Definition at line 24 of file utility.hpp.

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

- [utility.hpp](#)

6.134 upcxx::nop_function< void(Arg...)> Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- `template<typename ... Arg1>`
void `operator()` (Arg1 &&...a) const

6.134.1 Detailed Description

```
template<typename ... Arg>
struct upcxx::nop_function< void(Arg...)>
```

Definition at line 30 of file utility.hpp.

6.134.2 Member Function Documentation

6.134.2.1 operator()

```
template<typename ... Arg>
template<typename ... Arg1>
void upcxx::nop_function< void(Arg...)>::operator() (
    Arg1 &&... a ) const [inline]
```

Definition at line 32 of file utility.hpp.

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

- [utility.hpp](#)

6.135 upcxx::detail::os_env_parse< T > Struct Template Reference

```
#include <os_env.hpp>
```

Public Member Functions

- [T operator\(\)](#) (std::string x)

6.135.1 Detailed Description

```
template<class T>  
struct upcxx::detail::os_env_parse< T >
```

Definition at line 19 of file os_env.hpp.

6.135.2 Member Function Documentation

6.135.2.1 operator()

```
template<class T >  
T upcxx::detail::os_env_parse< T >::operator() (  
    std::string x ) [inline]
```

Definition at line 30 of file os_env.hpp.

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

- [os_env.hpp](#)

6.136 upcxx::detail::os_env_parse< std::string > Struct Template Reference

```
#include <os_env.hpp>
```

Public Member Functions

- [std::string operator\(\)](#) (std::string x)

6.136.1 Detailed Description

```
template<>  
struct upcxx::detail::os_env_parse< std::string >
```

Definition at line 22 of file os_env.hpp.

6.136.2 Member Function Documentation

6.136.2.1 operator()

```
std::string upcxx::detail::os_env_parse< std::string >::operator() (
    std::string x ) [inline]
```

Definition at line 23 of file os_env.hpp.

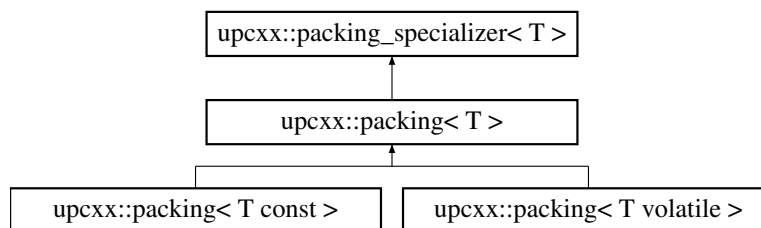
The documentation for this struct was generated from the following file:

- [os_env.hpp](#)

6.137 upcxx::packing< T > Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for upcxx::packing< T >:



6.137.1 Detailed Description

```
template<typename T>
struct upcxx::packing< T >
```

Definition at line 577 of file packing.hpp.

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

- [packing.hpp](#)

6.138 upcxx::packing< bound_function< Fn, B... > > Struct Template Reference

```
#include <bind.hpp>
```


Static Public Member Functions

- static void `size_ubound` (`parcel_layout` &ub, const `bound_function`< Fn, B... > &fn)
- static void `pack` (`parcel_writer` &w, const `bound_function`< Fn, B... > &fn)
- static `bound_function`< Fn, B... > `unpack` (`parcel_reader` &r)

6.138.1 Detailed Description

```
template<typename Fn, typename ... B>
struct upcxx::packing< bound_function< Fn, B... > >
```

Definition at line 222 of file `bind.hpp`.

6.138.2 Member Function Documentation

6.138.2.1 `pack()`

```
template<typename Fn , typename ... B>
static void upcxx::packing< bound_function< Fn, B... > >::pack (
    parcel_writer & w,
    const bound_function< Fn, B... > & fn ) [inline], [static]
```

Definition at line 228 of file `bind.hpp`.

6.138.2.2 `size_ubound()`

```
template<typename Fn , typename ... B>
static void upcxx::packing< bound_function< Fn, B... > >::size_ubound (
    parcel_layout & ub,
    const bound_function< Fn, B... > & fn ) [inline], [static]
```

Definition at line 223 of file `bind.hpp`.

6.138.2.3 `unpack()`

```
template<typename Fn , typename ... B>
static bound_function<Fn,B...> upcxx::packing< bound_function< Fn, B... > >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 233 of file `bind.hpp`.

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

- [bind.hpp](#)

6.139 upcxx::packing< std::array< T, n > > Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void [size_ubound](#) ([parcel_layout](#) &ub, const std::array< T, n > &x)
- static void [pack](#) ([parcel_writer](#) &w, const std::array< T, n > &x)
- static std::array< T, n > [unpack](#) ([parcel_reader](#) &r)

6.139.1 Detailed Description

```
template<typename T, std::size_t n>  
struct upcxx::packing< std::array< T, n > >
```

Definition at line 695 of file packing.hpp.

6.139.2 Member Function Documentation

6.139.2.1 pack()

```
template<typename T , std::size_t n>  
static void upcxx::packing< std::array< T, n > >::pack (  
    parcel\_writer & w,  
    const std::array< T, n > & x ) [inline], [static]
```

Definition at line 701 of file packing.hpp.

6.139.2.2 size_ubound()

```
template<typename T , std::size_t n>  
static void upcxx::packing< std::array< T, n > >::size_ubound (  
    parcel\_layout & ub,  
    const std::array< T, n > & x ) [inline], [static]
```

Definition at line 696 of file packing.hpp.

6.139.2.3 `unpack()`

```
template<typename T , std::size_t n>
static std::array<T,n> upcxx::packing< std::array< T, n > >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 706 of file `packing.hpp`.

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

- [packing.hpp](#)

6.140 `upcxx::packing< std::pair< A, B > >` Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void `size_ubound` (`parcel_layout` &ub, const `std::pair< A, B > &x`)
- static void `pack` (`parcel_writer` &w, const `std::pair< A, B > &x`)
- static `std::pair< A, B >` `unpack` (`parcel_reader` &r)

6.140.1 Detailed Description

```
template<typename A, typename B>
struct upcxx::packing< std::pair< A, B > >
```

Definition at line 672 of file `packing.hpp`.

6.140.2 Member Function Documentation

6.140.2.1 `pack()`

```
template<typename A , typename B >
static void upcxx::packing< std::pair< A, B > >::pack (
    parcel_writer & w,
    const std::pair< A, B > &x ) [inline], [static]
```

Definition at line 678 of file `packing.hpp`.

6.140.2.2 size_ubound()

```
template<typename A , typename B >
static void upcxx::packing< std::pair< A, B > >::size_ubound (
    parcel_layout & ub,
    const std::pair< A, B > & x ) [inline], [static]
```

Definition at line 673 of file packing.hpp.

6.140.2.3 unpack()

```
template<typename A , typename B >
static std::pair<A,B> upcxx::packing< std::pair< A, B > >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 683 of file packing.hpp.

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

- [packing.hpp](#)

6.141 upcxx::packing< std::string > Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void [size_ubound](#) ([parcel_layout](#) &ub, const std::string &x)
- static void [pack](#) ([parcel_writer](#) &w, const std::string &x)
- static std::string [unpack](#) ([parcel_reader](#) &r)

6.141.1 Detailed Description

```
template<>
struct upcxx::packing< std::string >
```

Definition at line 724 of file packing.hpp.

6.141.2 Member Function Documentation

6.141.2.1 `pack()`

```
static void upcxx::packing< std::string >::pack (
    parcel_writer & w,
    const std::string & x ) [inline], [static]
```

Definition at line 731 of file `packing.hpp`.

6.141.2.2 `size_ubound()`

```
static void upcxx::packing< std::string >::size_ubound (
    parcel_layout & ub,
    const std::string & x ) [inline], [static]
```

Definition at line 725 of file `packing.hpp`.

6.141.2.3 `unpack()`

```
static std::string upcxx::packing< std::string >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 737 of file `packing.hpp`.

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

- [packing.hpp](#)

6.142 `upcxx::packing< std::tuple< T... > >` Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void `size_ubound` (`parcel_layout` &ub, const `std::tuple< T... >` &x)
- static void `pack` (`parcel_writer` &w, const `std::tuple< T... >` &x)
- template<typename `Storage`, int ... `i`>
static `std::tuple< T... >` `move_from_storage` (`Storage` &storage, `upcxx::index_sequence< i... >`)
- static `std::tuple< T... >` `unpack` (`parcel_reader` &r)

6.142.1 Detailed Description

```
template<typename ... T>
struct upcxx::packing< std::tuple< T... > >
```

Definition at line 632 of file `packing.hpp`.

6.142.2 Member Function Documentation

6.142.2.1 move_from_storage()

```
template<typename ... T>
template<typename Storage , int ... i>
static std::tuple<T...> upcxx::packing< std::tuple< T... > >::move_from_storage (
    Storage & storage,
    upcxx::index_sequence< i... > ) [inline], [static]
```

Definition at line 642 of file packing.hpp.

6.142.2.2 pack()

```
template<typename ... T>
static void upcxx::packing< std::tuple< T... > >::pack (
    parcel_writer & w,
    const std::tuple< T... > & x ) [inline], [static]
```

Definition at line 637 of file packing.hpp.

6.142.2.3 size_ubound()

```
template<typename ... T>
static void upcxx::packing< std::tuple< T... > >::size_ubound (
    parcel_layout & ub,
    const std::tuple< T... > & x ) [inline], [static]
```

Definition at line 633 of file packing.hpp.

6.142.2.4 unpack()

```
template<typename ... T>
static std::tuple<T...> upcxx::packing< std::tuple< T... > >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 651 of file packing.hpp.

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

- [packing.hpp](#)

6.143 `upcxx::packing< std::unordered_map< K, V > >` Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void `size_ubound` (`parcel_layout` &ub, const `std::unordered_map< K, V >` &xs)
- static void `pack` (`parcel_writer` &w, const `std::unordered_map< K, V >` &xs)
- static `std::unordered_map< K, V >` `unpack` (`parcel_reader` &r)

6.143.1 Detailed Description

```
template<typename K, typename V>  
struct upcxx::packing< std::unordered_map< K, V > >
```

Definition at line 809 of file `packing.hpp`.

6.143.2 Member Function Documentation

6.143.2.1 `pack()`

```
template<typename K , typename V >  
static void upcxx::packing< std::unordered_map< K, V > >::pack (  
    parcel_writer & w,  
    const std::unordered_map< K, V > & xs ) [inline], [static]
```

Definition at line 817 of file `packing.hpp`.

6.143.2.2 `size_ubound()`

```
template<typename K , typename V >  
static void upcxx::packing< std::unordered_map< K, V > >::size_ubound (  
    parcel_layout & ub,  
    const std::unordered_map< K, V > & xs ) [inline], [static]
```

Definition at line 810 of file `packing.hpp`.

6.143.2.3 `unpack()`

```
template<typename K , typename V >
static std::unordered_map<K,V> upcxx::packing< std::unordered_map< K, V > >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 824 of file `packing.hpp`.

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

- [packing.hpp](#)

6.144 `upcxx::packing< std::unordered_set< T > >` Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void `size_ubound` (`parcel_layout` &ub, const `std::unordered_set< T >` &xs)
- static void `pack` (`parcel_writer` &w, const `std::unordered_set< T >` &xs)
- static `std::unordered_set< T >` `unpack` (`parcel_reader` &r)

6.144.1 Detailed Description

```
template<typename T>
struct upcxx::packing< std::unordered_set< T > >
```

Definition at line 780 of file `packing.hpp`.

6.144.2 Member Function Documentation

6.144.2.1 `pack()`

```
template<typename T >
static void upcxx::packing< std::unordered_set< T > >::pack (
    parcel_writer & w,
    const std::unordered_set< T > & xs ) [inline], [static]
```

Definition at line 788 of file `packing.hpp`.

6.144.2.2 `size_ubound()`

```
template<typename T >
static void upcxx::packing< std::unordered_set< T > >::size_ubound (
    parcel_layout & ub,
    const std::unordered_set< T > & xs ) [inline], [static]
```

Definition at line 781 of file `packing.hpp`.

6.144.2.3 `unpack()`

```
template<typename T >
static std::unordered_set<T> upcxx::packing< std::unordered_set< T > >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 795 of file `packing.hpp`.

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

- [packing.hpp](#)

6.145 `upcxx::packing< std::vector< T > >` Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void `size_ubound` (`parcel_layout` &`ub`, const std::vector< T > &`x`)
- static void `pack` (`parcel_writer` &`w`, const std::vector< T > &`x`)
- static std::vector< T > `unpack` (`parcel_reader` &`r`)

6.145.1 Detailed Description

```
template<typename T>
struct upcxx::packing< std::vector< T > >
```

Definition at line 751 of file `packing.hpp`.

6.145.2 Member Function Documentation

6.145.2.1 pack()

```
template<typename T >
static void upcxx::packing< std::vector< T > >::pack (
    parcel_writer & w,
    const std::vector< T > & x ) [inline], [static]
```

Definition at line 759 of file packing.hpp.

6.145.2.2 size_ubound()

```
template<typename T >
static void upcxx::packing< std::vector< T > >::size_ubound (
    parcel_layout & ub,
    const std::vector< T > & x ) [inline], [static]
```

Definition at line 752 of file packing.hpp.

6.145.2.3 unpack()

```
template<typename T >
static std::vector<T> upcxx::packing< std::vector< T > >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 766 of file packing.hpp.

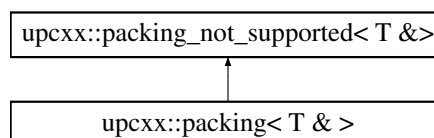
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.146 upcxx::packing< T & > Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for upcxx::packing< T & >:



Additional Inherited Members

6.146.1 Detailed Description

```
template<typename T>
struct upcxx::packing< T & >
```

Definition at line 585 of file `packing.hpp`.

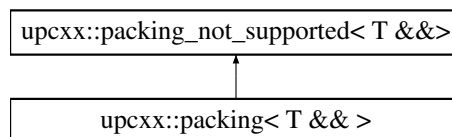
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.147 `upcxx::packing< T && >` Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for `upcxx::packing< T && >`:



Additional Inherited Members

6.147.1 Detailed Description

```
template<typename T>
struct upcxx::packing< T && >
```

Definition at line 587 of file `packing.hpp`.

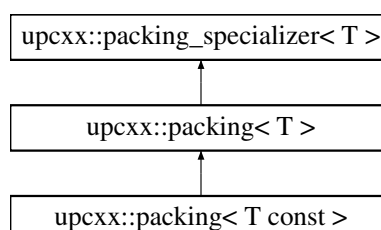
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.148 `upcxx::packing< T const >` Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for `upcxx::packing< T const >`:



6.148.1 Detailed Description

```
template<typename T>
struct upcxx::packing< T const >
```

Definition at line 580 of file packing.hpp.

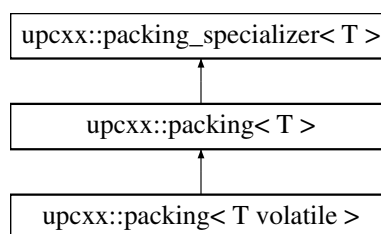
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.149 upcxx::packing< T volatile > Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for upcxx::packing< T volatile >:



6.149.1 Detailed Description

```
template<typename T>
struct upcxx::packing< T volatile >
```

Definition at line 582 of file packing.hpp.

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

- [packing.hpp](#)

6.150 upcxx::packing_empty< T, is_default_constructible > Struct Template Reference

```
#include <packing.hpp>
```

6.150.1 Detailed Description

```
template<typename T, bool is_default_constructible = std::is_default_constructible<T>::value>
struct upcxx::packing_empty< T, is_default_constructible >
```

Definition at line 306 of file packing.hpp.

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

- [packing.hpp](#)

6.151 upcxx::packing_empty< T, false > Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void [size_ubound](#) ([parcel_layout](#) &ub, const T &x)
- static void [pack](#) ([parcel_writer](#) &w, const T &x)
- static T [unpack](#) ([parcel_reader](#) &r)

Static Public Attributes

- static constexpr bool [is_trivial](#) = true

6.151.1 Detailed Description

```
template<typename T>  
struct upcxx::packing_empty< T, false >
```

Definition at line 318 of file packing.hpp.

6.151.2 Member Function Documentation

6.151.2.1 pack()

```
template<typename T >  
static void upcxx::packing_empty< T, false >::pack (  
    parcel\_writer & w,  
    const T & x ) [inline], [static]
```

Definition at line 322 of file packing.hpp.

6.151.2.2 size_ubound()

```
template<typename T >  
static void upcxx::packing_empty< T, false >::size_ubound (  
    parcel\_layout & ub,  
    const T & x ) [inline], [static]
```

Definition at line 321 of file packing.hpp.

6.151.2.3 unpack()

```
template<typename T >
static T upcxx::packing_empty< T, false >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 323 of file packing.hpp.

6.151.3 Member Data Documentation

6.151.3.1 is_trivial

```
template<typename T >
constexpr bool upcxx::packing_empty< T, false >::is_trivial = true [static]
```

Definition at line 319 of file packing.hpp.

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

- [packing.hpp](#)

6.152 upcxx::packing_empty< T, true > Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void [size_ubound](#) ([parcel_layout](#) &ub, const T &x)
- static void [pack](#) ([parcel_writer](#) &w, const T &x)
- static T [unpack](#) ([parcel_reader](#) &r)

Static Public Attributes

- static constexpr bool [is_trivial](#) = true

6.152.1 Detailed Description

```
template<typename T>
struct upcxx::packing_empty< T, true >
```

Definition at line 309 of file packing.hpp.

6.152.2 Member Function Documentation

6.152.2.1 pack()

```
template<typename T >
static void upcxx::packing_empty< T, true >::pack (
    parcel_writer & w,
    const T & x ) [inline], [static]
```

Definition at line 313 of file packing.hpp.

6.152.2.2 size_ubound()

```
template<typename T >
static void upcxx::packing_empty< T, true >::size_ubound (
    parcel_layout & ub,
    const T & x ) [inline], [static]
```

Definition at line 312 of file packing.hpp.

6.152.2.3 unpack()

```
template<typename T >
static T upcxx::packing_empty< T, true >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 314 of file packing.hpp.

6.152.3 Member Data Documentation

6.152.3.1 is_trivial

```
template<typename T >
constexpr bool upcxx::packing_empty< T, true >::is_trivial = true [static]
```

Definition at line 310 of file packing.hpp.

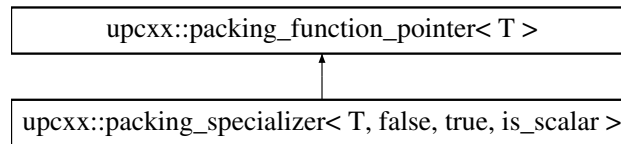
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.153 upcxx::packing_function_pointer< T > Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for upcxx::packing_function_pointer< T >:



Static Public Member Functions

- static void [size_ubound](#) ([parcel_layout](#) &ub, T fp)
- static void [pack](#) ([parcel_writer](#) &w, T fp)
- static T [unpack](#) ([parcel_reader](#) &r)

Static Public Attributes

- static constexpr bool [is_trivial](#) = false

6.153.1 Detailed Description

```
template<typename T>
struct upcxx::packing_function_pointer< T >
```

Definition at line 521 of file packing.hpp.

6.153.2 Member Function Documentation

6.153.2.1 pack()

```
template<typename T >
static void upcxx::packing_function_pointer< T >::pack (
    parcel_writer & w,
    T fp ) [inline], [static]
```

Definition at line 533 of file packing.hpp.

6.153.2.2 `size_ubound()`

```
template<typename T >
static void upcxx::packing_function_pointer< T >::size_ubound (
    parcel_layout & ub,
    T fp ) [inline], [static]
```

Definition at line 529 of file `packing.hpp`.

6.153.2.3 `unpack()`

```
template<typename T >
static T upcxx::packing_function_pointer< T >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 541 of file `packing.hpp`.

6.153.3 Member Data Documentation

6.153.3.1 `is_trivial`

```
template<typename T >
constexpr bool upcxx::packing_function_pointer< T >::is_trivial = false [static]
```

Definition at line 527 of file `packing.hpp`.

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

- [packing.hpp](#)

6.154 `upcxx::packing_is_trivial< T, false_ >` Struct Template Reference

```
#include <packing.hpp>
```

Static Public Attributes

- static constexpr bool `value` = false

6.154.1 Detailed Description

```
template<typename T, typename false_ = std::false_type >
struct upcxx::packing_is_trivial< T, false_ >
```

Definition at line 293 of file `packing.hpp`.

6.154.2 Member Data Documentation

6.154.2.1 value

```
template<typename T , typename false_ = std::false_type>
constexpr bool upcxx::packing_is_trivial< T, false_ >::value = false [static]
```

Definition at line 294 of file packing.hpp.

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

- [packing.hpp](#)

6.155 upcxx::packing_is_trivial< T, std::integral_constant< bool, false &packing< T >::is_trivial > > Struct Template Reference

```
#include <packing.hpp>
```

Static Public Attributes

- static constexpr bool [value](#) = [packing](#)<T>::is_trivial

6.155.1 Detailed Description

```
template<typename T>
struct upcxx::packing_is_trivial< T, std::integral_constant< bool, false &packing< T >::is_trivial > >
```

Definition at line 297 of file packing.hpp.

6.155.2 Member Data Documentation

6.155.2.1 value

```
template<typename T >
constexpr bool upcxx::packing_is_trivial< T, std::integral_constant< bool, false &packing< T >::is_trivial > >::value = packing<T>::is_trivial [static]
```

Definition at line 298 of file packing.hpp.

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

- [packing.hpp](#)

6.156 `upcxx::packing_not_supported< T >` Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void `size_ubound` (`parcel_layout` &`ub`, const T &`x`)

6.156.1 Detailed Description

```
template<typename T>  
struct upcxx::packing_not_supported< T >
```

Definition at line 276 of file `packing.hpp`.

6.156.2 Member Function Documentation

6.156.2.1 `size_ubound()`

```
template<typename T>  
static void upcxx::packing_not_supported< T >::size_ubound (  
    parcel_layout & ub,  
    const T & x ) [inline], [static]
```

Definition at line 277 of file `packing.hpp`.

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

- [packing.hpp](#)

6.157 `upcxx::packing_opaque< T, is_empty, is_trivially_copyable >` Struct Template Reference

```
#include <packing.hpp>
```

6.157.1 Detailed Description

```
template<typename T, bool is_empty = std::is_empty<T>::value, bool is_trivially_copyable = std::is_trivially_copyable<T>::value>  
struct upcxx::packing_opaque< T, is_empty, is_trivially_copyable >
```

Definition at line 353 of file `packing.hpp`.

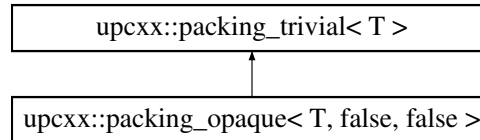
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.158 `upcxx::packing_opaque< T, false, false >` Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for `upcxx::packing_opaque< T, false, false >`:



Additional Inherited Members

6.158.1 Detailed Description

```
template<typename T>
struct upcxx::packing_opaque< T, false, false >
```

Definition at line 365 of file `packing.hpp`.

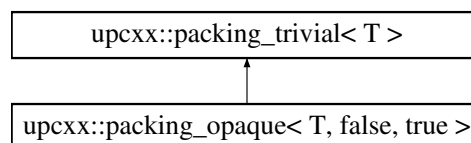
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.159 `upcxx::packing_opaque< T, false, true >` Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for `upcxx::packing_opaque< T, false, true >`:



Additional Inherited Members

6.159.1 Detailed Description

```
template<typename T>
struct upcxx::packing_opaque< T, false, true >
```

Definition at line 360 of file `packing.hpp`.

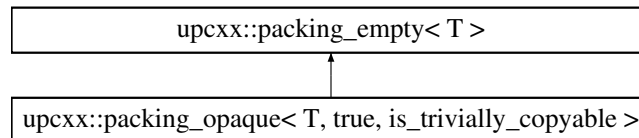
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.160 upcxx::packing_opaque< T, true, is_trivially_copyable > Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for upcxx::packing_opaque< T, true, is_trivially_copyable >:



6.160.1 Detailed Description

```
template<typename T, bool is_trivially_copyable>
struct upcxx::packing_opaque< T, true, is_trivially_copyable >
```

Definition at line 356 of file packing.hpp.

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

- [packing.hpp](#)

6.161 upcxx::packing_pack_reflector Struct Reference

```
#include <packing.hpp>
```

Public Member Functions

- template<typename T >
void [operator\(\)](#) (const T &mbr)
- template<typename T >
void [opaque](#) (const T &mbr)

Public Attributes

- [parcel_writer](#) & w

6.161.1 Detailed Description

Definition at line 394 of file packing.hpp.

6.161.2 Member Function Documentation

6.161.2.1 opaque()

```
template<typename T >
void upcxx::packing_pack_reflector::opaque (
    const T & mbr ) [inline]
```

Definition at line 402 of file packing.hpp.

6.161.2.2 operator()

```
template<typename T >
void upcxx::packing_pack_reflector::operator() (
    const T & mbr ) [inline]
```

Definition at line 398 of file packing.hpp.

6.161.3 Member Data Documentation

6.161.3.1 w

```
parcel_writer& upcxx::packing_pack_reflector::w
```

Definition at line 395 of file packing.hpp.

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

- [packing.hpp](#)

6.162 upcxx::packing_reflected< T, is_default_constructible > Struct Template Reference

```
#include <packing.hpp>
```

6.162.1 Detailed Description

```
template<typename T, bool is_default_constructible = std::is_default_constructible<T>::value>
struct upcxx::packing_reflected< T, is_default_constructible >
```

Definition at line 440 of file `packing.hpp`.

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

- [packing.hpp](#)

6.163 `upcxx::packing_reflected< T, false >` Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void `size_ubound` (`parcel_layout` &ub, const T &x)
- static void `pack` (`parcel_writer` &w, const T &x)
- static T `unpack` (`parcel_reader` &r)

6.163.1 Detailed Description

```
template<typename T>
struct upcxx::packing_reflected< T, false >
```

Definition at line 463 of file `packing.hpp`.

6.163.2 Member Function Documentation

6.163.2.1 `pack()`

```
template<typename T >
static void upcxx::packing_reflected< T, false >::pack (
    parcel_writer & w,
    const T & x ) [inline], [static]
```

Definition at line 469 of file `packing.hpp`.

6.163.2.2 size_ubound()

```
template<typename T >
static void upcxx::packing_reflected< T, false >::size_ubound (
    parcel_layout & ub,
    const T & x ) [inline], [static]
```

Definition at line 464 of file packing.hpp.

6.163.2.3 unpack()

```
template<typename T >
static T upcxx::packing_reflected< T, false >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 474 of file packing.hpp.

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

- [packing.hpp](#)

6.164 upcxx::packing_reflected< T, true > Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void [size_ubound](#) ([parcel_layout](#) &ub, const T &x)
- static void [pack](#) ([parcel_writer](#) &w, const T &x)
- static T [unpack](#) ([parcel_reader](#) &r)

6.164.1 Detailed Description

```
template<typename T>
struct upcxx::packing_reflected< T, true >
```

Definition at line 443 of file packing.hpp.

6.164.2 Member Function Documentation

6.164.2.1 pack()

```
template<typename T >
static void upcxx::packing_reflected< T, true >::pack (
    parcel_writer & w,
    const T & x ) [inline], [static]
```

Definition at line 449 of file packing.hpp.

6.164.2.2 size_ubound()

```
template<typename T >
static void upcxx::packing_reflected< T, true >::size_ubound (
    parcel_layout & ub,
    const T & x ) [inline], [static]
```

Definition at line 444 of file packing.hpp.

6.164.2.3 unpack()

```
template<typename T >
static T upcxx::packing_reflected< T, true >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 454 of file packing.hpp.

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

- [packing.hpp](#)

6.165 upcxx::packing_specializer< T, is_empty, is_funptr, is_scalar > Struct Template Reference

```
#include <packing.hpp>
```

6.165.1 Detailed Description

```
template<typename T, bool is_empty = std::is_empty<T>::value, bool is_funptr = std::is_pointer<T>::value && std::is_↵
function<typename std::remove_pointer<T>::type>::value, bool is_scalar = std::is_scalar<T>::value>
struct upcxx::packing_specializer< T, is_empty, is_funptr, is_scalar >
```

Definition at line 556 of file packing.hpp.

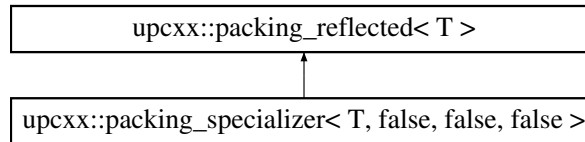
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.166 `upcxx::packing_specializer< T, false, false, false >` Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for `upcxx::packing_specializer< T, false, false, false >`:



6.166.1 Detailed Description

```
template<typename T>
struct upcxx::packing_specializer< T, false, false, false >
```

Definition at line 571 of file `packing.hpp`.

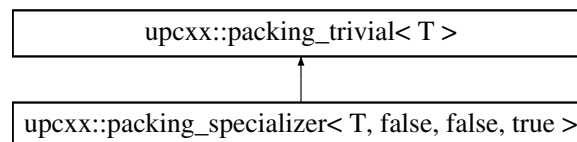
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.167 `upcxx::packing_specializer< T, false, false, true >` Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for `upcxx::packing_specializer< T, false, false, true >`:



Additional Inherited Members

6.167.1 Detailed Description

```
template<typename T>
struct upcxx::packing_specializer< T, false, false, true >
```

Definition at line 567 of file `packing.hpp`.

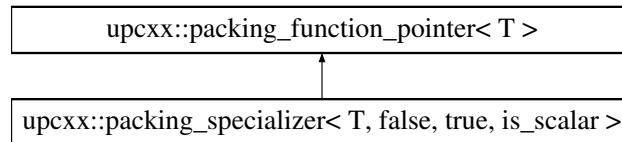
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.168 upcxx::packing_specializer< T, false, true, is_scalar > Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for upcxx::packing_specializer< T, false, true, is_scalar >:



Additional Inherited Members

6.168.1 Detailed Description

```
template<typename T, bool is_scalar>  
struct upcxx::packing_specializer< T, false, true, is_scalar >
```

Definition at line 563 of file packing.hpp.

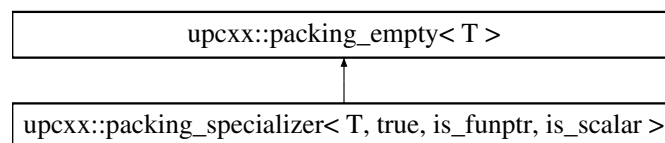
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.169 upcxx::packing_specializer< T, true, is_funptr, is_scalar > Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for upcxx::packing_specializer< T, true, is_funptr, is_scalar >:



6.169.1 Detailed Description

```
template<typename T, bool is_funptr, bool is_scalar>  
struct upcxx::packing_specializer< T, true, is_funptr, is_scalar >
```

Definition at line 559 of file packing.hpp.

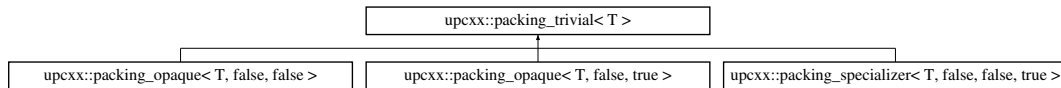
The documentation for this struct was generated from the following file:

- [packing.hpp](#)

6.170 upcxx::packing_trivial< T > Struct Template Reference

```
#include <packing.hpp>
```

Inheritance diagram for upcxx::packing_trivial< T >:



Static Public Member Functions

- static void [size_ubound](#) ([parcel_layout](#) &ub, const T &x)
- static void [pack](#) ([parcel_writer](#) &w, const T &x)
- static T [unpack](#) ([parcel_reader](#) &r)

Static Public Attributes

- static constexpr bool [is_trivial](#) = true

6.170.1 Detailed Description

```
template<typename T>
struct upcxx::packing_trivial< T >
```

Definition at line 333 of file packing.hpp.

6.170.2 Member Function Documentation

6.170.2.1 pack()

```
template<typename T >
static void upcxx::packing_trivial< T >::pack (
    parcel_writer & w,
    const T & x ) [inline], [static]
```

Definition at line 339 of file packing.hpp.

6.170.2.2 `size_ubound()`

```
template<typename T >
static void upcxx::packing_trivial< T >::size_ubound (
    parcel_layout & ub,
    const T & x ) [inline], [static]
```

Definition at line 336 of file `packing.hpp`.

6.170.2.3 `unpack()`

```
template<typename T >
static T upcxx::packing_trivial< T >::unpack (
    parcel_reader & r ) [inline], [static]
```

Definition at line 342 of file `packing.hpp`.

6.170.3 Member Data Documentation

6.170.3.1 `is_trivial`

```
template<typename T >
constexpr bool upcxx::packing_trivial< T >::is_trivial = true [static]
```

Definition at line 334 of file `packing.hpp`.

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

- [packing.hpp](#)

6.171 `upcxx::detail::packing_tuple_each< n, i, T >` Struct Template Reference

```
#include <packing.hpp>
```

Public Types

- typedef `std::tuple_element< i, std::tuple< T... > >::type` `Ti`

Static Public Member Functions

- static void [size_ubound](#) ([parcel_layout](#) &ub, const std::tuple< T... > &x)
- static void [pack](#) ([parcel_writer](#) &w, const std::tuple< T... > &x)
- template<typename Storage >
static void [unpack_into](#) ([parcel_reader](#) &r, Storage &storage)
- template<typename Storage >
static void [destruct](#) (Storage &storage)

6.171.1 Detailed Description

```
template<int n, int i, typename ... T>
struct upcxx::detail::packing_tuple_each< n, i, T >
```

Definition at line 594 of file packing.hpp.

6.171.2 Member Typedef Documentation

6.171.2.1 Ti

```
template<int n, int i, typename ... T>
typedef std::tuple_element<i, std::tuple<T...> >::type upcxx::detail::packing\_tuple\_each< n,
i, T >::Ti
```

Definition at line 595 of file packing.hpp.

6.171.3 Member Function Documentation

6.171.3.1 destruct()

```
template<int n, int i, typename ... T>
template<typename Storage >
static void upcxx::detail::packing\_tuple\_each< n, i, T >::destruct (
    Storage & storage ) [inline], [static]
```

Definition at line 614 of file packing.hpp.

6.171.3.2 pack()

```
template<int n, int i, typename ... T>
static void upcxx::detail::packing_tuple_each< n, i, T >::pack (
    parcel_writer & w,
    const std::tuple< T... > & x ) [inline], [static]
```

Definition at line 602 of file packing.hpp.

6.171.3.3 size_ubound()

```
template<int n, int i, typename ... T>
static void upcxx::detail::packing_tuple_each< n, i, T >::size_ubound (
    parcel_layout & ub,
    const std::tuple< T... > & x ) [inline], [static]
```

Definition at line 597 of file packing.hpp.

6.171.3.4 unpack_into()

```
template<int n, int i, typename ... T>
template<typename Storage >
static void upcxx::detail::packing_tuple_each< n, i, T >::unpack_into (
    parcel_reader & r,
    Storage & storage ) [inline], [static]
```

Definition at line 608 of file packing.hpp.

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

- [packing.hpp](#)

6.172 upcxx::detail::packing_tuple_each< n, n, T... > Struct Template Reference

```
#include <packing.hpp>
```

Static Public Member Functions

- static void [size_ubound](#) ([parcel_layout](#) &ub, const std::tuple< T... > &x)
- static void [pack](#) ([parcel_writer](#) &w, const std::tuple< T... > &x)
- template<typename Storage >
static void [unpack_into](#) ([parcel_reader](#) &r, Storage &)
- template<typename Storage >
static void [destruct](#) (Storage &)

6.172.1 Detailed Description

```
template<int n, typename ... T>
struct upcxx::detail::packing_tuple_each< n, n, T... >
```

Definition at line 621 of file packing.hpp.

6.172.2 Member Function Documentation

6.172.2.1 destruct()

```
template<int n, typename ... T>
template<typename Storage >
static void upcxx::detail::packing_tuple_each< n, n, T... >::destruct (
    Storage & ) [inline], [static]
```

Definition at line 627 of file packing.hpp.

6.172.2.2 pack()

```
template<int n, typename ... T>
static void upcxx::detail::packing_tuple_each< n, n, T... >::pack (
    parcel_writer & w,
    const std::tuple< T... > & x ) [inline], [static]
```

Definition at line 623 of file packing.hpp.

6.172.2.3 size_ubound()

```
template<int n, typename ... T>
static void upcxx::detail::packing_tuple_each< n, n, T... >::size_ubound (
    parcel_layout & ub,
    const std::tuple< T... > & x ) [inline], [static]
```

Definition at line 622 of file packing.hpp.

6.172.2.4 unpack_into()

```
template<int n, typename ... T>
template<typename Storage >
static void upcxx::detail::packing_tuple_each< n, n, T... >::unpack_into (
    parcel_reader & r,
    Storage & ) [inline], [static]
```

Definition at line 625 of file packing.hpp.

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

- [packing.hpp](#)

6.173 upcxx::packing_ubound_reflector Struct Reference

```
#include <packing.hpp>
```

Public Member Functions

- template<typename T >
void [operator\(\)](#) (const T &mbr)
- template<typename T >
void [opaque](#) (const T &mbr)

Public Attributes

- [parcel_layout](#) & [ub](#)

6.173.1 Detailed Description

Definition at line 380 of file packing.hpp.

6.173.2 Member Function Documentation

6.173.2.1 opaque()

```
template<typename T >
void upcxx::packing_ubound_reflector::opaque (
    const T & mbr ) [inline]
```

Definition at line 388 of file packing.hpp.

6.173.2.2 operator()

```
template<typename T >
void upcxx::packing_ubound_reflector::operator() (
    const T & mbr ) [inline]
```

Definition at line 384 of file packing.hpp.

6.173.3 Member Data Documentation

6.173.3.1 ub

```
parcel_layout& upcxx::packing_ubound_reflector::ub
```

Definition at line 381 of file packing.hpp.

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

- [packing.hpp](#)

6.174 upcxx::packing_unpack_reflector< member_assignment_not_construction > Struct Template Reference

```
#include <packing.hpp>
```

6.174.1 Detailed Description

```
template<bool member_assignment_not_construction>
struct upcxx::packing_unpack_reflector< member_assignment_not_construction >
```

Definition at line 409 of file packing.hpp.

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

- [packing.hpp](#)

6.175 upcxx::packing_unpack_reflector< false > Struct Template Reference

```
#include <packing.hpp>
```

Public Member Functions

- `template<typename T >`
`void operator() (T &mbr)`
- `template<typename T >`
`void opaque (T &mbr)`

Public Attributes

- `parcel_reader & r`

6.175.1 Detailed Description

```
template<>
struct upcxx::packing_unpack_reflector< false >
```

Definition at line 425 of file packing.hpp.

6.175.2 Member Function Documentation

6.175.2.1 opaque()

```
template<typename T >
void upcxx::packing_unpack_reflector< false >::opaque (
    T & mbr ) [inline]
```

Definition at line 433 of file packing.hpp.

6.175.2.2 operator()

```
template<typename T >
void upcxx::packing_unpack_reflector< false >::operator() (
    T & mbr ) [inline]
```

Definition at line 429 of file packing.hpp.

6.175.3 Member Data Documentation

6.175.3.1 r

```
parcel_reader& upcxx::packing_unpack_reflector< false >::r
```

Definition at line 426 of file packing.hpp.

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

- [packing.hpp](#)

6.176 upcxx::packing_unpack_reflector< true > Struct Template Reference

```
#include <packing.hpp>
```

Public Member Functions

- `template<typename T >`
void [operator\(\)](#) (T &mbr)
- `template<typename T >`
void [opaque](#) (T &mbr)

Public Attributes

- [parcel_reader](#) &r

6.176.1 Detailed Description

```
template<>
struct upcxx::packing_unpack_reflector< true >
```

Definition at line 412 of file packing.hpp.

6.176.2 Member Function Documentation

6.176.2.1 opaque()

```
template<typename T >
void upcxx::packing_unpack_reflector< true >::opaque (
    T & mbr ) [inline]
```

Definition at line 420 of file packing.hpp.

6.176.2.2 operator()

```
template<typename T >
void upcxx::packing_unpack_reflector< true >::operator() (
    T & mbr ) [inline]
```

Definition at line 416 of file packing.hpp.

6.176.3 Member Data Documentation

6.176.3.1 r

```
parcel_reader& upcxx::packing_unpack_reflector< true >::r
```

Definition at line 413 of file packing.hpp.

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

- [packing.hpp](#)

6.177 upcxx::parcel_layout Class Reference

```
#include <packing.hpp>
```

Public Member Functions

- constexpr [parcel_layout](#) (std::size_t [size](#)=0, std::size_t align=1)
- std::size_t [size](#) () const
- std::size_t [alignment](#) () const
- std::size_t [size_aligned](#) () const
- template<std::size_t align = 1>
std::size_t [add_bytes](#) (std::size_t [size](#))
- std::size_t [add_bytes](#) (std::size_t [size](#), std::size_t align)
- template<typename T >
std::size_t [add_trivial_aligned](#) ()
- template<typename T >
std::size_t [add_trivial_unaligned](#) ()
- std::size_t [embed](#) ([parcel_layout](#) that)

Friends

- std::ostream & [operator<<](#) (std::ostream &o, const [parcel_layout](#) &x)

6.177.1 Detailed Description

Definition at line 42 of file packing.hpp.

6.177.2 Constructor & Destructor Documentation

6.177.2.1 parcel_layout()

```
constexpr upcxx::parcel_layout::parcel_layout (
    std::size_t size = 0,
    std::size_t align = 1 ) [inline]
```

Definition at line 47 of file packing.hpp.

6.177.3 Member Function Documentation

6.177.3.1 add_bytes() [1/2]

```
template<std::size_t align = 1>
std::size_t upcxx::parcel_layout::add_bytes (
    std::size_t size ) [inline]
```

Definition at line 58 of file packing.hpp.

6.177.3.2 add_bytes() [2/2]

```
std::size_t upcxx::parcel_layout::add_bytes (
    std::size_t size,
    std::size_t align ) [inline]
```

Definition at line 65 of file packing.hpp.

6.177.3.3 add_trivial_aligned()

```
template<typename T >
std::size_t upcxx::parcel_layout::add_trivial_aligned ( ) [inline]
```

Definition at line 73 of file packing.hpp.

6.177.3.4 add_trivial_unaligned()

```
template<typename T >  
std::size_t upcxx::parcel_layout::add_trivial_unaligned ( ) [inline]
```

Definition at line 77 of file packing.hpp.

6.177.3.5 alignment()

```
std::size_t upcxx::parcel_layout::alignment ( ) const [inline]
```

Definition at line 53 of file packing.hpp.

6.177.3.6 embed()

```
std::size_t upcxx::parcel_layout::embed (   
    parcel_layout that ) [inline]
```

Definition at line 81 of file packing.hpp.

6.177.3.7 size()

```
std::size_t upcxx::parcel_layout::size ( ) const [inline]
```

Definition at line 52 of file packing.hpp.

6.177.3.8 size_aligned()

```
std::size_t upcxx::parcel_layout::size_aligned ( ) const [inline]
```

Definition at line 55 of file packing.hpp.

6.177.4 Friends And Related Function Documentation

6.177.4.1 operator<<

```
std::ostream& operator<< (
    std::ostream & o,
    const parcel\_layout & x ) [friend]
```

Definition at line 89 of file `packing.hpp`.

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

- [packing.hpp](#)

6.178 `upcxx::parcel_reader` Class Reference

```
#include <packing.hpp>
```

Public Member Functions

- [parcel_reader](#) (void const *buf, std::size_t offset=0)
- [~parcel_reader](#) ()
- char const * [buffer](#) () const
- [parcel_layout](#) layout () const
- char const * [pop](#) (std::size_t size, std::size_t align)
- char [pop_char](#) ()
- std::uint8_t [pop_uint8](#) ()
- std::int8_t [pop_int8](#) ()
- template<typename T >
T const & [pop_trivial_aligned](#) ()
- template<typename T >
T const * [pop_trivial_aligned](#) (std::size_t n)
- template<typename T >
T [pop_trivial_unaligned](#) ()
- template<typename T >
char const * [pop_trivial_unaligned](#) (std::size_t n)

6.178.1 Detailed Description

Definition at line 187 of file `packing.hpp`.

6.178.2 Constructor & Destructor Documentation

6.178.2.1 parcel_reader()

```
upcxx::parcel_reader::parcel_reader (
    void const * buf,
    std::size_t offset = 0 ) [inline]
```

Definition at line 192 of file packing.hpp.

6.178.2.2 ~parcel_reader()

```
upcxx::parcel_reader::~~parcel_reader ( ) [inline]
```

Definition at line 196 of file packing.hpp.

6.178.3 Member Function Documentation

6.178.3.1 buffer()

```
char const* upcxx::parcel_reader::buffer ( ) const [inline]
```

Definition at line 200 of file packing.hpp.

6.178.3.2 layout()

```
parcel\_layout upcxx::parcel_reader::layout ( ) const [inline]
```

Definition at line 201 of file packing.hpp.

6.178.3.3 pop()

```
char const* upcxx::parcel_reader::pop (
    std::size_t size,
    std::size_t align ) [inline]
```

Definition at line 203 of file packing.hpp.

6.178.3.4 pop_char()

```
char upcxx::parcel_reader::pop_char ( ) [inline]
```

Definition at line 208 of file packing.hpp.

6.178.3.5 pop_int8()

```
std::int8_t upcxx::parcel_reader::pop_int8 ( ) [inline]
```

Definition at line 218 of file packing.hpp.

6.178.3.6 pop_trivial_aligned() [1/2]

```
template<typename T >  
T const& upcxx::parcel_reader::pop_trivial_aligned ( ) [inline]
```

Definition at line 225 of file packing.hpp.

6.178.3.7 pop_trivial_aligned() [2/2]

```
template<typename T >  
T const* upcxx::parcel_reader::pop_trivial_aligned (   
    std::size_t n ) [inline]
```

Definition at line 230 of file packing.hpp.

6.178.3.8 pop_trivial_unaligned() [1/2]

```
template<typename T >  
T upcxx::parcel_reader::pop_trivial_unaligned ( ) [inline]
```

Definition at line 236 of file packing.hpp.

6.178.3.9 pop_trivial_unaligned() [2/2]

```
template<typename T >  
char const* upcxx::parcel_reader::pop_trivial_unaligned (   
    std::size_t n ) [inline]
```

Definition at line 248 of file packing.hpp.

6.178.3.10 pop_uint8()

```
std::uint8_t upcxx::parcel_reader::pop_uint8 ( ) [inline]
```

Definition at line 213 of file packing.hpp.

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

- [packing.hpp](#)

6.179 upcxx::parcel_writer Struct Reference

```
#include <packing.hpp>
```

Public Member Functions

- [parcel_writer](#) (void *buf, std::size_t offset=0)
- char * [buffer](#) () const
- [parcel_layout layout](#) () const
- std::size_t [size](#) () const
- std::size_t [alignment](#) () const
- char * [put](#) (std::size_t size, std::size_t align)
- char * [put_char](#) (char x)
- std::uint8_t * [put_uint8](#) (std::uint8_t x)
- std::int8_t * [put_int8](#) (std::int8_t x)
- template<typename T >
T * [put_trivial_aligned](#) (const T &x)
- template<typename T >
T * [put_trivial_aligned](#) (const T *x, std::size_t n)
- template<typename T >
void * [put_trivial_unaligned](#) (const T &x)
- template<typename T >
void * [put_trivial_unaligned](#) (const T *x, std::size_t n)

Public Attributes

- char * __restrict [buf_](#)
- [parcel_layout lay_](#)

6.179.1 Detailed Description

Definition at line 99 of file packing.hpp.

6.179.2 Constructor & Destructor Documentation

6.179.2.1 parcel_writer()

```
upcxx::parcel_writer::parcel_writer (
    void * buf,
    std::size_t offset = 0 ) [inline]
```

Definition at line 104 of file packing.hpp.

6.179.3 Member Function Documentation

6.179.3.1 alignment()

```
std::size_t upcxx::parcel_writer::alignment ( ) const [inline]
```

Definition at line 113 of file packing.hpp.

6.179.3.2 buffer()

```
char* upcxx::parcel_writer::buffer ( ) const [inline]
```

Definition at line 109 of file packing.hpp.

6.179.3.3 layout()

```
parcel\_layout upcxx::parcel_writer::layout ( ) const [inline]
```

Definition at line 110 of file packing.hpp.

6.179.3.4 put()

```
char* upcxx::parcel_writer::put (
    std::size_t size,
    std::size_t align ) [inline]
```

Definition at line 115 of file packing.hpp.

6.179.3.5 put_char()

```
char* upcxx::parcel_writer::put_char (  
    char x ) [inline]
```

Definition at line 120 of file packing.hpp.

6.179.3.6 put_int8()

```
std::int8_t* upcxx::parcel_writer::put_int8 (  
    std::int8_t x ) [inline]
```

Definition at line 130 of file packing.hpp.

6.179.3.7 put_trivial_aligned() [1/2]

```
template<typename T >  
T* upcxx::parcel_writer::put_trivial_aligned (  
    const T & x ) [inline]
```

Definition at line 137 of file packing.hpp.

6.179.3.8 put_trivial_aligned() [2/2]

```
template<typename T >  
T* upcxx::parcel_writer::put_trivial_aligned (  
    const T * x,  
    std::size_t n ) [inline]
```

Definition at line 145 of file packing.hpp.

6.179.3.9 put_trivial_unaligned() [1/2]

```
template<typename T >  
void* upcxx::parcel_writer::put_trivial_unaligned (  
    const T & x ) [inline]
```

Definition at line 158 of file packing.hpp.

6.179.3.10 put_trivial_unaligned() [2/2]

```
template<typename T >
void* upcxx::parcel_writer::put_trivial_unaligned (
    const T * x,
    std::size_t n ) [inline]
```

Definition at line 165 of file packing.hpp.

6.179.3.11 put_uint8()

```
std::uint8_t* upcxx::parcel_writer::put_uint8 (
    std::uint8_t x ) [inline]
```

Definition at line 125 of file packing.hpp.

6.179.3.12 size()

```
std::size_t upcxx::parcel_writer::size ( ) const [inline]
```

Definition at line 112 of file packing.hpp.

6.179.4 Member Data Documentation

6.179.4.1 buf_

```
char* __restrict upcxx::parcel_writer::buf_
```

Definition at line 100 of file packing.hpp.

6.179.4.2 lay_

```
parcel\_layout upcxx::parcel_writer::lay_
```

Definition at line 101 of file packing.hpp.

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

- [packing.hpp](#)

6.180 `upcxx::print_proxy< T >` Struct Template Reference

```
#include <reflection.hpp>
```

Public Attributes

- `T` const & [subject](#)

Friends

- `std::ostream` & [operator<<](#) (`std::ostream` &`o`, [print_proxy](#) &`me`)

6.180.1 Detailed Description

```
template<typename T>  
struct upcxx::print_proxy< T >
```

Definition at line 252 of file `reflection.hpp`.

6.180.2 Friends And Related Function Documentation

6.180.2.1 `operator<<`

```
template<typename T >  
std::ostream& operator<< (  
    std::ostream & o,  
    print\_proxy< T > & me ) [friend]
```

Definition at line 255 of file `reflection.hpp`.

6.180.3 Member Data Documentation

6.180.3.1 `subject`

```
template<typename T >  
T const& upcxx::print\_proxy< T >::subject
```

Definition at line 253 of file `reflection.hpp`.

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

- [reflection.hpp](#)

6.181 upcxx::print_reflector Struct Reference

```
#include <reflection.hpp>
```

Public Member Functions

- template<typename T >
void [opaque](#) (const T &x)
- template<typename T >
void [operator\(\)](#) (const T &x)

Public Attributes

- std::ostream & [o](#)
- bool [comma](#)

6.181.1 Detailed Description

Definition at line 235 of file reflection.hpp.

6.181.2 Member Function Documentation

6.181.2.1 opaque()

```
template<typename T >  
void upcxx::print_reflector::opaque (  
    const T & x ) [inline]
```

Definition at line 240 of file reflection.hpp.

6.181.2.2 operator()

```
template<typename T >  
void upcxx::print_reflector::operator() (  
    const T & x ) [inline]
```

Definition at line 246 of file reflection.hpp.

6.181.3 Member Data Documentation

6.181.3.1 comma

```
bool upcxx::print_reflector::comma
```

Definition at line 237 of file reflection.hpp.

6.181.3.2 o

```
std::ostream& upcxx::print_reflector::o
```

Definition at line 236 of file reflection.hpp.

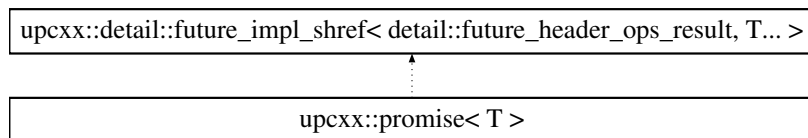
The documentation for this struct was generated from the following file:

- [reflection.hpp](#)

6.182 upcxx::promise< T > Class Template Reference

```
#include <promise.hpp>
```

Inheritance diagram for upcxx::promise< T >:



6.182.1 Detailed Description

```
template<typename ... T>
class upcxx::promise< T >
```

Definition at line 11 of file promise.hpp.

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

- [future/promise.hpp](#)

6.183 upcxx::promise_cx< T > Struct Template Reference

```
#include <completion.hpp>
```

Public Attributes

- [promise< T... > & pro_](#)

6.183.1 Detailed Description

```
template<typename ... T>
struct upcxx::promise_cx< T >
```

Definition at line 20 of file completion.hpp.

6.183.2 Member Data Documentation

6.183.2.1 pro_

```
template<typename ... T>
promise<T...>& upcxx::promise\_cx< T >::pro\_
```

Definition at line 21 of file completion.hpp.

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

- [completion.hpp](#)

6.184 upcxx::detail::promise_like< Fu > Struct Template Reference

```
#include <promise.hpp>
```

6.184.1 Detailed Description

```
template<typename Fu>
struct upcxx::detail::promise_like< Fu >
```

Definition at line 19 of file promise.hpp.

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

- [future/promise.hpp](#)

6.185 upcxx::detail::promise_like< future1< Kind, T... > > Struct Template Reference

```
#include <promise.hpp>
```

Public Types

- using `type` = `promise`< T... >

6.185.1 Detailed Description

```
template<typename Kind, typename ... T>
struct upcxx::detail::promise_like< future1< Kind, T... > >
```

Definition at line 21 of file `promise.hpp`.

6.185.2 Member Typedef Documentation

6.185.2.1 `type`

```
template<typename Kind , typename ... T>
using upcxx::detail::promise_like< future1< Kind, T... > >::type = promise<T...>
```

Definition at line 22 of file `promise.hpp`.

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

- [future/promise.hpp](#)

6.186 upcxx::reflection< Subject > Struct Template Reference

```
#include <reflection.hpp>
```

Public Member Functions

- template<typename Reflector , typename Subject1 >
void `operator()` (Reflector &re, Subject1 &&subj)

6.186.1 Detailed Description

```
template<typename Subject>
struct upcxx::reflection< Subject >
```

Definition at line 42 of file `reflection.hpp`.

6.186.2 Member Function Documentation

6.186.2.1 operator()

```
template<typename Subject >
template<typename Reflector , typename Subject1 >
void upcxx::reflection< Subject >::operator() (
    Reflector & re,
    Subject1 && subj ) [inline]
```

Definition at line 44 of file reflection.hpp.

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

- [reflection.hpp](#)

6.187 upcxx::reflection< std::array< T, n > > Struct Template Reference

```
#include <reflection.hpp>
```

Public Member Functions

- template<typename Re , typename Array >
void [operator\(\)](#) (Re &re, Array &&x) const

6.187.1 Detailed Description

```
template<typename T, std::size_t n>
struct upcxx::reflection< std::array< T, n > >
```

Definition at line 221 of file reflection.hpp.

6.187.2 Member Function Documentation

6.187.2.1 operator()

```
template<typename T , std::size_t n>
template<typename Re , typename Array >
void upcxx::reflection< std::array< T, n > >::operator() (
    Re & re,
    Array && x ) const [inline]
```

Definition at line 223 of file reflection.hpp.

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

- [reflection.hpp](#)

6.188 `upcxx::reflection< std::pair< A, B > >` Struct Template Reference

```
#include <reflection.hpp>
```

Public Member Functions

- `template<typename Re , typename Pair >`
`void operator() (Re &re, Pair &&x) const`

6.188.1 Detailed Description

```
template<typename A, typename B>  
struct upcxx::reflection< std::pair< A, B > >
```

Definition at line 208 of file `reflection.hpp`.

6.188.2 Member Function Documentation

6.188.2.1 `operator()`

```
template<typename A , typename B >  
template<typename Re , typename Pair >  
void upcxx::reflection< std::pair< A, B > >::operator() (  
    Re & re,  
    Pair && x ) const [inline]
```

Definition at line 210 of file `reflection.hpp`.

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

- [reflection.hpp](#)

6.189 `upcxx::reflection< std::tuple< Ts... > >` Struct Template Reference

```
#include <reflection.hpp>
```

Public Member Functions

- `template<typename Re , typename Tup1 >`
`void operator() (Re &re, Tup1 &&x) const`

6.189.1 Detailed Description

```
template<typename ... Ts>
struct upcxx::reflection< std::tuple< Ts... > >
```

Definition at line 196 of file reflection.hpp.

6.189.2 Member Function Documentation

6.189.2.1 operator()

```
template<typename ... Ts>
template<typename Re , typename Tup1 >
void upcxx::reflection< std::tuple< Ts... > >::operator() (
    Re & re,
    Tup1 && x ) const [inline]
```

Definition at line 198 of file reflection.hpp.

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

- [reflection.hpp](#)

6.190 upcxx::reflection_tuple< Tup, i, n > Struct Template Reference

```
#include <reflection.hpp>
```

Public Member Functions

- template<typename Re , typename Tup1 >
void [operator\(\)](#) (Re &re, Tup1 &&x) const

6.190.1 Detailed Description

```
template<typename Tup, int i, int n>
struct upcxx::reflection_tuple< Tup, i, n >
```

Definition at line 181 of file reflection.hpp.

6.190.2 Member Function Documentation

6.190.2.1 `operator()`

```
template<typename Tup , int i, int n>
template<typename Re , typename Tup1 >
void upcxx::reflection_tuple< Tup, i, n >::operator() (
    Re & re,
    Tup1 && x ) const [inline]
```

Definition at line 183 of file reflection.hpp.

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

- [reflection.hpp](#)

6.191 `upcxx::reflection_tuple< Tup, n, n >` Struct Template Reference

```
#include <reflection.hpp>
```

Public Member Functions

- `template<typename Re , typename Tup1 >`
`void operator() (Re &re, Tup1 &&x) const`

6.191.1 Detailed Description

```
template<typename Tup, int n>
struct upcxx::reflection_tuple< Tup, n, n >
```

Definition at line 189 of file reflection.hpp.

6.191.2 Member Function Documentation

6.191.2.1 `operator()`

```
template<typename Tup , int n>
template<typename Re , typename Tup1 >
void upcxx::reflection_tuple< Tup, n, n >::operator() (
    Re & re,
    Tup1 && x ) const [inline]
```

Definition at line 191 of file reflection.hpp.

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

- [reflection.hpp](#)

6.192 upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function Struct Reference

```
#include <impl_mapped.hpp>
```

Public Types

- typedef [apply_futured_as_future_return_t](#)< Fn(FuArg)> [fn_return_t](#)

Public Member Functions

- [result_lrefs_function](#) (fn_return_t result)
- [result_lrefs_function](#) (const [result_lrefs_function](#) &that)
- [result_lrefs_function](#) & [operator=](#) (const [result_lrefs_function](#) &that)
- [result_lrefs_function](#) ([result_lrefs_function](#) &&that)
- [result_lrefs_function](#) & [operator=](#) ([result_lrefs_function](#) &&that)
- auto [operator\(\)](#) () const -> [decltype\(getter_\(\)\)](#)

Public Attributes

- [decltype\(std::declval< fn_return_t >\(\).impl_.result_lrefs_getter\(\)\)](#) typedef [fn_return_getter_t](#)
- [fn_return_t](#) [result_](#)
- [fn_return_getter_t](#) [getter_](#)

6.192.1 Detailed Description

```
template<typename FuArg, typename Fn, typename ... T>
struct upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function
```

Definition at line 38 of file impl_mapped.hpp.

6.192.2 Member Typedef Documentation

6.192.2.1 fn_return_t

```
template<typename FuArg , typename Fn , typename ... T>
typedef apply\_futured\_as\_future\_return\_t<Fn(FuArg)> upcxx::detail::future\_impl\_mapped< FuArg,
Fn, T >::result\_lrefs\_function::fn\_return\_t
```

Definition at line 39 of file impl_mapped.hpp.

6.192.3 Constructor & Destructor Documentation

6.192.3.1 result_lrefs_function() [1/3]

```
template<typename FuArg , typename Fn , typename ... T>
upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function::result_lrefs_↵
function (
    fn_return_t result ) [inline]
```

Definition at line 45 of file impl_mapped.hpp.

6.192.3.2 result_lrefs_function() [2/3]

```
template<typename FuArg , typename Fn , typename ... T>
upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function::result_lrefs_↵
function (
    const result_lrefs_function & that ) [inline]
```

Definition at line 50 of file impl_mapped.hpp.

6.192.3.3 result_lrefs_function() [3/3]

```
template<typename FuArg , typename Fn , typename ... T>
upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function::result_lrefs_↵
function (
    result_lrefs_function && that ) [inline]
```

Definition at line 60 of file impl_mapped.hpp.

6.192.4 Member Function Documentation

6.192.4.1 operator>()

```
template<typename FuArg , typename Fn , typename ... T>
auto upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function::operator() ( )
const -> decltype(getter_()) [inline]
```

Definition at line 70 of file impl_mapped.hpp.

6.192.4.2 operator=() [1/2]

```
template<typename FuArg , typename Fn , typename ... T>
result_lrefs_function& upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_↵
function::operator= (
    const result_lrefs_function & that ) [inline]
```

Definition at line 54 of file impl_mapped.hpp.

6.192.4.3 operator=() [2/2]

```
template<typename FuArg , typename Fn , typename ... T>
result_lrefs_function& upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_↵
function::operator= (
    result_lrefs_function && that ) [inline]
```

Definition at line 64 of file impl_mapped.hpp.

6.192.5 Member Data Documentation**6.192.5.1 fn_return_getter_t**

```
template<typename FuArg , typename Fn , typename ... T>
decltype(std::declval<fn_return_t>().impl_.result_lrefs_getter()) typedef upcxx::detail↵
::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function::fn_return_getter_t
```

Definition at line 40 of file impl_mapped.hpp.

6.192.5.2 getter_

```
template<typename FuArg , typename Fn , typename ... T>
fn_return_getter_t upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function↵
::getter_
```

Definition at line 43 of file impl_mapped.hpp.

6.192.5.3 `result_`

```
template<typename FuArg , typename Fn , typename ... T>
fn_return_t upcxx::detail::future_impl_mapped< FuArg, Fn, T >::result_lrefs_function::result↔
-
```

Definition at line 42 of file `impl_mapped.hpp`.

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

- [future/impl_mapped.hpp](#)

6.193 `upcxx::detail::rget_byval< T >` Struct Template Reference

```
#include <rget.hpp>
```

6.193.1 Detailed Description

```
template<typename T>
struct upcxx::detail::rget_byval< T >
```

Definition at line 11 of file `rget.hpp`.

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

- [rget.hpp](#)

6.194 `upcxx::detail::rget_futures_of< Mode, R, O >` Struct Template Reference

```
#include <rget.hpp>
```

Public Types

- using `return_type` = void

Public Member Functions

- `rget_futures_of` (`rget_states`< Mode, R, O > &st)
- void `operator()` ()

6.194.1 Detailed Description

```
template<typename Mode, typename R, typename O>
struct upcxx::detail::rget_futures_of< Mode, R, O >
```

Definition at line 126 of file `rget.hpp`.

6.194.2 Member Typedef Documentation

6.194.2.1 return_type

```
template<typename Mode, typename R, typename O>
using upcxx::detail::rget_futures_of< Mode, R, O >::return_type = void
```

Definition at line 127 of file rget.hpp.

6.194.3 Constructor & Destructor Documentation

6.194.3.1 rget_futures_of()

```
template<typename Mode, typename R, typename O>
upcxx::detail::rget_futures_of< Mode, R, O >::rget_futures_of (
    rget_states< Mode, R, O > & st ) [inline]
```

Definition at line 129 of file rget.hpp.

6.194.4 Member Function Documentation

6.194.4.1 operator()

```
template<typename Mode, typename R, typename O>
void upcxx::detail::rget_futures_of< Mode, R, O >::operator() ( ) [inline]
```

Definition at line 131 of file rget.hpp.

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

- [rget.hpp](#)

6.195 upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > > Struct Template Reference

```
#include <rget.hpp>
```

Public Types

- using `return_type = future<>`

Public Member Functions

- `rget_futures_of` (`rget_states< rget_byref, R, future_cx< 0 >> &st`)
- `future operator()` (`()`)

Public Attributes

- `future fut_`

6.195.1 Detailed Description

```
template<typename R>
struct upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > >
```

Definition at line 135 of file rget.hpp.

6.195.2 Member Typedef Documentation

6.195.2.1 return_type

```
template<typename R >
using upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > >::return_type = future<>
```

Definition at line 136 of file rget.hpp.

6.195.3 Constructor & Destructor Documentation

6.195.3.1 rget_futures_of()

```
template<typename R >
upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > >::rget_futures_of (
    rget_states< rget_byref, R, future_cx< 0 >> & st ) [inline]
```

Definition at line 140 of file rget.hpp.

6.195.4 Member Function Documentation

6.195.4.1 operator()

```
template<typename R >
future upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > >::operator() ( ) [inline]
```

Definition at line 144 of file rget.hpp.

6.195.5 Member Data Documentation

6.195.5.1 fut_

```
template<typename R >
future upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > >::fut_
```

Definition at line 138 of file rget.hpp.

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

- [rget.hpp](#)

6.196 upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > > Struct Template Reference

```
#include <rget.hpp>
```

Public Types

- using `return_type` = `future< T >`

Public Member Functions

- `rget_futures_of` (`rget_states< rget_byval< T >, R, future_cx< 0 >> &st`)
- `future< T > operator()` ()

Public Attributes

- `future< T > fut_`

6.196.1 Detailed Description

```
template<typename T, typename R>
struct upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > >
```

Definition at line 148 of file rget.hpp.

6.196.2 Member Typedef Documentation

6.196.2.1 return_type

```
template<typename T , typename R >
using upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > >::return_type =
future<T>
```

Definition at line 149 of file rget.hpp.

6.196.3 Constructor & Destructor Documentation

6.196.3.1 rget_futures_of()

```
template<typename T , typename R >
upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > >::rget_futures_of (
    rget_states< rget_byval< T >, R, future_cx< 0 > > & st ) [inline]
```

Definition at line 153 of file rget.hpp.

6.196.4 Member Function Documentation

6.196.4.1 operator>()

```
template<typename T , typename R >
future<T> upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > >::operator()
( ) [inline]
```

Definition at line 157 of file rget.hpp.

6.196.5 Member Data Documentation

6.196.5.1 fut_

```
template<typename T , typename R >
future<T> upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > >::fut_
```

Definition at line 151 of file rget.hpp.

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

- [rget.hpp](#)

6.197 upcxx::detail::rget_state_operxn< Mode, Cx > Struct Template Reference

```
#include <rget.hpp>
```

6.197.1 Detailed Description

```
template<typename Mode, typename Cx>
struct upcxx::detail::rget_state_operxn< Mode, Cx >
```

Definition at line 14 of file rget.hpp.

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

- [rget.hpp](#)

6.198 upcxx::detail::rget_state_operxn< rget_byref, future_cx< ordinal > > Struct Template Reference

```
#include <rget.hpp>
```

Public Member Functions

- [rget_state_operxn](#) (future_cx< ordinal >)
- void [completed](#) ()

Public Attributes

- [promise pro](#)

6.198.1 Detailed Description

```
template<int ordinal>
struct upcxx::detail::rget_state_operxn< rget_byref, future_cx< ordinal > >
```

Definition at line 32 of file rget.hpp.

6.198.2 Constructor & Destructor Documentation

6.198.2.1 rget_state_operxn()

```
template<int ordinal>
upcxx::detail::rget_state_operxn< rget_byref, future_cx< ordinal > >::rget_state_operxn (
    future_cx< ordinal > ) [inline]
```

Definition at line 35 of file rget.hpp.

6.198.3 Member Function Documentation

6.198.3.1 completed()

```
template<int ordinal>
void upcxx::detail::rget_state_operxn< rget_byref, future_cx< ordinal > >::completed ( )
[inline]
```

Definition at line 37 of file rget.hpp.

6.198.4 Member Data Documentation

6.198.4.1 pro

```
template<int ordinal>
promise upcxx::detail::rget_state_operxn< rget_byref, future_cx< ordinal > >::pro
```

Definition at line 33 of file rget.hpp.

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

- [rget.hpp](#)

6.199 upcxx::detail::rget_state_operxn< rget_byref, nil_cx > Struct Template Reference

```
#include <rget.hpp>
```

Public Member Functions

- [rget_state_operxn \(nil_cx\)](#)
- void [completed](#) ()

6.199.1 Detailed Description

```
template<>  
struct upcxx::detail::rget_state_operxn< rget_byref, nil_cx >
```

Definition at line 19 of file rget.hpp.

6.199.2 Constructor & Destructor Documentation

6.199.2.1 rget_state_operxn()

```
upcxx::detail::rget_state_operxn< rget_byref, nil_cx >::rget_state_operxn (  
    nil_cx ) [inline]
```

Definition at line 20 of file rget.hpp.

6.199.3 Member Function Documentation

6.199.3.1 completed()

```
void upcxx::detail::rget_state_operxn< rget_byref, nil_cx >::completed ( ) [inline]
```

Definition at line 22 of file rget.hpp.

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

- [rget.hpp](#)

6.200 `upcxx::detail::rget_state_operxn< rget_byref, promise_cx<> >` Struct Template Reference

```
#include <rget.hpp>
```

Public Member Functions

- `rget_state_operxn` (`promise_cx<> &cx`)
- void `completed` ()

Public Attributes

- `promise * pro`

6.200.1 Detailed Description

```
template<>  
struct upcxx::detail::rget_state_operxn< rget_byref, promise_cx<> >
```

Definition at line 64 of file `rget.hpp`.

6.200.2 Constructor & Destructor Documentation

6.200.2.1 `rget_state_operxn()`

```
upcxx::detail::rget_state_operxn< rget_byref, promise_cx<> >::rget_state_operxn (  
    promise_cx<> & cx ) [inline]
```

Definition at line 67 of file `rget.hpp`.

6.200.3 Member Function Documentation

6.200.3.1 `completed()`

```
void upcxx::detail::rget_state_operxn< rget_byref, promise_cx<> >::completed ( ) [inline]
```

Definition at line 71 of file `rget.hpp`.

6.200.4 Member Data Documentation

6.200.4.1 pro

```
promise* upcxx::detail::rget_state_operxn< rget_byref, promise_cx<> >::pro
```

Definition at line 65 of file rget.hpp.

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

- [rget.hpp](#)

6.201 upcxx::detail::rget_state_operxn< rget_byval< T >, future_cx< ordinal > > Struct Template Reference

```
#include <rget.hpp>
```

Public Member Functions

- [rget_state_operxn](#) (future_cx< ordinal >)
- void [completed](#) (T *buf_d)

Public Attributes

- [promise](#)< T > [pro](#)

6.201.1 Detailed Description

```
template<typename T, int ordinal>  
struct upcxx::detail::rget_state_operxn< rget_byval< T >, future_cx< ordinal > >
```

Definition at line 43 of file rget.hpp.

6.201.2 Constructor & Destructor Documentation

6.201.2.1 rget_state_operxn()

```
template<typename T , int ordinal>
upcxx::detail::rget_state_operxn< rget_byval< T >, future_cx< ordinal > >::rget_state_operxn
(
    future_cx< ordinal > ) [inline]
```

Definition at line 46 of file rget.hpp.

6.201.3 Member Function Documentation

6.201.3.1 completed()

```
template<typename T , int ordinal>
void upcxx::detail::rget_state_operxn< rget_byval< T >, future_cx< ordinal > >::completed (
    T * buf_d ) [inline]
```

Definition at line 48 of file rget.hpp.

6.201.4 Member Data Documentation

6.201.4.1 pro

```
template<typename T , int ordinal>
promise<T> upcxx::detail::rget_state_operxn< rget_byval< T >, future_cx< ordinal > >::pro
```

Definition at line 44 of file rget.hpp.

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

- [rget.hpp](#)

6.202 upcxx::detail::rget_state_operxn< rget_byval< T >, nil_cx > Struct Template Reference

```
#include <rget.hpp>
```

Public Member Functions

- [rget_state_operxn](#) (nil_cx)
- void [completed](#) (T *buffer)

6.202.1 Detailed Description

```
template<typename T>
struct upcxx::detail::rget_state_operxn< rget_byval< T >, nil_cx >
```

Definition at line 25 of file rget.hpp.

6.202.2 Constructor & Destructor Documentation

6.202.2.1 rget_state_operxn()

```
template<typename T >
upcxx::detail::rget_state_operxn< rget_byval< T >, nil_cx >::rget_state_operxn (
    nil_cx ) [inline]
```

Definition at line 26 of file rget.hpp.

6.202.3 Member Function Documentation

6.202.3.1 completed()

```
template<typename T >
void upcxx::detail::rget_state_operxn< rget_byval< T >, nil_cx >::completed (
    T * buffer ) [inline]
```

Definition at line 28 of file rget.hpp.

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

- [rget.hpp](#)

6.203 upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T > > Struct Template Reference

```
#include <rget.hpp>
```

Public Member Functions

- [rget_state_operxn](#) ([promise_cx](#)<> &cx)
- void [completed](#) (T *buf_d)

Public Attributes

- [promise< T > * pro](#)

6.203.1 Detailed Description

```
template<typename T>
struct upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T > >
```

Definition at line 76 of file rget.hpp.

6.203.2 Constructor & Destructor Documentation

6.203.2.1 rget_state_operxn()

```
template<typename T >
upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T > >::rget_state_operxn (
    promise_cx<> & cx ) [inline]
```

Definition at line 79 of file rget.hpp.

6.203.3 Member Function Documentation

6.203.3.1 completed()

```
template<typename T >
void upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T > >::completed (
    T * buf_d ) [inline]
```

Definition at line 83 of file rget.hpp.

6.203.4 Member Data Documentation

6.203.4.1 pro

```
template<typename T >
promise<T>* upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T > >::pro
```

Definition at line 77 of file rget.hpp.

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

- [rget.hpp](#)

6.204 `upcxx::detail::rget_state_remote< Cx >` Struct Template Reference

```
#include <rget.hpp>
```

6.204.1 Detailed Description

```
template<typename Cx>  
struct upcxx::detail::rget_state_remote< Cx >
```

Definition at line 16 of file `rget.hpp`.

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

- [rget.hpp](#)

6.205 `upcxx::detail::rget_state_remote< nil_cx >` Struct Template Reference

```
#include <rget.hpp>
```

Public Member Functions

- [rget_state_remote](#) (`nil_cx`)
- void [completed](#) (`inrank_t` owner)

6.205.1 Detailed Description

```
template<>  
struct upcxx::detail::rget_state_remote< nil_cx >
```

Definition at line 93 of file `rget.hpp`.

6.205.2 Constructor & Destructor Documentation

6.205.2.1 `rget_state_remote()`

```
upcxx::detail::rget_state_remote< nil_cx >::rget_state_remote (  
    nil_cx ) [inline]
```

Definition at line 94 of file `rget.hpp`.

6.205.3 Member Function Documentation

6.205.3.1 completed()

```
void upcxx::detail::rget_state_remote< nil_cx >::completed (
    intrank_t owner ) [inline]
```

Definition at line 96 of file rget.hpp.

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

- [rget.hpp](#)

6.206 upcxx::detail::rget_state_remote< rpc_cx< Fn > > Struct Template Reference

```
#include <rget.hpp>
```

Public Member Functions

- [rget_state_remote](#) (rpc_cx< Fn > &cx)
- void [completed](#) (intrank_t owner)

Public Attributes

- Fn [fn](#)

6.206.1 Detailed Description

```
template<typename Fn>
struct upcxx::detail::rget_state_remote< rpc_cx< Fn > >
```

Definition at line 100 of file rget.hpp.

6.206.2 Constructor & Destructor Documentation

6.206.2.1 rget_state_remote()

```
template<typename Fn >
upcxx::detail::rget_state_remote< rpc_cx< Fn > >::rget_state_remote (
    rpc_cx< Fn > & cx ) [inline]
```

Definition at line 103 of file rget.hpp.

6.206.3 Member Function Documentation

6.206.3.1 completed()

```
template<typename Fn >
void upcxx::detail::rget_state_remote< rpc_cx< Fn > >::completed (
    intrank_t owner ) [inline]
```

Definition at line 107 of file rget.hpp.

6.206.4 Member Data Documentation

6.206.4.1 fn

```
template<typename Fn >
Fn upcxx::detail::rget_state_remote< rpc_cx< Fn > >::fn
```

Definition at line 101 of file rget.hpp.

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

- [rget.hpp](#)

6.207 upcxx::detail::rget_states< Mode, R, O > Struct Template Reference

```
#include <rget.hpp>
```

Public Member Functions

- [rget_states](#) (intrank_t owner, completions< nil_cx, R, O > &&cxs)

Public Attributes

- [intrank_t owner](#)
- [rget_state_remote](#)< R > r
- [rget_state_operxn](#)< Mode, O > o

6.207.1 Detailed Description

```
template<typename Mode, typename R, typename O>
struct upcxx::detail::rget_states< Mode, R, O >
```

Definition at line 113 of file rget.hpp.

6.207.2 Constructor & Destructor Documentation

6.207.2.1 rget_states()

```
template<typename Mode, typename R, typename O>
upcxx::detail::rget_states< Mode, R, O >::rget_states (
    intrank_t owner,
    completions< nil_cx, R, O > && cxs ) [inline]
```

Definition at line 118 of file rget.hpp.

6.207.3 Member Data Documentation

6.207.3.1 o

```
template<typename Mode, typename R, typename O>
rget_state_operxn<Mode,O> upcxx::detail::rget_states< Mode, R, O >::o
```

Definition at line 116 of file rget.hpp.

6.207.3.2 owner

```
template<typename Mode, typename R, typename O>
intrank_t upcxx::detail::rget_states< Mode, R, O >::owner
```

Definition at line 114 of file rget.hpp.

6.207.3.3 r

```
template<typename Mode, typename R, typename O>
rget_state_remote<R> upcxx::detail::rget_states< Mode, R, O >::r
```

Definition at line 115 of file rget.hpp.

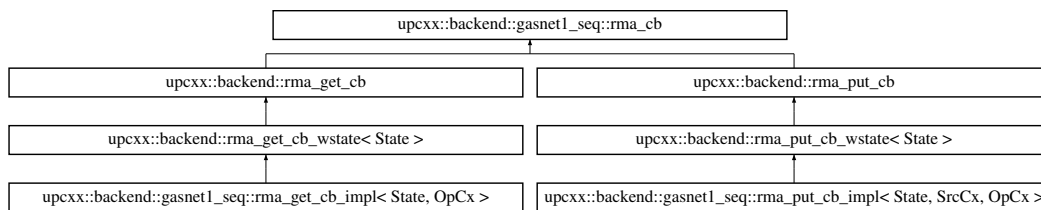
The documentation for this struct was generated from the following file:

- [rget.hpp](#)

6.208 upcxx::backend::gasnet1_seq::rma_cb Struct Reference

```
#include <backend.hpp>
```

Inheritance diagram for upcxx::backend::gasnet1_seq::rma_cb:



Public Member Functions

- virtual void [fire_and_delete](#) ()=0

Public Attributes

- [rma_cb](#) * next
- std::uintptr_t [handle](#)

6.208.1 Detailed Description

Definition at line 56 of file backend.hpp.

6.208.2 Member Function Documentation

6.208.2.1 fire_and_delete()

```
virtual void upcxx::backend::gasnet1_seq::rma_cb::fire_and_delete ( ) [pure virtual]
```

6.208.3 Member Data Documentation

6.208.3.1 handle

```
std::uintptr_t upcxx::backend::gasnet1_seq::rma_cb::handle
```

Definition at line 58 of file backend.hpp.

6.208.3.2 next

```
rma_cb* upcxx::backend::gasnet1_seq::rma_cb::next
```

Definition at line 57 of file backend.hpp.

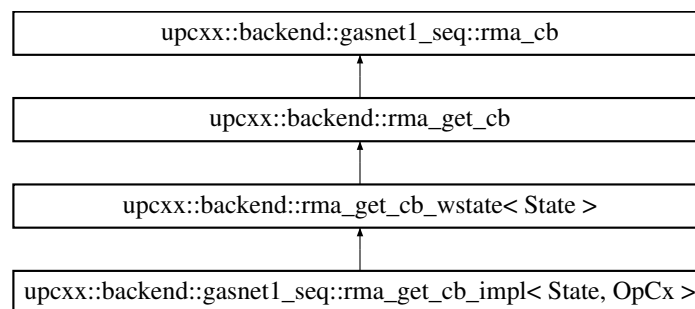
The documentation for this struct was generated from the following file:

- backend/gasnet1_seq/backend.hpp

6.209 upcxx::backend::rma_get_cb Struct Reference

```
#include <backend.hpp>
```

Inheritance diagram for upcxx::backend::rma_get_cb:



Additional Inherited Members

6.209.1 Detailed Description

Definition at line 161 of file backend.hpp.

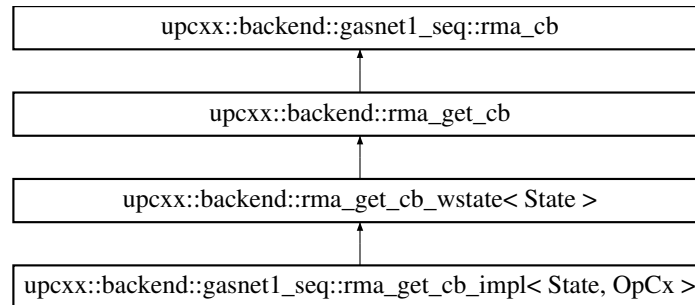
The documentation for this struct was generated from the following file:

- backend/gasnet1_seq/backend.hpp

6.210 upcxx::backend::gasnet1_seq::rma_get_cb_impl< State, OpCx > Struct Template Reference

```
#include <backend.hpp>
```

Inheritance diagram for upcxx::backend::gasnet1_seq::rma_get_cb_impl< State, OpCx >:



Public Member Functions

- [rma_get_cb_impl](#) (State &&state, OpCx &&op_cx)

Public Attributes

- [OpCx op_cx](#)

6.210.1 Detailed Description

```
template<typename State, typename OpCx>
struct upcxx::backend::gasnet1_seq::rma_get_cb_impl< State, OpCx >
```

Definition at line 197 of file backend.hpp.

6.210.2 Constructor & Destructor Documentation

6.210.2.1 rma_get_cb_impl()

```
template<typename State , typename OpCx >
upcxx::backend::gasnet1_seq::rma_get_cb_impl< State, OpCx >::rma_get_cb_impl (
    State && state,
    OpCx && op_cx ) [inline]
```

Definition at line 200 of file backend.hpp.

6.210.3 Member Data Documentation

6.210.3.1 op_cx

```
template<typename State , typename OpCx >
OpCx upcxx::backend::gasnet1_seq::rma_get_cb_impl< State, OpCx >::op_cx
```

Definition at line 198 of file backend.hpp.

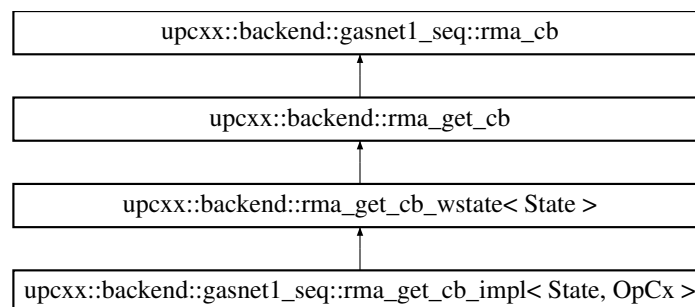
The documentation for this struct was generated from the following file:

- [backend/gasnet1_seq/backend.hpp](#)

6.211 upcxx::backend::rma_get_cb_wstate< State > Struct Template Reference

```
#include <backend.hpp>
```

Inheritance diagram for upcxx::backend::rma_get_cb_wstate< State >:



Public Member Functions

- [rma_get_cb_wstate](#) (State &&st)

Public Attributes

- State [state](#)

6.211.1 Detailed Description

```
template<typename State>
struct upcxx::backend::rma_get_cb_wstate< State >
```

Definition at line 169 of file backend.hpp.

6.211.2 Constructor & Destructor Documentation

6.211.2.1 rma_get_cb_wstate()

```
template<typename State >
upcxx::backend::rma_get_cb_wstate< State >::rma_get_cb_wstate (
    State && st ) [inline]
```

Definition at line 171 of file backend.hpp.

6.211.3 Member Data Documentation

6.211.3.1 state

```
template<typename State >
State upcxx::backend::rma_get_cb_wstate< State >::state
```

Definition at line 170 of file backend.hpp.

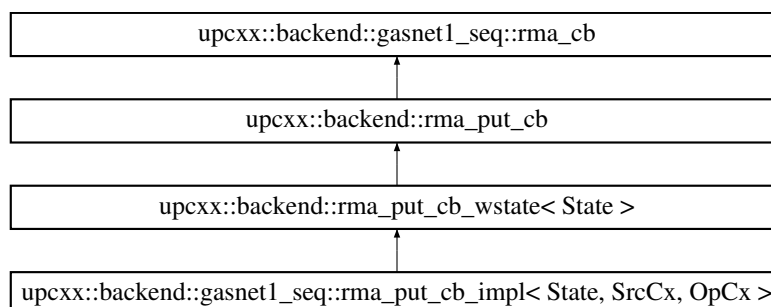
The documentation for this struct was generated from the following file:

- [backend/gasnet1_seq/backend.hpp](#)

6.212 upcxx::backend::rma_put_cb Struct Reference

```
#include <backend.hpp>
```

Inheritance diagram for upcxx::backend::rma_put_cb:



Additional Inherited Members

6.212.1 Detailed Description

Definition at line 160 of file backend.hpp.

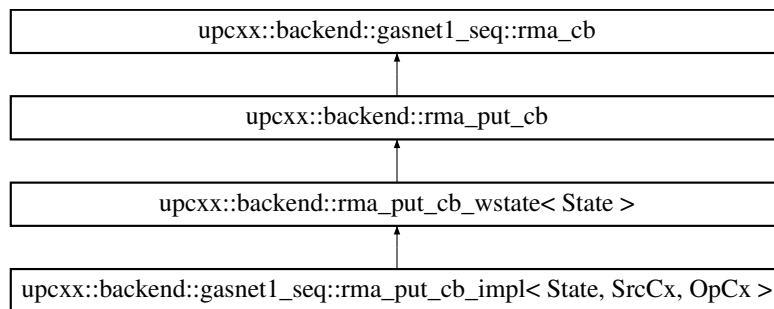
The documentation for this struct was generated from the following file:

- backend/gasnet1_seq/backend.hpp

6.213 upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx > Struct Template Reference

```
#include <backend.hpp>
```

Inheritance diagram for upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx >:



Public Member Functions

- [rma_put_cb_impl](#) (State &&state, SrcCx &&src_cx, OpCx &&op_cx)

Public Attributes

- SrcCx [src_cx](#)
- OpCx [op_cx](#)
- [src_cx](#) {std::move(src_cx)}

6.213.1 Detailed Description

```
template<typename State, typename SrcCx, typename OpCx>
struct upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx >
```

Definition at line 179 of file backend.hpp.

6.213.2 Constructor & Destructor Documentation

6.213.2.1 `rma_put_cb_impl()`

```
template<typename State , typename SrcCx , typename OpCx >
upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx >::rma_put_cb_impl (
    State && state,
    SrcCx && src_cx,
    OpCx && op_cx ) [inline]
```

Definition at line 183 of file backend.hpp.

6.213.3 Member Data Documentation

6.213.3.1 `op_cx`

```
template<typename State , typename SrcCx , typename OpCx >
OpCx upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx >::op_cx
```

Definition at line 181 of file backend.hpp.

6.213.3.2 `src_cx` [1/2]

```
template<typename State , typename SrcCx , typename OpCx >
SrcCx upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx >::src_cx
```

Definition at line 180 of file backend.hpp.

6.213.3.3 `src_cx` [2/2]

```
template<typename State , typename SrcCx , typename OpCx >
upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx >::src_cx {std::move(src_cx)}
```

Definition at line 185 of file backend.hpp.

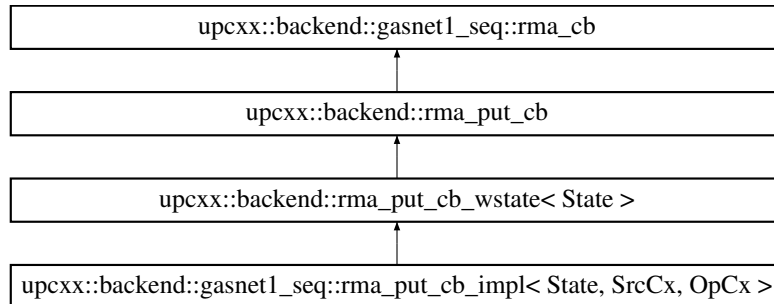
The documentation for this struct was generated from the following file:

- [backend/gasnet1_seq/backend.hpp](#)

6.214 upcxx::backend::rma_put_cb_wstate< State > Struct Template Reference

```
#include <backend.hpp>
```

Inheritance diagram for upcxx::backend::rma_put_cb_wstate< State >:



Public Member Functions

- [rma_put_cb_wstate](#) (State &&st)

Public Attributes

- State [state](#)

6.214.1 Detailed Description

```
template<typename State>
struct upcxx::backend::rma_put_cb_wstate< State >
```

Definition at line 164 of file backend.hpp.

6.214.2 Constructor & Destructor Documentation

6.214.2.1 rma_put_cb_wstate()

```
template<typename State >
upcxx::backend::rma_put_cb_wstate< State >::rma_put_cb_wstate (
    State && st ) [inline]
```

Definition at line 166 of file backend.hpp.

6.214.3 Member Data Documentation

6.214.3.1 state

```
template<typename State >  
State upcxx::backend::rma_put_cb_wstate< State >::state
```

Definition at line 165 of file backend.hpp.

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

- [backend/gasnet1_seq/backend.hpp](#)

6.215 upcxx::rpc_cx< Fn > Struct Template Reference

```
#include <completion.hpp>
```

Public Member Functions

- [rpc_cx](#) (Fn fn)

Public Attributes

- Fn [fn_](#)

6.215.1 Detailed Description

```
template<typename Fn>  
struct upcxx::rpc_cx< Fn >
```

Definition at line 11 of file completion.hpp.

6.215.2 Constructor & Destructor Documentation

6.215.2.1 rpc_cx()

```
template<typename Fn>  
upcxx::rpc_cx< Fn >::rpc_cx (  
    Fn fn ) [inline]
```

Definition at line 13 of file completion.hpp.

6.215.3 Member Data Documentation

6.215.3.1 fn_

```
template<typename Fn>
Fn upcxx::rpc_cx< Fn >::fn_
```

Definition at line 12 of file completion.hpp.

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

- [completion.hpp](#)

6.216 upcxx::detail::rpc_recipient_after< Pro > Struct Template Reference

```
#include <rpc.hpp>
```

Public Member Functions

- template<typename ... Args>
void [operator\(\)](#) (Args &&...args)

Public Attributes

- [inrank_t initiator_](#)
- Pro * [pro_](#)

6.216.1 Detailed Description

```
template<typename Pro>
struct upcxx::detail::rpc_recipient_after< Pro >
```

Definition at line 25 of file rpc.hpp.

6.216.2 Member Function Documentation

6.216.2.1 operator()

```
template<typename Pro >
template<typename ... Args>
void upcxx::detail::rpc_recipient_after< Pro >::operator() (
    Args &&... args ) [inline]
```

Definition at line 30 of file rpc.hpp.

6.216.3 Member Data Documentation

6.216.3.1 initiator_

```
template<typename Pro >
inrank_t upcxx::detail::rpc_recipient_after< Pro >::initiator_
```

Definition at line 26 of file rpc.hpp.

6.216.3.2 pro_

```
template<typename Pro >
Pro* upcxx::detail::rpc_recipient_after< Pro >::pro_
```

Definition at line 27 of file rpc.hpp.

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

- [rpc.hpp](#)

6.217 upcxx::detail::rpc_return< ValidType, Fn, Args > Struct Template Reference

```
#include <rpc.hpp>
```

Public Types

- using [type](#) = typename [detail::future_from_tuple_t](#)< [detail::future_kind_shref](#)< [detail::future_header_ops](#)↔
[_result](#) >, typename decltype([upcxx::apply_tupled_as_future](#)([upcxx::bind](#)(std::declval< Fn && >(), std↔
::declval< Args && >()...), std::tuple<>{}))::results_type >

6.217.1 Detailed Description

```
template<typename ValidType, typename Fn, typename ... Args>
struct upcxx::detail::rpc_return< ValidType, Fn, Args >
```

Definition at line 53 of file rpc.hpp.

6.217.2 Member Typedef Documentation

6.217.2.1 type

```
template<typename ValidType , typename Fn , typename ... Args>
using upcxx::detail::rpc_return< ValidType, Fn, Args >::type = typename detail::future_from←
_tuple_t< detail::future_kind_shref<detail::future_header_ops_result>, typename decltype(
upcxx::apply_tupled_as_future( upcxx::bind(std::declval<Fn&&>(), std::declval<Args&&>()...),
std::tuple<>{} ) )::results_type >
```

Definition at line 62 of file `rpc.hpp`.

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

- [rpc.hpp](#)

6.218 `upcxx::detail::rput_byval< T >` Struct Template Reference

```
#include <rput.hpp>
```

6.218.1 Detailed Description

```
template<typename T>
struct upcxx::detail::rput_byval< T >
```

Definition at line 11 of file `rput.hpp`.

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

- [rput.hpp](#)

6.219 `upcxx::detail::rput_futures_of< S, R, O >` Struct Template Reference

```
#include <rput.hpp>
```

Public Types

- using `return_type` = void

Public Member Functions

- template<typename Mode >
`rput_futures_of` (`rput_states`< Mode, S, R, O > &st)
- void `operator`() ()

6.219.1 Detailed Description

```
template<typename S, typename R, typename O>  
struct upcxx::detail::rput_futures_of< S, R, O >
```

Definition at line 98 of file rput.hpp.

6.219.2 Member Typedef Documentation

6.219.2.1 return_type

```
template<typename S, typename R, typename O>  
using upcxx::detail::rput_futures_of< S, R, O >::return_type = void
```

Definition at line 99 of file rput.hpp.

6.219.3 Constructor & Destructor Documentation

6.219.3.1 rput_futures_of()

```
template<typename S, typename R, typename O>  
template<typename Mode >  
upcxx::detail::rput_futures_of< S, R, O >::rput_futures_of (  
    rput_states< Mode, S, R, O > & st ) [inline]
```

Definition at line 102 of file rput.hpp.

6.219.4 Member Function Documentation

6.219.4.1 operator()

```
template<typename S, typename R, typename O>  
void upcxx::detail::rput_futures_of< S, R, O >::operator() ( ) [inline]
```

Definition at line 104 of file rput.hpp.

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

- [rput.hpp](#)

6.220 upcxx::detail::rput_futures_of< future_cx< S_ord >, R, future_cx< O_ord > > Struct Template Reference

```
#include <rput.hpp>
```

Public Types

- using `return_type` = `std::tuple< future<>, future<> >`

Public Member Functions

- `template<typename Mode > rput_futures_of (rput_states< Mode, future_cx< S_ord >, R, future_cx< O_ord >> &st)`
- `std::tuple< future<>, future<> > operator() ()`

Public Attributes

- `future s_`
- `future o_`

6.220.1 Detailed Description

```
template<int S_ord, typename R, int O_ord>  
struct upcxx::detail::rput_futures_of< future_cx< S_ord >, R, future_cx< O_ord > >
```

Definition at line 136 of file rput.hpp.

6.220.2 Member Typedef Documentation

6.220.2.1 return_type

```
template<int S_ord, typename R , int O_ord>  
using upcxx::detail::rput_futures_of< future_cx< S_ord >, R, future_cx< O_ord > >::return_↵  
type = std::tuple<future<>,future<> >
```

Definition at line 137 of file rput.hpp.

6.220.3 Constructor & Destructor Documentation

6.220.3.1 rput_futures_of()

```
template<int S_ord, typename R , int O_ord>
template<typename Mode >
upcxx::detail::rput_futures_of< future_cx< S_ord > , R, future_cx< O_ord > >::rput_futures_of
(
    rput_states< Mode, future_cx< S_ord > , R, future_cx< O_ord >> & st ) [inline]
```

Definition at line 142 of file rput.hpp.

6.220.4 Member Function Documentation

6.220.4.1 operator()

```
template<int S_ord, typename R , int O_ord>
std::tuple<future<>, future<> > upcxx::detail::rput_futures_of< future_cx< S_ord > , R,
future_cx< O_ord > >::operator() ( ) [inline]
```

Definition at line 147 of file rput.hpp.

6.220.5 Member Data Documentation

6.220.5.1 o_

```
template<int S_ord, typename R , int O_ord>
future upcxx::detail::rput_futures_of< future_cx< S_ord > , R, future_cx< O_ord > >::o_
```

Definition at line 139 of file rput.hpp.

6.220.5.2 s_

```
template<int S_ord, typename R , int O_ord>
future upcxx::detail::rput_futures_of< future_cx< S_ord > , R, future_cx< O_ord > >::s_
```

Definition at line 139 of file rput.hpp.

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

- [rput.hpp](#)

6.221 upcxx::detail::rput_futures_of< future_cx< S_ord >, R, O > Struct Template Reference

```
#include <rput.hpp>
```

Public Types

- using `return_type` = `future<>`

Public Member Functions

- `template<typename Mode > rput_futures_of (rput_states< Mode, future_cx< S_ord >, R, O > &st)`
- `future operator() ()`

Public Attributes

- `future fut_`

6.221.1 Detailed Description

```
template<typename R, typename O, int S_ord>  
struct upcxx::detail::rput_futures_of< future_cx< S_ord >, R, O >
```

Definition at line 108 of file rput.hpp.

6.221.2 Member Typedef Documentation

6.221.2.1 return_type

```
template<typename R , typename O , int S_ord>  
using upcxx::detail::rput_futures_of< future_cx< S_ord >, R, O >::return_type = future<>
```

Definition at line 109 of file rput.hpp.

6.221.3 Constructor & Destructor Documentation

6.221.3.1 rput_futures_of()

```
template<typename R , typename O , int S_ord>
template<typename Mode >
upcxx::detail::rput_futures_of< future_cx< S_ord > , R, O >::rput_futures_of (
    rput_states< Mode, future_cx< S_ord > , R, O > & st ) [inline]
```

Definition at line 114 of file rput.hpp.

6.221.4 Member Function Documentation

6.221.4.1 operator()

```
template<typename R , typename O , int S_ord>
future upcxx::detail::rput_futures_of< future_cx< S_ord > , R, O >::operator() ( ) [inline]
```

Definition at line 118 of file rput.hpp.

6.221.5 Member Data Documentation

6.221.5.1 fut_

```
template<typename R , typename O , int S_ord>
future upcxx::detail::rput_futures_of< future_cx< S_ord > , R, O >::fut_
```

Definition at line 111 of file rput.hpp.

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

- [rput.hpp](#)

6.222 upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > > Struct Template Reference

```
#include <rput.hpp>
```

Public Types

- using [return_type](#) = [future](#)<>

Public Member Functions

- `template<typename Mode > rput_futures_of (rput_states< Mode, S, R, future_cx< O_ord >> &st)`
- `future operator() ()`

Public Attributes

- `future fut_`

6.222.1 Detailed Description

```
template<typename S, typename R, int O_ord>
struct upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > >
```

Definition at line 122 of file rput.hpp.

6.222.2 Member Typedef Documentation

6.222.2.1 return_type

```
template<typename S , typename R , int O_ord>
using upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > >::return_type = future<>
```

Definition at line 123 of file rput.hpp.

6.222.3 Constructor & Destructor Documentation

6.222.3.1 rput_futures_of()

```
template<typename S , typename R , int O_ord>
template<typename Mode >
upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > >::rput_futures_of (
    rput_states< Mode, S, R, future_cx< O_ord >> & st ) [inline]
```

Definition at line 128 of file rput.hpp.

6.222.4 Member Function Documentation

6.222.4.1 operator()

```
template<typename S , typename R , int O_ord>
future upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > >::operator() ( ) [inline]
```

Definition at line 132 of file rput.hpp.

6.222.5 Member Data Documentation

6.222.5.1 fut_

```
template<typename S , typename R , int O_ord>
future upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > >::fut_
```

Definition at line 125 of file rput.hpp.

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

- [rput.hpp](#)

6.223 upcxx::detail::rput_state_here< Cx > Struct Template Reference

```
#include <rput.hpp>
```

6.223.1 Detailed Description

```
template<typename Cx>
struct upcxx::detail::rput_state_here< Cx >
```

Definition at line 14 of file rput.hpp.

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

- [rput.hpp](#)

6.224 upcxx::detail::rput_state_here< future_cx< ordinal > > Struct Template Reference

```
#include <rput.hpp>
```

Public Member Functions

- [rput_state_here](#) (future_cx< ordinal >)
- void [completed](#) ()

Public Attributes

- [promise pro](#)

6.224.1 Detailed Description

```
template<int ordinal>
struct upcxx::detail::rput_state_here< future_cx< ordinal > >
```

Definition at line 26 of file rput.hpp.

6.224.2 Constructor & Destructor Documentation

6.224.2.1 rput_state_here()

```
template<int ordinal>
upcxx::detail::rput_state_here< future_cx< ordinal > >::rput_state_here (
    future_cx< ordinal > ) [inline]
```

Definition at line 29 of file rput.hpp.

6.224.3 Member Function Documentation

6.224.3.1 completed()

```
template<int ordinal>
void upcxx::detail::rput_state_here< future_cx< ordinal > >::completed ( ) [inline]
```

Definition at line 31 of file rput.hpp.

6.224.4 Member Data Documentation

6.224.4.1 pro

```
template<int ordinal>
promise upcxx::detail::rput_state_here< future_cx< ordinal > >::pro
```

Definition at line 27 of file rput.hpp.

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

- [rput.hpp](#)

6.225 upcxx::detail::rput_state_here< nil_cx > Struct Template Reference

```
#include <rput.hpp>
```

Public Member Functions

- [rput_state_here](#) (nil_cx)
- void [completed](#) ()

6.225.1 Detailed Description

```
template<>
struct upcxx::detail::rput_state_here< nil_cx >
```

Definition at line 19 of file rput.hpp.

6.225.2 Constructor & Destructor Documentation

6.225.2.1 rput_state_here()

```
upcxx::detail::rput_state_here< nil_cx >::rput_state_here (
    nil_cx ) [inline]
```

Definition at line 20 of file rput.hpp.

6.225.3 Member Function Documentation

6.225.3.1 `completed()`

```
void upcxx::detail::rput_state_here< nil_cx >::completed ( ) [inline]
```

Definition at line 22 of file `rput.hpp`.

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

- [rput.hpp](#)

6.226 `upcxx::detail::rput_state_here< promise_cx<> >` Struct Template Reference

```
#include <rput.hpp>
```

Public Member Functions

- [rput_state_here](#) (`promise_cx<> &cx`)
- void [completed](#) ()

Public Attributes

- `promise * pro`

6.226.1 Detailed Description

```
template<>
struct upcxx::detail::rput_state_here< promise_cx<> >
```

Definition at line 37 of file `rput.hpp`.

6.226.2 Constructor & Destructor Documentation

6.226.2.1 `rput_state_here()`

```
upcxx::detail::rput_state_here< promise_cx<> >::rput_state_here (
    promise_cx<> & cx ) [inline]
```

Definition at line 40 of file `rput.hpp`.

6.226.3 Member Function Documentation

6.226.3.1 completed()

```
void upcxx::detail::rput_state_here< promise_cx<> >::completed ( ) [inline]
```

Definition at line 44 of file rput.hpp.

6.226.4 Member Data Documentation

6.226.4.1 pro

```
promise* upcxx::detail::rput_state_here< promise_cx<> >::pro
```

Definition at line 38 of file rput.hpp.

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

- [rput.hpp](#)

6.227 upcxx::detail::rput_state_remote< Cx > Struct Template Reference

```
#include <rput.hpp>
```

6.227.1 Detailed Description

```
template<typename Cx>
struct upcxx::detail::rput_state_remote< Cx >
```

Definition at line 16 of file rput.hpp.

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

- [rput.hpp](#)

6.228 upcxx::detail::rput_state_remote< nil_cx > Struct Template Reference

```
#include <rput.hpp>
```

Public Member Functions

- [rput_state_remote](#) (nil_cx)
- void [completed](#) (inrank_t target)

6.228.1 Detailed Description

```
template<>
struct upcxx::detail::rput_state_remote< nil_cx >
```

Definition at line 50 of file `rput.hpp`.

6.228.2 Constructor & Destructor Documentation

6.228.2.1 `rput_state_remote()`

```
upcxx::detail::rput_state_remote< nil_cx >::rput_state_remote (
    nil_cx ) [inline]
```

Definition at line 51 of file `rput.hpp`.

6.228.3 Member Function Documentation

6.228.3.1 `completed()`

```
void upcxx::detail::rput_state_remote< nil_cx >::completed (
    intrank_t target ) [inline]
```

Definition at line 53 of file `rput.hpp`.

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

- [rput.hpp](#)

6.229 `upcxx::detail::rput_state_remote< rpc_cx< Fn > >` Struct Template Reference

```
#include <rput.hpp>
```

Public Member Functions

- `rput_state_remote` (`rpc_cx< Fn > &&cx`)
- void `completed` (`intrank_t target`)

Public Attributes

- [Fn `fn`](#)

6.229.1 Detailed Description

```
template<typename Fn>
struct upcxx::detail::rput_state_remote< rpc_cx< Fn > >
```

Definition at line 57 of file `rput.hpp`.

6.229.2 Constructor & Destructor Documentation

6.229.2.1 `rput_state_remote()`

```
template<typename Fn >
upcxx::detail::rput_state_remote< rpc_cx< Fn > >::rput_state_remote (
    rpc_cx< Fn > && cx ) [inline]
```

Definition at line 60 of file `rput.hpp`.

6.229.3 Member Function Documentation

6.229.3.1 `completed()`

```
template<typename Fn >
void upcxx::detail::rput_state_remote< rpc_cx< Fn > >::completed (
    intrank_t target ) [inline]
```

Definition at line 64 of file `rput.hpp`.

6.229.4 Member Data Documentation

6.229.4.1 `fn`

```
template<typename Fn >
Fn upcxx::detail::rput_state_remote< rpc_cx< Fn > >::fn
```

Definition at line 58 of file `rput.hpp`.

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

- [rput.hpp](#)

6.230 `upcxx::detail::rput_states< Mode, S, R, O >` Struct Template Reference

```
#include <rput.hpp>
```

6.230.1 Detailed Description

```
template<typename Mode, typename S, typename R, typename O>
struct upcxx::detail::rput_states< Mode, S, R, O >
```

Definition at line 70 of file `rput.hpp`.

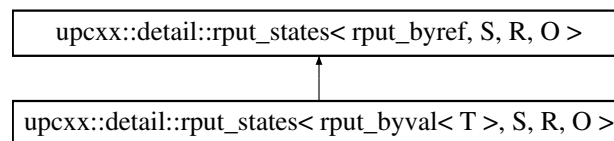
The documentation for this struct was generated from the following file:

- [rput.hpp](#)

6.231 `upcxx::detail::rput_states< rput_byref, S, R, O >` Struct Template Reference

```
#include <rput.hpp>
```

Inheritance diagram for `upcxx::detail::rput_states< rput_byref, S, R, O >`:



Public Member Functions

- [rput_states](#) (`inrank_t target, completions< S, R, O > &&cxs`)

Public Attributes

- [inrank_t target](#)
- [rput_state_here< S > s](#)
- [rput_state_remote< R > r](#)
- [rput_state_here< O > o](#)

6.231.1 Detailed Description

```
template<typename S, typename R, typename O>
struct upcxx::detail::rput_states< rput_byref, S, R, O >
```

Definition at line 73 of file `rput.hpp`.

6.231.2 Constructor & Destructor Documentation

6.231.2.1 rput_states()

```
template<typename S , typename R , typename O >  
upcxx::detail::rput_states< rput_byref, S, R, O >::rput_states (   
    intrank_t target,  
    completions< S, R, O > && cxs ) [inline]
```

Definition at line 79 of file rput.hpp.

6.231.3 Member Data Documentation

6.231.3.1 o

```
template<typename S , typename R , typename O >  
rput_state_here<O> upcxx::detail::rput_states< rput_byref, S, R, O >::o
```

Definition at line 77 of file rput.hpp.

6.231.3.2 r

```
template<typename S , typename R , typename O >  
rput_state_remote<R> upcxx::detail::rput_states< rput_byref, S, R, O >::r
```

Definition at line 76 of file rput.hpp.

6.231.3.3 s

```
template<typename S , typename R , typename O >  
rput_state_here<S> upcxx::detail::rput_states< rput_byref, S, R, O >::s
```

Definition at line 75 of file rput.hpp.

6.231.3.4 target

```
template<typename S , typename R , typename O >
inrank_t upcxx::detail::rput_states< rput_byref, S, R, O >::target
```

Definition at line 74 of file rput.hpp.

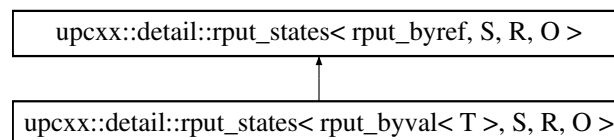
The documentation for this struct was generated from the following file:

- [rput.hpp](#)

6.232 upcxx::detail::rput_states< rput_byval< T >, S, R, O > Struct Template Reference

```
#include <rput.hpp>
```

Inheritance diagram for upcxx::detail::rput_states< rput_byval< T >, S, R, O >:



Public Member Functions

- [rput_states](#) (T &&value, [inrank_t](#) target, [completions](#)< S, R, O > &&cxs)

Public Attributes

- [T](#) [buffer](#)

6.232.1 Detailed Description

```
template<typename T, typename S, typename R, typename O>
struct upcxx::detail::rput_states< rput_byval< T >, S, R, O >
```

Definition at line 88 of file rput.hpp.

6.232.2 Constructor & Destructor Documentation

6.232.2.1 rput_states()

```
template<typename T , typename S , typename R , typename O >
upcxx::detail::rput_states< rput_byval< T >, S, R, O >::rput_states (
    T && value,
    intrank_t target,
    completions< S, R, O > && cxs ) [inline]
```

Definition at line 91 of file rput.hpp.

6.232.3 Member Data Documentation

6.232.3.1 buffer

```
template<typename T , typename S , typename R , typename O >
T upcxx::detail::rput_states< rput_byval< T >, S, R, O >::buffer
```

Definition at line 89 of file rput.hpp.

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

- [rput.hpp](#)

6.233 upcxx::trait_all< Tr > Struct Template Reference

```
#include <utility.hpp>
```

6.233.1 Detailed Description

```
template<template< typename... > class ... Tr>
struct upcxx::trait_all< Tr >
```

Definition at line 157 of file utility.hpp.

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

- [utility.hpp](#)

6.234 upcxx::trait_all< Tr0, Trs... > Struct Template Reference

```
#include <utility.hpp>
```


Classes

- [struct `type`](#)

6.234.1 Detailed Description

```
template<template< typename... > class Tr0, template< typename... > class ... Trs>
struct upcxx::trait_all< Tr0, Trs... >
```

Definition at line 167 of file `utility.hpp`.

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

- [utility.hpp](#)

6.235 `upcxx::trait_all<>` Struct Template Reference

```
#include <utility.hpp>
```

Public Types

- `template<typename T >`
`using type = std::true_type`

6.235.1 Detailed Description

```
template<>
struct upcxx::trait_all<>
```

Definition at line 160 of file `utility.hpp`.

6.235.2 Member Typedef Documentation

6.235.2.1 `type`

```
template<typename T >
using upcxx::trait\_all<>::type = std::true_type
```

Definition at line 162 of file `utility.hpp`.

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

- [utility.hpp](#)

6.236 `upcxx::trait_any< Tr >` Struct Template Reference

```
#include <utility.hpp>
```

6.236.1 Detailed Description

```
template<template< typename... > class ... Tr>  
struct upcxx::trait_any< Tr >
```

Definition at line 136 of file `utility.hpp`.

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

- [utility.hpp](#)

6.237 `upcxx::trait_any< Tr0, Trs... >` Struct Template Reference

```
#include <utility.hpp>
```

Classes

- struct [type](#)

6.237.1 Detailed Description

```
template<template< typename... > class Tr0, template< typename... > class ... Trs>  
struct upcxx::trait_any< Tr0, Trs... >
```

Definition at line 146 of file `utility.hpp`.

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

- [utility.hpp](#)

6.238 `upcxx::trait_any<>` Struct Template Reference

```
#include <utility.hpp>
```

Public Types

- `template<typename T >`
using [type](#) = `std::false_type`

6.238.1 Detailed Description

```
template<>
struct upcxx::trait_any<>
```

Definition at line 139 of file `utility.hpp`.

6.238.2 Member Typedef Documentation

6.238.2.1 `type`

```
template<typename T >
using upcxx::trait_any<>::type = std::false_type
```

Definition at line 141 of file `utility.hpp`.

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

- [utility.hpp](#)

6.239 `upcxx::trait_forall< Test, T >` Struct Template Reference

```
#include <utility.hpp>
```

6.239.1 Detailed Description

```
template<template< typename... > class Test, typename ... T>
struct upcxx::trait_forall< Test, T >
```

Definition at line 115 of file `utility.hpp`.

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

- [utility.hpp](#)

6.240 `upcxx::trait_forall< Test >` Struct Template Reference

```
#include <utility.hpp>
```

Static Public Attributes

- static constexpr bool `value` = true

6.240.1 Detailed Description

```
template<template< typename... > class Test>  
struct upcxx::trait_forall< Test >
```

Definition at line 117 of file utility.hpp.

6.240.2 Member Data Documentation

6.240.2.1 value

```
template<template< typename... > class Test>  
constexpr bool upcxx::trait_forall< Test >::value = true [static]
```

Definition at line 118 of file utility.hpp.

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

- [utility.hpp](#)

6.241 upcxx::trait_forall< Test, T, Ts... > Struct Template Reference

```
#include <utility.hpp>
```

Static Public Attributes

- static constexpr bool [value](#) = Test<T>::value && [trait_forall](#)<Test,Ts...>::value

6.241.1 Detailed Description

```
template<template< typename... > class Test, typename T, typename ... Ts>  
struct upcxx::trait_forall< Test, T, Ts... >
```

Definition at line 121 of file utility.hpp.

6.241.2 Member Data Documentation

6.241.2.1 value

```
template<template< typename... > class Test, typename T , typename ... Ts>
constexpr bool upcxx::trait_forall< Test, T, Ts... >::value = Test<T>::value && trait_forall<Test,Ts...>::value [static]
```

Definition at line 122 of file utility.hpp.

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

- [utility.hpp](#)

6.242 `upcxx::trait_forall_tupled< Test, Tuple >` Struct Template Reference

```
#include <utility.hpp>
```

6.242.1 Detailed Description

```
template<template< typename... > class Test, typename Tuple>
struct upcxx::trait_forall_tupled< Test, Tuple >
```

Definition at line 126 of file utility.hpp.

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

- [utility.hpp](#)

6.243 `upcxx::trait_forall_tupled< Test, std::tuple< T... > >` Struct Template Reference

```
#include <utility.hpp>
```

Static Public Attributes

- static constexpr bool `value` = `trait_forall<Test, T...>::value`

6.243.1 Detailed Description

```
template<template< typename... > class Test, typename ... T>
struct upcxx::trait_forall_tupled< Test, std::tuple< T... > >
```

Definition at line 128 of file utility.hpp.

6.243.2 Member Data Documentation

6.243.2.1 value

```
template<template< typename... > class Test, typename ... T>
constexpr bool upcxx::trait_forall_tupled< Test, std::tuple< T... > >::value = trait_forall<Test, T...>::value [static]
```

Definition at line 129 of file utility.hpp.

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

- [utility.hpp](#)

6.244 upcxx::detail::tuple_get_or_void< i, TupRef, in_range > Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- auto [operator\(\)](#) (TupRef t) -> decltype(std::get< i >(t))

6.244.1 Detailed Description

```
template<int i, typename TupRef, bool in_range = ( 0 <= i && i < std::tuple_size<typename std::decay<TupRef>::type>::value )>
struct upcxx::detail::tuple_get_or_void< i, TupRef, in_range >
```

Definition at line 237 of file utility.hpp.

6.244.2 Member Function Documentation

6.244.2.1 operator()

```
template<int i, typename TupRef , bool in_range = ( 0 <= i && i < std::tuple_size<typename
std::decay<TupRef>::type>::value )>
auto upcxx::detail::tuple_get_or_void< i, TupRef, in_range >::operator() (
    TupRef t ) -> decltype(std::get<i>(t)) [inline]
```

Definition at line 238 of file utility.hpp.

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

- [utility.hpp](#)

6.245 `upcxx::detail::tuple_get_or_void< i, TupRef, false >` Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- void `operator()` (TupRef t)

6.245.1 Detailed Description

```
template<int i, typename TupRef>  
struct upcxx::detail::tuple_get_or_void< i, TupRef, false >
```

Definition at line 245 of file `utility.hpp`.

6.245.2 Member Function Documentation

6.245.2.1 `operator()`

```
template<int i, typename TupRef >  
void upcxx::detail::tuple_get_or_void< i, TupRef, false >::operator() (  
    TupRef t ) [inline]
```

Definition at line 246 of file `utility.hpp`.

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

- [utility.hpp](#)

6.246 `upcxx::detail::tuple_rvals_get< i, Tup >` Struct Template Reference

```
#include <utility.hpp>
```

6.246.1 Detailed Description

```
template<int i, typename Tup>  
struct upcxx::detail::tuple_rvals_get< i, Tup >
```

Definition at line 336 of file `utility.hpp`.

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

- [utility.hpp](#)

6.247 `upcxx::detail::tuple_rvals_get< Tup &&, i, Ti & >` Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- `Ti & operator()` (`Tup &&tup`) const

6.247.1 Detailed Description

```
template<int i, typename Tup>  
struct upcxx::detail::tuple_rvals_get< Tup &&, i, Ti & >
```

Definition at line 380 of file `utility.hpp`.

6.247.2 Member Function Documentation

6.247.2.1 `operator()`

```
template<int i, typename Tup >  
Ti& upcxx::detail::tuple_rvals_get< Tup &&, i, Ti & >::operator() (  
    Tup && tup ) const [inline]
```

Definition at line 381 of file `utility.hpp`.

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

- [utility.hpp](#)

6.248 `upcxx::detail::tuple_rvals_get< Tup &&, i, Ti && >` Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- `Ti && operator()` (`Tup &&tup`) const

6.248.1 Detailed Description

```
template<int i, typename Tup>  
struct upcxx::detail::tuple_rvals_get< Tup &&, i, Ti && >
```

Definition at line 386 of file `utility.hpp`.

6.248.2 Member Function Documentation

6.248.2.1 `operator()`

```
template<int i, typename Tup >
Ti&& upcxx::detail::tuple_rvals_get< Tup &&, i, Ti && >::operator() (
    Tup && tup ) const [inline]
```

Definition at line 387 of file `utility.hpp`.

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

- [utility.hpp](#)

6.249 `upcxx::detail::tuple_rvals_get< Tup &&, i, Ti >` Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- `Ti operator()` (`Tup &&tup`) `const`

6.249.1 Detailed Description

```
template<int i, typename Tup>
struct upcxx::detail::tuple_rvals_get< Tup &&, i, Ti >
```

Definition at line 392 of file `utility.hpp`.

6.249.2 Member Function Documentation

6.249.2.1 `operator()`

```
template<int i, typename Tup >
Ti upcxx::detail::tuple_rvals_get< Tup &&, i, Ti >::operator() (
    Tup && tup ) const [inline]
```

Definition at line 393 of file `utility.hpp`.

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

- [utility.hpp](#)

6.250 `upcxx::detail::tuple_rvals_get< Tup &, i, Ti & >` Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- `Ti & operator()` (`Tup &tup`) const

6.250.1 Detailed Description

```
template<int i, typename Tup>  
struct upcxx::detail::tuple_rvals_get< Tup &, i, Ti & >
```

Definition at line 340 of file `utility.hpp`.

6.250.2 Member Function Documentation

6.250.2.1 `operator()`

```
template<int i, typename Tup >  
Ti& upcxx::detail::tuple_rvals_get< Tup &, i, Ti & >::operator() (  
    Tup & tup ) const [inline]
```

Definition at line 341 of file `utility.hpp`.

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

- [utility.hpp](#)

6.251 `upcxx::detail::tuple_rvals_get< Tup &, i, Ti && >` Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- `Ti && operator()` (`Tup &tup`) const

6.251.1 Detailed Description

```
template<int i, typename Tup>  
struct upcxx::detail::tuple_rvals_get< Tup &, i, Ti && >
```

Definition at line 346 of file `utility.hpp`.

6.251.2 Member Function Documentation

6.251.2.1 `operator()`

```
template<int i, typename Tup >
Ti&& upcxx::detail::tuple_rvals_get< Tup &, i, Ti && >::operator() (
    Tup & tup ) const [inline]
```

Definition at line 347 of file `utility.hpp`.

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

- [utility.hpp](#)

6.252 `upcxx::detail::tuple_rvals_get< Tup &, i, Ti >` Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- `Ti && operator()` (`Tup &tup`) const

6.252.1 Detailed Description

```
template<int i, typename Tup>
struct upcxx::detail::tuple_rvals_get< Tup &, i, Ti >
```

Definition at line 352 of file `utility.hpp`.

6.252.2 Member Function Documentation

6.252.2.1 `operator()`

```
template<int i, typename Tup >
Ti&& upcxx::detail::tuple_rvals_get< Tup &, i, Ti >::operator() (
    Tup & tup ) const [inline]
```

Definition at line 353 of file `utility.hpp`.

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

- [utility.hpp](#)

6.253 `upcxx::detail::tuple_rvals_get< Tup const &, i, Ti & >` Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- `Ti & operator()` (`Tup const &tup`) const

6.253.1 Detailed Description

```
template<int i, typename Tup>
struct upcxx::detail::tuple_rvals_get< Tup const &, i, Ti & >
```

Definition at line 360 of file `utility.hpp`.

6.253.2 Member Function Documentation

6.253.2.1 `operator()`

```
template<int i, typename Tup >
Ti& upcxx::detail::tuple_rvals_get< Tup const &, i, Ti & >::operator() (
    Tup const & tup ) const [inline]
```

Definition at line 361 of file `utility.hpp`.

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

- [utility.hpp](#)

6.254 `upcxx::detail::tuple_rvals_get< Tup const &, i, Ti && >` Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- `Ti && operator()` (`Tup const &tup`) const

6.254.1 Detailed Description

```
template<int i, typename Tup>
struct upcxx::detail::tuple_rvals_get< Tup const &, i, Ti && >
```

Definition at line 366 of file `utility.hpp`.

6.254.2 Member Function Documentation

6.254.2.1 `operator()`

```
template<int i, typename Tup >
Ti&& upcxx::detail::tuple_rvals_get< Tup const &, i, Ti && >::operator() (
    Tup const & tup ) const [inline]
```

Definition at line 367 of file `utility.hpp`.

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

- [utility.hpp](#)

6.255 `upcxx::detail::tuple_rvals_get< Tup const &, i, Ti >` Struct Template Reference

```
#include <utility.hpp>
```

Public Member Functions

- `Ti const &` [operator\(\)](#) (`Tup const &tup`) `const`

6.255.1 Detailed Description

```
template<int i, typename Tup>
struct upcxx::detail::tuple_rvals_get< Tup const &, i, Ti >
```

Definition at line 372 of file `utility.hpp`.

6.255.2 Member Function Documentation

6.255.2.1 `operator()`

```
template<int i, typename Tup >
Ti const& upcxx::detail::tuple_rvals_get< Tup const &, i, Ti >::operator() (
    Tup const & tup ) const [inline]
```

Definition at line 373 of file `utility.hpp`.

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

- [utility.hpp](#)

6.256 `upcxx::tuple_types_into< Tuple, Into >` Struct Template Reference

```
#include <utility.hpp>
```

6.256.1 Detailed Description

```
template<typename Tuple, template< typename... > class Into>  
struct upcxx::tuple_types_into< Tuple, Into >
```

Definition at line 177 of file `utility.hpp`.

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

- [utility.hpp](#)

6.257 `upcxx::tuple_types_into< std::tuple< T... >, Into >` Struct Template Reference

```
#include <utility.hpp>
```

Public Types

- `typedef Into< T... >` [type](#)

6.257.1 Detailed Description

```
template<typename ... T, template< typename... > class Into>  
struct upcxx::tuple_types_into< std::tuple< T... >, Into >
```

Definition at line 179 of file `utility.hpp`.

6.257.2 Member Typedef Documentation

6.257.2.1 `type`

```
template<typename ... T, template< typename... > class Into>  
typedef Into<T...> upcxx::tuple\_types\_into< std::tuple< T... >, Into >::type
```

Definition at line 180 of file `utility.hpp`.

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

- [utility.hpp](#)

6.258 `upcxx::trait_all< Tr0, Trs... >::type< T >` Struct Template Reference

```
#include <utility.hpp>
```

Static Public Attributes

- static constexpr bool `value` = `Tr0<T>::value` && `trait_all<Trs...>::template type<T>::value`

6.258.1 Detailed Description

```
template<template< typename... > class Tr0, template< typename... > class ... Trs>  
template<typename T>  
struct upcxx::trait_all< Tr0, Trs... >::type< T >
```

Definition at line 169 of file `utility.hpp`.

6.258.2 Member Data Documentation

6.258.2.1 `value`

```
template<template< typename... > class Tr0, template< typename... > class ... Trs>  
template<typename T>  
constexpr bool upcxx::trait_all< Tr0, Trs... >::type< T >::value = Tr0<T>::value && trait_↵  
_all<Trs...>::template type<T>::value [static]
```

Definition at line 170 of file `utility.hpp`.

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

- [utility.hpp](#)

6.259 `upcxx::trait_any< Tr0, Trs... >::type< T >` Struct Template Reference

```
#include <utility.hpp>
```

Static Public Attributes

- static constexpr bool `value` = `Tr0<T>::value` || `trait_any<Trs...>::template type<T>::value`

6.259.1 Detailed Description

```
template<template< typename... > class Tr0, template< typename... > class ... Trs>
template<typename T>
struct upcxx::trait_any< Tr0, Trs... >::type< T >
```

Definition at line 148 of file utility.hpp.

6.259.2 Member Data Documentation

6.259.2.1 value

```
template<template< typename... > class Tr0, template< typename... > class ... Trs>
template<typename T >
constexpr bool upcxx::trait_any< Tr0, Trs... >::type< T >::value = Tr0<T>::value || trait_↔
any<Trs...>::template type<T>::value [static]
```

Definition at line 149 of file utility.hpp.

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

- [utility.hpp](#)

Chapter 7

File Documentation

7.1 `allocate.hpp` File Reference

```
#include <upcxx/backend.hpp>
#include <upcxx/diagnostic.hpp>
#include <upcxx/global_ptr.hpp>
#include <algorithm>
#include <cmath>
#include <cstdint>
#include <cstddef>
#include <new>
#include <type_traits>
```

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- `template<typename T, std::size_t alignment = alignof(T)>`
`global_ptr< T > upcxx::allocate (std::size_t n=1)`
- `template<typename T >`
`void upcxx::deallocate (global_ptr< T > gptr)`
- `template<bool throws, typename T, typename ... Args>`
`global_ptr< T > upcxx::detail::new_ (Args &&...args)`
- `template<typename T, typename ... Args>`
`global_ptr< T > upcxx::new_ (Args &&...args)`
- `template<typename T, typename ... Args>`
`global_ptr< T > upcxx::new_ (const std::nothrow_t &tag, Args &&...args)`
- `template<bool throws, typename T >`
`global_ptr< T > upcxx::detail::new_array (std::size_t n)`
- `template<typename T >`
`global_ptr< T > upcxx::new_array (std::size_t n)`
- `template<typename T >`
`global_ptr< T > upcxx::new_array (std::size_t n, const std::nothrow_t &tag)`
- `template<typename T >`
`void upcxx::delete_ (global_ptr< T > gptr)`
- `template<typename T >`
`void upcxx::delete_array (global_ptr< T > gptr)`

7.2 allreduce.hpp File Reference

```
#include <upcxx/backend.hpp>
#include <upcxx/dist_object.hpp>
#include <upcxx/rpc.hpp>
```

Classes

- struct [upcxx::detail::allreduce_state](#)< T, Op >

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- template<typename T1 , typename BinaryOp , typename T = typename std::decay<T1>::type>
future< T > [upcxx::allreduce](#) (T1 &&value, BinaryOp op)

7.3 atomic.hpp File Reference

```
#include <atomic>
#include <upcxx/backend.hpp>
#include <upcxx/global_ptr.hpp>
#include <upcxx/rpc.hpp>
```

Namespaces

- [upcxx](#)

Functions

- template<typename T >
future< T > [upcxx::atomic_get](#) (global_ptr< T > p, std::memory_order order)
- template<typename T >
future [upcxx::atomic_put](#) (global_ptr< T > p, T val, std::memory_order order)
- template<typename T >
future< T > [upcxx::atomic_fetch_add](#) (global_ptr< T > p, T val, std::memory_order order)

7.4 backend/gasnet1_seq/backend.cpp File Reference

```
#include <upcxx/os_env.hpp>
#include <upcxx/backend/gasnet1_seq/backend.hpp>
#include <cstring>
#include <gasnet.h>
#include <unistd.h>
#include <upcxx/dl_malloc.h>
```

Enumerations

- enum

7.4.1 Enumeration Type Documentation

7.4.1.1 anonymous enum

anonymous enum

Definition at line 45 of file backend.cpp.

7.5 backend/gasnet1_seq/backend.hpp File Reference

```
#include <upcxx/backend.hpp>
#include <upcxx/command.hpp>
#include <cstdint>
#include <cstdlib>
```

Classes

- struct [upcxx::backend::gasnet1_seq::action](#)
- struct [upcxx::backend::gasnet1_seq::rma_cb](#)
- struct [upcxx::backend::gasnet1_seq::action_impl< Fn >](#)
- struct [upcxx::backend::rma_put_cb](#)
- struct [upcxx::backend::rma_get_cb](#)
- struct [upcxx::backend::rma_put_cb_wstate< State >](#)
- struct [upcxx::backend::rma_get_cb_wstate< State >](#)
- struct [upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx >](#)
- struct [upcxx::backend::gasnet1_seq::rma_get_cb_impl< State, OpCx >](#)

Namespaces

- [upcxx](#)
- [upcxx::backend](#)
- [upcxx::backend::gasnet1_seq](#)

Functions

- void [upcxx::backend::gasnet1_seq::send_am_eager_restricted](#) (inrank_t recipient, void *command_buf, std::size_t buf_size, std::size_t buf_align)
- template<typename Fn >
void [upcxx::backend::gasnet1_seq::send_am_restricted](#) (inrank_t recipient, Fn &&fn)
- void [upcxx::backend::gasnet1_seq::send_am_eager_queued](#) (progress_level level, inrank_t recipient, void *command_buf, std::size_t buf_size, std::size_t buf_align)
- void [upcxx::backend::gasnet1_seq::send_am_rdzv](#) (progress_level level, inrank_t recipient, void *command_buf, std::size_t buf_size, std::size_t buf_align)
- template<typename Fn1 >
void [upcxx::backend::during_user](#) (Fn1 &&fn)
- void [upcxx::backend::during_user](#) (promise<> &&pro)
- void [upcxx::backend::during_user](#) (promise<> *pro)
- template<typename Fn1 >
void [upcxx::backend::during_level](#) (progress_level level, Fn1 &&fn)
- template<upcxx::progress_level level, typename Fn >
void [upcxx::backend::send_am](#) (inrank_t recipient, Fn &&fn)
- template<typename State , typename SrcCx , typename OpCx >
rma_put_cb_wstate< State > * [upcxx::backend::make_rma_put_cb](#) (State state, SrcCx src_cx, OpCx op←_cx)
- template<typename State , typename OpCx >
rma_get_cb_wstate< State > * [upcxx::backend::make_rma_get_cb](#) (State state, OpCx op_cx)

Variables

- std::size_t [upcxx::backend::gasnet1_seq::am_size_rdzv_cutover](#)
- bool [upcxx::backend::gasnet1_seq::in_user_progress_](#) = false
- action * [upcxx::backend::gasnet1_seq::user_actions_head_](#) = nullptr
- action ** [upcxx::backend::gasnet1_seq::user_actions_tailp_](#) = &gasnet1_seq::user_actions_head_

7.6 backend.hpp File Reference

```
#include <upcxx/future.hpp>
#include <upcxx/packing.hpp>
#include <cstdint>
#include <cstddef>
```

Classes

- struct [upcxx::backend::rma_put_cb_wstate< State >](#)
- struct [upcxx::backend::rma_get_cb_wstate< State >](#)

Namespaces

- [upcxx](#)
- [upcxx::backend](#)

Macros

- `#define gasnet1_seq 100`

Typedefs

- typedef int [upcxx::inrank_t](#)
- typedef unsigned int [upcxx::uinrank_t](#)

Enumerations

- enum [upcxx::progress_level](#) { [upcxx::progress_level::internal](#), [upcxx::progress_level::user](#) }

Functions

- void [upcxx::init](#) ()
- void [upcxx::finalize](#) ()
- [inrank_t](#) [upcxx::rank_n](#) ()
- [inrank_t](#) [upcxx::rank_me](#) ()
- void * [upcxx::allocate](#) (std::size_t size, std::size_t alignment=alignof(std::max_align_t))
- void [upcxx::deallocate](#) (void *p)
- void [upcxx::progress](#) ([progress_level](#) lev=[progress_level::user](#))
- void [upcxx::barrier](#) ()
- template<typename Fn >
void [upcxx::backend::during_user](#) (Fn &&fn)
- void [upcxx::backend::during_user](#) (promise<> &&pro)
- void [upcxx::backend::during_user](#) (promise<> *pro)
- template<typename Fn >
void [upcxx::backend::during_level](#) ([progress_level](#) level, Fn &&fn)
- template<[upcxx::progress_level](#) level, typename Fn >
void [upcxx::backend::send_am](#) ([inrank_t](#) recipient, Fn &&fn)
- template<typename State , typename SrcCx , typename OpCx >
[rma_put_cb_wstate](#)< State > * [upcxx::backend::make_rma_put_cb](#) (State state, SrcCx src_cx, OpCx op←
_cx)
- void [upcxx::backend::rma_put](#) ([inrank_t](#) rank_d, void *buf_d, void *buf_s, std::size_t buf_size, [rma_put_cb](#) *cb)
- template<typename State , typename OpCx >
[rma_get_cb_wstate](#)< State > * [upcxx::backend::make_rma_get_cb](#) (State state, OpCx op_cx)
- void [upcxx::backend::rma_get](#) (void *buf_d, [inrank_t](#) rank_s, void *buf_s, std::size_t buf_size, [rma_get_cb](#) *cb)

Variables

- [inrank_t](#) [upcxx::backend::rank_n](#)
- [inrank_t](#) [upcxx::backend::rank_me](#)

7.6.1 Macro Definition Documentation

7.6.1.1 gasnet1_seq

```
#define gasnet1_seq 100
```

Definition at line 110 of file backend.hpp.

7.7 bind.hpp File Reference

```
#include <upcxx/future.hpp>
#include <upcxx/packing.hpp>
#include <upcxx/utility.hpp>
#include <tuple>
#include <type_traits>
#include <utility>
```

Classes

- [struct upcxx::binding< T >](#)
- [struct upcxx::binding_is_trivial< T, trivial >](#)
- [struct upcxx::binding_is_trivial< T, std::integral_constant< bool, binding< T >::is_trivial > >](#)
- [struct upcxx::binding< T & >](#)
- [struct upcxx::binding< T && >](#)
- [struct upcxx::binding< T const >](#)
- [struct upcxx::binding< T volatile >](#)
- [struct upcxx::detail::bound_function_base< Fn, BndTup, all_trivial >](#)
- [struct upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true >](#)
- [struct upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >](#)
- [struct upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator< Arg >](#)
- [struct upcxx::bound_function< Fn, B >](#)
- [struct upcxx::packing< bound_function< Fn, B... > >](#)
- [struct upcxx::detail::bind< FnDecayed >](#)
- [struct upcxx::detail::bind< bound_function< Fn0, B0... > >](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- template<typename Fn , typename ... B>
auto [upcxx::bind](#) (Fn &&fn, B &&...b) -> decltype(detail::bind< typename std::decay< Fn >::type >()(std::forward< Fn >(fn), std::forward< B >(b)...))
- template<typename Fn >
Fn && [upcxx::bind](#) (Fn &&fn)
- template<typename ... P, int ... heads, int tail>
auto [upcxx::detail::bind_last](#) (std::tuple< P... > parms, [upcxx::index_sequence](#)< heads... >, std::integral_constant< int, tail >) -> decltype(detail::bind< typename std::decay< typename std::tuple_element< tail, std::tuple< P... >>::type >::type >()(std::move(std::get< tail >(parms)), std::move(std::get< heads >(parms))...))
- template<typename ... P>
auto [upcxx::bind_last](#) (P &&...parm) -> decltype(detail::bind_last(std::tuple< P &&... >
- template<typename Fn >
Fn && [upcxx::bind_last](#) (Fn &&fn)

7.8 broadcast.hpp File Reference

```
#include <upcxx/backend.hpp>
#include <upcxx/dist_object.hpp>
#include <upcxx/rpc.hpp>
```

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- template<typename T >
void [upcxx::detail::broadcast_receive](#) (T const &value, intrank_t rank_ub, dist_id< promise< T >> id)
- template<typename T1 , typename T = typename std::decay<T1>::type>
future< T > [upcxx::broadcast](#) (T1 &&value, intrank_t root)

7.9 command.hpp File Reference

```
#include <upcxx/diagnostic.hpp>
#include <upcxx/future.hpp>
#include <upcxx/packing.hpp>
```

Classes

- struct [upcxx::commanding](#)< Fn >

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- `template<typename Fn >`
void [upcxx::command_size_ubound](#) (parcel_layout &ub, Fn &&fn)
- `template<typename Fn >`
void [upcxx::command_pack](#) (parcel_writer &w, std::size_t size_ub, Fn &&fn)
- future [upcxx::command_execute](#) (parcel_reader &r)
- `template<typename Fn >`
future [upcxx::detail::command_executor](#) (parcel_reader &r)
- `template<typename Fn1 >`
void [upcxx::command_size_ubound](#) (parcel_layout &ub, Fn1 &&fn)
- `template<typename Fn1 >`
void [upcxx::command_pack](#) (parcel_writer &w, std::size_t size_ub, Fn1 &&fn)

7.10 completion.hpp File Reference

```
#include <upcxx/bind.hpp>
#include <upcxx/future.hpp>
```

Classes

- struct [upcxx::nil_cx](#)
- struct [upcxx::rpc_cx< Fn >](#)
- struct [upcxx::future_cx< ordinal >](#)
- struct [upcxx::promise_cx< T >](#)
- struct [upcxx::detail::disjoin_cx< A, B, B_ord_bump >](#)
- struct [upcxx::detail::disjoin_cx< A, nil_cx, B_ord_bump >](#)
- struct [upcxx::detail::disjoin_cx< nil_cx, B, B_ord_bump >](#)
- struct [upcxx::detail::disjoin_cx< nil_cx, future_cx< B_ord_old >, B_ord_bump >](#)
- struct [upcxx::detail::disjoin_cx< nil_cx, nil_cx, B_ord0 >](#)
- struct [upcxx::detail::is_future_cx< Cx >](#)
- struct [upcxx::detail::is_future_cx< future_cx< ordinal > >](#)
- struct [upcxx::completions< SourceCx, RemoteCx, OpxnCx >](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- `template<typename AS , typename AR , typename AO , typename BS , typename BR , typename BO > completions< typename detail::disjoin_cx< AS, BS, completions< AS, AR, AO >::future_n >::type, typename detail::disjoin_cx< AR, BR, completions< AS, AR, AO >::future_n >::type, typename detail::disjoin_cx< AO, BO, completions< AS, AR, AO >::future_n >::type > upcxx::operator| (completions< AS, AR, AO > a, completions< BS, BR, BO > b)`
- `template<typename ... T> completions< promise_cx< T... >, nil_cx, nil_cx > upcxx::source_cx_as_promise (promise< T... > &pro)`
- `template<typename ... T> completions< nil_cx, nil_cx, promise_cx< T... > > upcxx::operxn_cx_as_promise (promise< T... > &pro)`
- `template<typename Fn , typename ... Args> auto upcxx::remote_cx_as_rpc (Fn &&fn, Args &&...args) -> completions< nil_cx, rpc_cx< decltype(upcxx::bind(std::forward< Fn >(fn), std::forward< Args >(args)...))>, nil_cx >`

Variables

- `constexpr completions< future_cx< 0 >, nil_cx, nil_cx > upcxx::source_cx_as_future = {}`
- `constexpr completions< nil_cx, nil_cx, future_cx< 0 > > upcxx::operxn_cx_as_future = {}`

7.11 diagnostic.cpp File Reference

```
#include <upcxx/diagnostic.hpp>
#include <iostream>
#include <sstream>
#include <sys/types.h>
#include <unistd.h>
#include <signal.h>
```

Namespaces

- [upcxx](#)

Variables

- `bool upcxx::dbgbrk_spin_init = false`
- `bool upcxx::dbgbrk_spin = true`

7.12 diagnostic.hpp File Reference

```
#include <sstream>
```

Namespaces

- [upcxx](#)

Macros

- `#define UPCXX_ASSERT_1(ok)`
- `#define UPCXX_ASSERT_2(ok, ios_msg)`
- `#define UPCXX_ASSERT_DISPATCH(_1, _2, NAME, ...) NAME`
- `#define UPCXX_ASSERT(...) UPCXX_ASSERT_DISPATCH(__VA_ARGS__, UPCXX_ASSERT_2, UPCXX_ASSERT_1)(__VA_ARGS__)`
- `#define UPCXX_ASSERT_ALWAYS(...) UPCXX_ASSERT_DISPATCH(__VA_ARGS__, UPCXX_ASSERT_2, UPCXX_ASSERT_1)(__VA_ARGS__)`
- `#define UPCXX_INVOKE_UB() ::upcxx::assert_failed(__FILE__, __LINE__)`

Functions

- `void upcxx::dbgbrk ()`
- `void upcxx::assert_failed (const char *file, int line, const char *msg=nullptr)`

7.12.1 Macro Definition Documentation

7.12.1.1 UPCXX_ASSERT

```
#define UPCXX_ASSERT(
    ... ) UPCXX_ASSERT_DISPATCH(__VA_ARGS__, UPCXX_ASSERT_2, UPCXX_ASSERT_1) (__VA_ARGS__)
```

Definition at line 30 of file diagnostic.hpp.

7.12.1.2 UPCXX_ASSERT_1

```
#define UPCXX_ASSERT_1(
    ok )
```

Value:

```
do { \
    if (!(ok)) \
        ::upcxx::assert_failed(__FILE__, __LINE__); \
} while (0);
```

Definition at line 11 of file diagnostic.hpp.

7.12.1.3 UPCXX_ASSERT_2

```
#define UPCXX_ASSERT_2(
    ok,
    ios_msg )
```

Value:

```
do { \
    if(!(ok)) { \
        ::std::stringstream ss; \
        ss << ios_msg; \
        ::upcxx::assert_failed(__FILE__, __LINE__, ss.str().c_str()); \
    } \
} while(0);
```

Definition at line 17 of file diagnostic.hpp.

7.12.1.4 UPCXX_ASSERT_ALWAYS

```
#define UPCXX_ASSERT_ALWAYS(
    ... ) UPCXX_ASSERT_DISPATCH(__VA_ARGS__, UPCXX_ASSERT_2, UPCXX_ASSERT_1) (__VA_ARGS__,
ARGS__)
```

Definition at line 33 of file diagnostic.hpp.

7.12.1.5 UPCXX_ASSERT_DISPATCH

```
#define UPCXX_ASSERT_DISPATCH(
    _1,
    _2,
    NAME,
    ... ) NAME
```

Definition at line 26 of file diagnostic.hpp.

7.12.1.6 UPCXX_INVOKE_UB

```
#define UPCXX_INVOKE_UB( ) ::upcxx::assert_failed(__FILE__, __LINE__)
```

Definition at line 36 of file diagnostic.hpp.

7.13 digest.cpp File Reference

```
#include <upcxx/digest.hpp>
```

7.14 digest.hpp File Reference

```
#include <cstdint>
#include <functional>
#include <iostream>
```

Classes

- struct [upcxx::digest](#)
- struct [std::hash<upcxx::digest>](#)

Namespaces

- [upcxx](#)
- [std](#)

Functions

- [std::ostream & upcxx::operator<<](#) ([std::ostream &o](#), [digest x](#))

7.15 dist_object.cpp File Reference

```
#include <upcxx/dist_object.hpp>
```

7.16 dist_object.hpp File Reference

```
#include <upcxx/bind.hpp>
#include <upcxx/digest.hpp>
#include <upcxx/future.hpp>
#include <upcxx/rpc.hpp>
#include <upcxx/utility.hpp>
#include <cstdint>
#include <unordered_map>
```

Classes

- struct [upcxx::dist_id<T>](#)
- class [upcxx::dist_object<T>](#)
- struct [upcxx::dist_id<T>](#)
- struct [std::hash<upcxx::dist_id<T>>](#)
- class [upcxx::dist_object<T>](#)
- struct [upcxx::binding<dist_object<T>&>](#)
- struct [upcxx::binding<dist_object<T>&&>](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)
- [std](#)

Macros

- `#define` [COMPARATOR](#)(op)

Functions

- `template<typename T >`
`promise< dist_object< T > & > * upcxx::detail::dist_promise (digest id)`
- `template<typename T >`
`std::ostream & upcxx::operator<< (std::ostream &o, dist_id< T > x)`

Variables

- `std::unordered_map< digest, void * > upcxx::detail::dist_master_promises`
- `std::uint64_t upcxx::detail::dist_master_id_bump = 0`

7.16.1 Macro Definition Documentation

7.16.1.1 COMPARATOR

```
#define COMPARATOR(  
    op )
```

Value:

```
friend bool operator op(dist_id a, dist_id b) {\  
    return a.dig_ op b.dig_; \  
}
```

Definition at line 64 of file `dist_object.hpp`.

7.17 dl_malloc.c File Reference

```
#include <sys/types.h>  
#include <stdio.h>  
#include <errno.h>  
#include <time.h>  
#include <stdlib.h>  
#include <string.h>  
#include <sys/mman.h>  
#include <fcntl.h>  
#include <unistd.h>  
#include <sched.h>  
#include <pthread.h>  
#include <sys/param.h>
```

Classes

- struct [mallinfo](#)
- struct [malloc_chunk](#)
- struct [malloc_tree_chunk](#)
- struct [malloc_segment](#)
- struct [malloc_state](#)
- struct [malloc_params](#)

Macros

- #define [ONLY_MSPACES](#) 1
- #define [DLMALLOC_VERSION](#) 20806
- #define [DLMALLOC_EXPORT](#) extern
- #define [MAX_SIZE_T](#) (~(size_t)0)
- #define [USE_LOCKS](#)
- #define [USE_SPIN_LOCKS](#) 0
- #define [MSPACES](#) 1
- #define [MALLOC_ALIGNMENT](#) ((size_t)(2 * sizeof(void *)))
- #define [FOOTERS](#) 0
- #define [ABORT](#) abort()
- #define [ABORT_ON_ASSERT_FAILURE](#) 1
- #define [PROCEED_ON_ERROR](#) 0
- #define [INSECURE](#) 0
- #define [MALLOC_INSPECT_ALL](#) 0
- #define [HAVE_MMAP](#) 1
- #define [MMAP_CLEARS](#) 1
- #define [HAVE_MREMAP](#) 0
- #define [MALLOC_FAILURE_ACTION](#) errno = ENOMEM;
- #define [HAVE_MORECORE](#) 0
- #define [MORECORE_CONTIGUOUS](#) 0
- #define [DEFAULT_GRANULARITY](#) ((size_t)64U * (size_t)1024U)
- #define [DEFAULT_TRIM_THRESHOLD](#) ((size_t)2U * (size_t)1024U * (size_t)1024U)
- #define [DEFAULT_MMAP_THRESHOLD](#) ((size_t)256U * (size_t)1024U)
- #define [MAX_RELEASE_CHECK_RATE](#) 4095
- #define [USE_BUILTIN_FFS](#) 0
- #define [USE_DEV_RANDOM](#) 0
- #define [NO_MALLINFO](#) 0
- #define [MALLINFO_FIELD_TYPE](#) size_t
- #define [NO_MALLOC_STATS](#) 0
- #define [NO_SEGMENT_TRAVERSAL](#) 0
- #define [M_TRIM_THRESHOLD](#) (-1)
- #define [M_GRANULARITY](#) (-2)
- #define [M_MMAP_THRESHOLD](#) (-3)
- #define [_STRUCT_MALLINFO](#)
- #define [STRUCT_MALLINFO_DECLARED](#) 1
- #define [NOINLINE](#)
- #define [FORCEINLINE](#)
- #define [assert](#)(x)
- #define [DEBUG](#) 0
- #define [LOCK_AT_FORK](#) 0
- #define [malloc_getpagesize](#) ((size_t)4096U)
- #define [SIZE_T_SIZE](#) (sizeof(size_t))
- #define [SIZE_T_BITSIZE](#) (sizeof(size_t) << 3)

- #define `SIZE_T_ZERO` ((size_t)0)
- #define `SIZE_T_ONE` ((size_t)1)
- #define `SIZE_T_TWO` ((size_t)2)
- #define `SIZE_T_FOUR` ((size_t)4)
- #define `TWO_SIZE_T_SIZES` (`SIZE_T_SIZE`<<1)
- #define `FOUR_SIZE_T_SIZES` (`SIZE_T_SIZE`<<2)
- #define `SIX_SIZE_T_SIZES` (`FOUR_SIZE_T_SIZES`+`TWO_SIZE_T_SIZES`)
- #define `HALF_MAX_SIZE_T` (`MAX_SIZE_T` / 2U)
- #define `CHUNK_ALIGN_MASK` (`MALLOC_ALIGNMENT` - `SIZE_T_ONE`)
- #define `is_aligned(A)` (((size_t)(A)) & (`CHUNK_ALIGN_MASK`)) == 0)
- #define `align_offset(A)`
- #define `MFAIL` ((void*)(`MAX_SIZE_T`))
- #define `CMFAIL` ((char*)(`MFAIL`)) /* defined for convenience */
- #define `MUNMAP_DEFAULT(a, s)` munmap((a), (s))
- #define `MMAP_PROT` (`PROT_READ`|`PROT_WRITE`)
- #define `MMAP_FLAGS` (`MAP_PRIVATE`)
- #define `MMAP_DEFAULT(s)`
- #define `DIRECT_MMAP_DEFAULT(s)` `MMAP_DEFAULT(s)`
- #define `CALL_MORECORE(S)` `MFAIL`
- #define `USE_MMAP_BIT` (`SIZE_T_ONE`)
- #define `CALL_MMAP(s)` `MMAP_DEFAULT(s)`
- #define `CALL_MUNMAP(a, s)` `MUNMAP_DEFAULT((a), (s))`
- #define `CALL_DIRECT_MMAP(s)` `DIRECT_MMAP_DEFAULT(s)`
- #define `CALL_MREMAP(addr, osz, nsz, mv)` `MFAIL`
- #define `USE_NONCONTIGUOUS_BIT` (4U)
- #define `EXTERN_BIT` (8U)
- #define `MLOCK_T` pthread_mutex_t
- #define `ACQUIRE_LOCK(lk)` pthread_mutex_lock(lk)
- #define `RELEASE_LOCK(lk)` pthread_mutex_unlock(lk)
- #define `TRY_LOCK(lk)` (!pthread_mutex_trylock(lk))
- #define `INITIAL_LOCK(lk)` pthread_init_lock(lk)
- #define `DESTROY_LOCK(lk)` pthread_mutex_destroy(lk)
- #define `USE_LOCK_BIT` (2U)
- #define `ACQUIRE_MALLOC_GLOBAL_LOCK()` `ACQUIRE_LOCK(&malloc_global_mutex);`
- #define `RELEASE_MALLOC_GLOBAL_LOCK()` `RELEASE_LOCK(&malloc_global_mutex);`
- #define `MCHUNK_SIZE` (sizeof(mchunk))
- #define `CHUNK_OVERHEAD` (`SIZE_T_SIZE`)
- #define `MMAP_CHUNK_OVERHEAD` (`TWO_SIZE_T_SIZES`)
- #define `MMAP_FOOT_PAD` (`FOUR_SIZE_T_SIZES`)
- #define `MIN_CHUNK_SIZE` ((`MCHUNK_SIZE` + `CHUNK_ALIGN_MASK`) & ~`CHUNK_ALIGN_MASK`)
- #define `chunk2mem(p)` ((void*)((char*)(p) + `TWO_SIZE_T_SIZES`))
- #define `mem2chunk(mem)` ((mchunkptr)((char*)(mem) - `TWO_SIZE_T_SIZES`))
- #define `align_as_chunk(A)` (mchunkptr)((A) + `align_offset(chunk2mem(A))`)
- #define `MAX_REQUEST` ((-`MIN_CHUNK_SIZE`) << 2)
- #define `MIN_REQUEST` (`MIN_CHUNK_SIZE` - `CHUNK_OVERHEAD` - `SIZE_T_ONE`)
- #define `pad_request(req)` (((req) + `CHUNK_OVERHEAD` + `CHUNK_ALIGN_MASK`) & ~`CHUNK_ALIGN_MASK`)
- #define `request2size(req)` (((req) < `MIN_REQUEST`)? `MIN_CHUNK_SIZE` : `pad_request(req)`)
- #define `PINUSE_BIT` (`SIZE_T_ONE`)
- #define `CINUSE_BIT` (`SIZE_T_TWO`)
- #define `FLAG4_BIT` (`SIZE_T_FOUR`)
- #define `INUSE_BITS` (`PINUSE_BIT`|`CINUSE_BIT`)
- #define `FLAG_BITS` (`PINUSE_BIT`|`CINUSE_BIT`|`FLAG4_BIT`)
- #define `FENCEPOST_HEAD` (`INUSE_BITS`|`SIZE_T_SIZE`)
- #define `cinuse(p)` ((p)->head & `CINUSE_BIT`)

- #define `pinuse(p)` `((p)->head & PINUSE_BIT)`
- #define `flag4inuse(p)` `((p)->head & FLAG4_BIT)`
- #define `is_inuse(p)` `((p)->head & INUSE_BITS) != PINUSE_BIT)`
- #define `is_mmapped(p)` `((p)->head & INUSE_BITS) == 0)`
- #define `chunksize(p)` `((p)->head & ~(FLAG_BITS))`
- #define `clear_pinuse(p)` `((p)->head &= ~PINUSE_BIT)`
- #define `set_flag4(p)` `((p)->head |= FLAG4_BIT)`
- #define `clear_flag4(p)` `((p)->head &= ~FLAG4_BIT)`
- #define `chunk_plus_offset(p, s)` `((mchunkptr)((char*)(p) + (s)))`
- #define `chunk_minus_offset(p, s)` `((mchunkptr)((char*)(p) - (s)))`
- #define `next_chunk(p)` `((mchunkptr)((char*)(p) + ((p)->head & ~FLAG_BITS)))`
- #define `prev_chunk(p)` `((mchunkptr)((char*)(p) - ((p)->prev_foot))`
- #define `next_pinuse(p)` `((next_chunk(p)->head) & PINUSE_BIT)`
- #define `get_foot(p, s)` `((mchunkptr)((char*)(p) + (s))->prev_foot)`
- #define `set_foot(p, s)` `((mchunkptr)((char*)(p) + (s))->prev_foot = (s))`
- #define `set_size_and_pinuse_of_free_chunk(p, s)` `((p)->head = (s|PINUSE_BIT), set_foot(p, s))`
- #define `set_free_with_pinuse(p, s, n)` `(clear_pinuse(n), set_size_and_pinuse_of_free_chunk(p, s))`
- #define `overhead_for(p)` `(is_mmapped(p)? MMAP_CHUNK_OVERHEAD : CHUNK_OVERHEAD)`
- #define `calloc_must_clear(p)` `(!is_mmapped(p))`
- #define `leftmost_child(t)` `((t)->child[0] != 0? (t)->child[0] : (t)->child[1])`
- #define `is_mmapped_segment(S)` `((S)->sflags & USE_MMAP_BIT)`
- #define `is_extern_segment(S)` `((S)->sflags & EXTERN_BIT)`
- #define `NSMALLBINS` (32U)
- #define `NREEBINS` (32U)
- #define `SMALLBIN_SHIFT` (3U)
- #define `SMALLBIN_WIDTH` `(SIZE_T_ONE << SMALLBIN_SHIFT)`
- #define `TREEBIN_SHIFT` (8U)
- #define `MIN_LARGE_SIZE` `(SIZE_T_ONE << TREEBIN_SHIFT)`
- #define `MAX_SMALL_SIZE` `(MIN_LARGE_SIZE - SIZE_T_ONE)`
- #define `MAX_SMALL_REQUEST` `(MAX_SMALL_SIZE - CHUNK_ALIGN_MASK - CHUNK_OVERHEAD)`
- #define `ensure_initialization()` `(void)(mparams.magic != 0 || init_mparams())`
- #define `is_initialized(M)` `((M)->top != 0)`
- #define `use_lock(M)` `((M)->mflags & USE_LOCK_BIT)`
- #define `enable_lock(M)` `((M)->mflags |= USE_LOCK_BIT)`
- #define `disable_lock(M)`
- #define `use_mmap(M)` `((M)->mflags & USE_MMAP_BIT)`
- #define `enable_mmap(M)` `((M)->mflags |= USE_MMAP_BIT)`
- #define `disable_mmap(M)` `((M)->mflags &= ~USE_MMAP_BIT)`
- #define `use_noncontiguous(M)` `((M)->mflags & USE_NONCONTIGUOUS_BIT)`
- #define `disable_contiguous(M)` `((M)->mflags |= USE_NONCONTIGUOUS_BIT)`
- #define `set_lock(M, L)`
- #define `page_align(S)` `((S) + (mparams.page_size - SIZE_T_ONE)) & ~(mparams.page_size - SIZE_T_←ONE)`
- #define `granularity_align(S)`
- #define `mmap_align(S)` `page_align(S)`
- #define `SYS_ALLOC_PADDING` `(TOP_FOOT_SIZE + MALLOC_ALIGNMENT)`
- #define `is_page_aligned(S)` `((size_t)(S) & (mparams.page_size - SIZE_T_ONE)) == 0)`
- #define `is_granularity_aligned(S)` `((size_t)(S) & (mparams.granularity - SIZE_T_ONE)) == 0)`
- #define `segment_holds(S, A)` `((char*)(A) >= S->base && (char*)(A) < S->base + S->size)`
- #define `should_trim(M, s)` `((s) > (M)->trim_check)`
- #define `TOP_FOOT_SIZE` `(align_offset(chunk2mem(0))+pad_request(sizeof(struct malloc_segment))+MI←N_CHUNK_SIZE)`
- #define `PREACTION(M)` `((use_lock(M))? ACQUIRE_LOCK(&(M)->mutex) : 0)`
- #define `POSTACTION(M)` `{ if (use_lock(M)) RELEASE_LOCK(&(M)->mutex); }`
- #define `CORRUPTION_ERROR_ACTION(m)` `ABORT`

- #define `USAGE_ERROR_ACTION`(m, p) `ABORT`
- #define `check_free_chunk`(M, P)
- #define `check_inuse_chunk`(M, P)
- #define `check_malloced_chunk`(M, P, N)
- #define `check_mmapped_chunk`(M, P)
- #define `check_malloc_state`(M)
- #define `check_top_chunk`(M, P)
- #define `is_small`(s) (((s) >> `SMALLBIN_SHIFT`) < `NSMALLBINS`)
- #define `small_index`(s) (`binmap_t`)((s) >> `SMALLBIN_SHIFT`)
- #define `small_index2size`(i) ((i) << `SMALLBIN_SHIFT`)
- #define `MIN_SMALL_INDEX` (`small_index`(`MIN_CHUNK_SIZE`))
- #define `smallbin_at`(M, i) ((`sbinptr`)((`char*`)&(M)->`smallbins`[(i)<<1]))
- #define `treebin_at`(M, i) (&(M)->`treebins`[i])
- #define `compute_tree_index`(S, I)
- #define `bit_for_tree_index`(i) (i == `NTREEBINS`-1)? (`SIZE_T_BITSIZE`-1) : (((i) >> 1) + `TREEBIN_SHIFT` - 2)
- #define `leftshift_for_tree_index`(i)
- #define `minsize_for_tree_index`(i)
- #define `idx2bit`(i) ((`binmap_t`)1 << (i))
- #define `mark_smallmap`(M, i) ((M)->`smallmap` |= `idx2bit`(i))
- #define `clear_smallmap`(M, i) ((M)->`smallmap` &= ~`idx2bit`(i))
- #define `smallmap_is_marked`(M, i) ((M)->`smallmap` & `idx2bit`(i))
- #define `mark_treemap`(M, i) ((M)->`treemap` |= `idx2bit`(i))
- #define `clear_treemap`(M, i) ((M)->`treemap` &= ~`idx2bit`(i))
- #define `treemap_is_marked`(M, i) ((M)->`treemap` & `idx2bit`(i))
- #define `least_bit`(x) ((x) & -(x))
- #define `left_bits`(x) ((x << 1) | -(x << 1))
- #define `same_or_left_bits`(x) ((x) | -(x))
- #define `compute_bit2idx`(X, I)
- #define `ok_address`(M, a) ((`char*`)(a) >= (M)->`least_addr`)
- #define `ok_next`(p, n) ((`char*`)(p) < (`char*`)(n))
- #define `ok_inuse`(p) `is_inuse`(p)
- #define `ok_pinuse`(p) `pinuse`(p)
- #define `ok_magic`(M) (1)
- #define `RTCHECK`(e) (e)
- #define `mark_inuse_foot`(M, p, s)
- #define `set_inuse`(M, p, s)
- #define `set_inuse_and_pinuse`(M, p, s)
- #define `set_size_and_pinuse_of_inuse_chunk`(M, p, s) ((p)->`head` = (s|`PINUSE_BIT`|`CINUSE_BIT`))
- #define `insert_small_chunk`(M, P, S)
- #define `unlink_small_chunk`(M, P, S)
- #define `unlink_first_small_chunk`(M, B, P, I)
- #define `replace_dv`(M, P, S)
- #define `insert_large_chunk`(M, X, S)
- #define `unlink_large_chunk`(M, X)
- #define `insert_chunk`(M, P, S)
- #define `unlink_chunk`(M, P, S)
- #define `internal_malloc`(m, b) `mSPACE_malloc`(m, b)
- #define `internal_free`(m, mem) `mSPACE_free`(m, mem);

Typedefs

- typedef void * [mspace](#)
- typedef struct [malloc_chunk](#) [mchunk](#)
- typedef struct [malloc_chunk](#) * [mchunkptr](#)
- typedef struct [malloc_chunk](#) * [sbinptr](#)
- typedef unsigned int [bindex_t](#)
- typedef unsigned int [binmap_t](#)
- typedef unsigned int [flag_t](#)
- typedef struct [malloc_tree_chunk](#) [tchunk](#)
- typedef struct [malloc_tree_chunk](#) * [tchunkptr](#)
- typedef struct [malloc_tree_chunk](#) * [tbinptr](#)
- typedef struct [malloc_segment](#) [msegment](#)
- typedef struct [malloc_segment](#) * [msegmentptr](#)
- typedef struct [malloc_state](#) * [mstate](#)

Functions

- [DLMALLOC_EXPORT](#) [mspace](#) [create_mspace](#) (size_t capacity, int locked)
- [DLMALLOC_EXPORT](#) size_t [destroy_mspace](#) ([mspace](#) msp)
- [DLMALLOC_EXPORT](#) [mspace](#) [create_mspace_with_base](#) (void *base, size_t capacity, int locked)
- [DLMALLOC_EXPORT](#) int [mspace_track_large_chunks](#) ([mspace](#) msp, int enable)
- [DLMALLOC_EXPORT](#) void * [mspace_malloc](#) ([mspace](#) msp, size_t bytes)
- [DLMALLOC_EXPORT](#) void [mspace_free](#) ([mspace](#) msp, void *mem)
- [DLMALLOC_EXPORT](#) void * [mspace_realloc](#) ([mspace](#) msp, void *mem, size_t newsize)
- [DLMALLOC_EXPORT](#) void * [mspace_calloc](#) ([mspace](#) msp, size_t n_elements, size_t elem_size)
- [DLMALLOC_EXPORT](#) void * [mspace_memalign](#) ([mspace](#) msp, size_t alignment, size_t bytes)
- [DLMALLOC_EXPORT](#) void ** [mspace_independent_calloc](#) ([mspace](#) msp, size_t n_elements, size_t elem_size, void *chunks[])↔
- [DLMALLOC_EXPORT](#) void ** [mspace_independent_comalloc](#) ([mspace](#) msp, size_t n_elements, size_t sizes[], void *chunks[])↔
- [DLMALLOC_EXPORT](#) size_t [mspace_footprint](#) ([mspace](#) msp)
- [DLMALLOC_EXPORT](#) size_t [mspace_max_footprint](#) ([mspace](#) msp)
- [DLMALLOC_EXPORT](#) struct mallinfo [mspace_mallinfo](#) ([mspace](#) msp)
- [DLMALLOC_EXPORT](#) size_t [mspace_usable_size](#) (const void *mem)
- [DLMALLOC_EXPORT](#) void [mspace_malloc_stats](#) ([mspace](#) msp)
- [DLMALLOC_EXPORT](#) int [mspace_trim](#) ([mspace](#) msp, size_t pad)
- [DLMALLOC_EXPORT](#) int [mspace_mallopt](#) (int, int)
- void * [mspace_realloc_in_place](#) ([mspace](#) msp, void *oldmem, size_t bytes)
- size_t [mspace_bulk_free](#) ([mspace](#) msp, void *array[], size_t nelem)
- size_t [mspace_footprint_limit](#) ([mspace](#) msp)
- size_t [mspace_set_footprint_limit](#) ([mspace](#) msp, size_t bytes)

7.17.1 Macro Definition Documentation

7.17.1.1 _STRUCT_MALLINFO

```
#define _STRUCT_MALLINFO
```

Definition at line 766 of file dl_malloc.c.

7.17.1.2 ABORT

```
#define ABORT abort()
```

Definition at line 627 of file dl_malloc.c.

7.17.1.3 ABORT_ON_ASSERT_FAILURE

```
#define ABORT_ON_ASSERT_FAILURE 1
```

Definition at line 630 of file dl_malloc.c.

7.17.1.4 ACQUIRE_LOCK

```
#define ACQUIRE_LOCK(  
    lk ) pthread_mutex_lock(lk)
```

Definition at line 2007 of file dl_malloc.c.

7.17.1.5 ACQUIRE_MALLOC_GLOBAL_LOCK

```
#define ACQUIRE_MALLOC_GLOBAL_LOCK( ) ACQUIRE_LOCK(&malloc_global_mutex);
```

Definition at line 2041 of file dl_malloc.c.

7.17.1.6 align_as_chunk

```
#define align_as_chunk(  
    A ) (mchunkptr)((A) + align_offset(chunk2mem(A)))
```

Definition at line 2224 of file dl_malloc.c.

7.17.1.7 align_offset

```
#define align_offset(  
    A )
```

Value:

```
((((size_t)(A) & CHUNK_ALIGN_MASK) == 0)? 0 :\n ((MALLOC_ALIGNMENT - ((size_t)(A) & CHUNK_ALIGN_MASK)) &\n  CHUNK_ALIGN_MASK))
```

Definition at line 1624 of file dl_malloc.c.

7.17.1.8 assert

```
#define assert(  
    x )
```

Definition at line 1455 of file dl_malloc.c.

7.17.1.9 bit_for_tree_index

```
#define bit_for_tree_index(  
    i ) ( i == NTREEBINS-1)? (SIZE_T_BITSIZE-1) : ((i) >> 1) + TREEBIN_SHIFT - 2)
```

Definition at line 2903 of file dl_malloc.c.

7.17.1.10 CALL_DIRECT_MMAP

```
#define CALL_DIRECT_MMAP(  
    s ) DIRECT_MMAP_DEFAULT(s)
```

Definition at line 1744 of file dl_malloc.c.

7.17.1.11 CALL_MMAP

```
#define CALL_MMAP(  
    s ) MMAP_DEFAULT(s)
```

Definition at line 1734 of file dl_malloc.c.

7.17.1.12 CALL_MORECORE

```
#define CALL_MORECORE(  
    s ) MFAIL
```

Define CALL_MORECORE

Definition at line 1722 of file dl_malloc.c.

7.17.1.13 CALL_MREMAP

```
#define CALL_MREMAP(  
    addr,  
    osz,  
    nsz,  
    mv ) MFAIL
```

Define CALL_MREMAP

Definition at line 1767 of file dl_malloc.c.

7.17.1.14 CALL_MUNMAP

```
#define CALL_MUNMAP(  
    a,  
    s ) MUNMAP_DEFAULT((a), (s))
```

Definition at line 1739 of file dl_malloc.c.

7.17.1.15 calloc_must_clear

```
#define calloc_must_clear(  
    p ) (!is_mmapped(p))
```

Definition at line 2300 of file dl_malloc.c.

7.17.1.16 check_free_chunk

```
#define check_free_chunk(  
    M,  
    P )
```

Definition at line 2798 of file dl_malloc.c.

7.17.1.17 check_inuse_chunk

```
#define check_inuse_chunk(  
    M,  
    P )
```

Definition at line 2799 of file dl_malloc.c.

7.17.1.18 `check_malloc_state`

```
#define check_malloc_state(  
    M )
```

Definition at line 2802 of file `dl_malloc.c`.

7.17.1.19 `check_mallosed_chunk`

```
#define check_mallosed_chunk(  
    M,  
    P,  
    N )
```

Definition at line 2800 of file `dl_malloc.c`.

7.17.1.20 `check_mmapped_chunk`

```
#define check_mmapped_chunk(  
    M,  
    P )
```

Definition at line 2801 of file `dl_malloc.c`.

7.17.1.21 `check_top_chunk`

```
#define check_top_chunk(  
    M,  
    P )
```

Definition at line 2803 of file `dl_malloc.c`.

7.17.1.22 `chunk2mem`

```
#define chunk2mem(  
    p ) ((void*)((char*)(p) + TWO_SIZE_T_SIZES))
```

Definition at line 2221 of file `dl_malloc.c`.

7.17.1.23 CHUNK_ALIGN_MASK

```
#define CHUNK_ALIGN_MASK (MALLOC_ALIGNMENT - SIZE_T_ONE)
```

Definition at line 1618 of file dl_malloc.c.

7.17.1.24 chunk_minus_offset

```
#define chunk_minus_offset(  
    p,  
    s ) ((mchunkptr)(((char*)(p)) - (s)))
```

Definition at line 2273 of file dl_malloc.c.

7.17.1.25 CHUNK_OVERHEAD

```
#define CHUNK_OVERHEAD (SIZE_T_SIZE)
```

Definition at line 2208 of file dl_malloc.c.

7.17.1.26 chunk_plus_offset

```
#define chunk_plus_offset(  
    p,  
    s ) ((mchunkptr)(((char*)(p)) + (s)))
```

Definition at line 2272 of file dl_malloc.c.

7.17.1.27 chunksize

```
#define chunksize(  
    p ) ((p)->head & ~(FLAG_BITS))
```

Definition at line 2265 of file dl_malloc.c.

7.17.1.28 cinuse

```
#define cinuse(  
    p ) ((p)->head & CINUSE_BIT)
```

Definition at line 2259 of file dl_malloc.c.

7.17.1.29 CINUSE_BIT

```
#define CINUSE_BIT (SIZE_T_TWO)
```

Definition at line 2250 of file dl_malloc.c.

7.17.1.30 clear_flag4

```
#define clear_flag4(  
    p ) ((p)->head &= ~FLAG4_BIT)
```

Definition at line 2269 of file dl_malloc.c.

7.17.1.31 clear_pinuse

```
#define clear_pinuse(  
    p ) ((p)->head &= ~PINUSE_BIT)
```

Definition at line 2267 of file dl_malloc.c.

7.17.1.32 clear_smallmap

```
#define clear_smallmap(  
    M,  
    i ) ((M)->smallmap &= ~idx2bit(i))
```

Definition at line 2924 of file dl_malloc.c.

7.17.1.33 clear_treemap

```
#define clear_treemap(  
    M,  
    i ) ((M)->treemap &= ~idx2bit(i))
```

Definition at line 2928 of file dl_malloc.c.

7.17.1.34 CMFAIL

```
#define CMFAIL ((char*)(MFAIL)) /* defined for convenience */
```

Definition at line 1639 of file dl_malloc.c.

7.17.1.35 compute_bit2idx

```
#define compute_bit2idx(  
    X,  
    I )
```

Value:

```
{\  
  unsigned int Y = X - 1;\br/>  unsigned int K = Y >> (16-4) & 16;\br/>  unsigned int N = K;      Y >>= K;\br/>  N += K = Y >> (8-3) & 8;  Y >>= K;\br/>  N += K = Y >> (4-2) & 4;  Y >>= K;\br/>  N += K = Y >> (2-1) & 2;  Y >>= K;\br/>  N += K = Y >> (1-0) & 1;  Y >>= K;\br/>  I = (bindex_t)(N + Y);\br/>}
```

Definition at line 2970 of file dl_malloc.c.

7.17.1.36 compute_tree_index

```
#define compute_tree_index(  
    S,  
    I )
```

Value:

```
{\  
  size_t X = S >> TREEBIN_SHIFT;\br/>  if (X == 0)\br/>    I = 0;\br/>  else if (X > 0xFFFF)\br/>    I = NTREEBINS-1;\br/>  else {\br/>    unsigned int Y = (unsigned int)X;\br/>    unsigned int N = ((Y - 0x100) >> 16) & 8;\br/>    unsigned int K = (((Y <= N) - 0x1000) >> 16) & 4;\br/>    N += K;\br/>    N += K = (((Y <= K) - 0x4000) >> 16) & 2;\br/>    K = 14 - N + ((Y <= K) >> 15);\br/>    I = (K << 1) + ((S >> (K + (TREEBIN_SHIFT-1) & 1));\  
  }\br/>}
```

Definition at line 2883 of file dl_malloc.c.

7.17.1.37 CORRUPTION_ERROR_ACTION

```
#define CORRUPTION_ERROR_ACTION(  
    m ) ABORT
```

Definition at line 2784 of file dl_malloc.c.

7.17.1.38 DEBUG

```
#define DEBUG 0
```

Definition at line 1457 of file dl_malloc.c.

7.17.1.39 DEFAULT_GRANULARITY

```
#define DEFAULT_GRANULARITY ((size_t)64U * (size_t)1024U)
```

Definition at line 680 of file dl_malloc.c.

7.17.1.40 DEFAULT_MMAP_THRESHOLD

```
#define DEFAULT_MMAP_THRESHOLD ((size_t)256U * (size_t)1024U)
```

Definition at line 692 of file dl_malloc.c.

7.17.1.41 DEFAULT_TRIM_THRESHOLD

```
#define DEFAULT_TRIM_THRESHOLD ((size_t)2U * (size_t)1024U * (size_t)1024U)
```

Definition at line 685 of file dl_malloc.c.

7.17.1.42 DESTROY_LOCK

```
#define DESTROY_LOCK(  
    lk ) pthread_mutex_destroy(lk)
```

Definition at line 2011 of file dl_malloc.c.

7.17.1.43 DIRECT_MMAP_DEFAULT

```
#define DIRECT_MMAP_DEFAULT(  
    s ) MMAP_DEFAULT(s)
```

Definition at line 1665 of file dl_malloc.c.

7.17.1.44 disable_contiguous

```
#define disable_contiguous(  
    M ) ((M)->mflags |= USE_NONCONTIGUOUS_BIT)
```

Definition at line 2664 of file dl_malloc.c.

7.17.1.45 disable_lock

```
#define disable_lock(  
    M )
```

Definition at line 2652 of file dl_malloc.c.

7.17.1.46 disable_mmap

```
#define disable_mmap(  
    M ) ((M)->mflags &= ~USE_MMAP_BIT)
```

Definition at line 2658 of file dl_malloc.c.

7.17.1.47 DLMALLOC_EXPORT

```
#define DLMALLOC_EXPORT extern
```

Definition at line 533 of file dl_malloc.c.

7.17.1.48 DLMALLOC_VERSION

```
#define DLMALLOC_VERSION 20806
```

Definition at line 529 of file dl_malloc.c.

7.17.1.49 enable_lock

```
#define enable_lock(  
    M ) ((M)->mflags |= USE_LOCK_BIT)
```

Definition at line 2648 of file dl_malloc.c.

7.17.1.50 enable_mmap

```
#define enable_mmap(  
    M ) ((M)->mflags |= USE_MMMap_BIT)
```

Definition at line 2656 of file dl_malloc.c.

7.17.1.51 ensure_initialization

```
#define ensure_initialization( ) (void)(mparams.magic != 0 || init_mparams())
```

Definition at line 2630 of file dl_malloc.c.

7.17.1.52 EXTERN_BIT

```
#define EXTERN_BIT (8U)
```

Definition at line 1774 of file dl_malloc.c.

7.17.1.53 FENCEPOST_HEAD

```
#define FENCEPOST_HEAD (INUSE_BITS|SIZE_T_SIZE)
```

Definition at line 2256 of file dl_malloc.c.

7.17.1.54 FLAG4_BIT

```
#define FLAG4_BIT (SIZE_T_FOUR)
```

Definition at line 2251 of file dl_malloc.c.

7.17.1.55 flag4inuse

```
#define flag4inuse(  
    p ) ((p)->head & FLAG4_BIT)
```

Definition at line 2261 of file dl_malloc.c.

7.17.1.56 FLAG_BITS

```
#define FLAG_BITS (PINUSE_BIT|CINUSE_BIT|FLAG4_BIT)
```

Definition at line 2253 of file dl_malloc.c.

7.17.1.57 FOOTERS

```
#define FOOTERS 0
```

Definition at line 624 of file dl_malloc.c.

7.17.1.58 FORCEINLINE

```
#define FORCEINLINE
```

Definition at line 813 of file dl_malloc.c.

7.17.1.59 FOUR_SIZE_T_SIZES

```
#define FOUR_SIZE_T_SIZES (SIZE_T_SIZE<<2)
```

Definition at line 1613 of file dl_malloc.c.

7.17.1.60 get_foot

```
#define get_foot(  
    p,  
    s ) (((mchunkptr)((char*)(p) + (s)))->prev_foot)
```

Definition at line 2283 of file dl_malloc.c.

7.17.1.61 granularity_align

```
#define granularity_align(  
    S )
```

Value:

```
((S) + (mparams.granularity - SIZE_T_ONE))\  
& ~(mparams.granularity - SIZE_T_ONE)
```

Definition at line 2676 of file dl_malloc.c.

7.17.1.62 HALF_MAX_SIZE_T

```
#define HALF_MAX_SIZE_T (MAX_SIZE_T / 2U)
```

Definition at line 1615 of file dl_malloc.c.

7.17.1.63 HAVE_MMAP

```
#define HAVE_MMAP 1
```

Definition at line 643 of file dl_malloc.c.

7.17.1.64 HAVE_MORECORE

```
#define HAVE_MORECORE 0
```

Definition at line 663 of file dl_malloc.c.

7.17.1.65 HAVE_MREMAP

```
#define HAVE_MREMAP 0
```

Definition at line 655 of file dl_malloc.c.

7.17.1.66 idx2bit

```
#define idx2bit(  
    i ) ((binmap_t) (1) << (i))
```

Definition at line 2920 of file dl_malloc.c.

7.17.1.67 INITIAL_LOCK

```
#define INITIAL_LOCK(  
    lk ) pthread_init_lock(lk)
```

Definition at line 2010 of file dl_malloc.c.

7.17.1.68 INSECURE

```
#define INSECURE 0
```

Definition at line 637 of file dl_malloc.c.

7.17.1.69 insert_chunk

```
#define insert_chunk(  
    M,  
    P,  
    S )
```

Value:

```
if (is_small(S)) insert_small_chunk(M, P, S)\  
else { tchunkptr TP = (tchunkptr) (P); insert_large_chunk(M, TP, S); }
```

Definition at line 3791 of file dl_malloc.c.

7.17.1.70 insert_large_chunk

```
#define insert_large_chunk(  
    M,  
    X,  
    S )
```

Definition at line 3650 of file dl_malloc.c.

7.17.1.71 insert_small_chunk

```
#define insert_small_chunk(
    M,
    P,
    S )
```

Value:

```
{\
  bindex_t I = small_index(S);\
  mchunkptr B = smallbin_at(M, I);\
  mchunkptr F = B;\
  assert(S >= MIN_CHUNK_SIZE);\
  if (!smallmap_is_marked(M, I))\
    mark_smallmap(M, I);\
  else if (RTCHECK(ok_address(M, B->fd)))\
    F = B->fd;\
  else {\
    CORRUPTION_ERROR_ACTION(M);\
  }\
  B->fd = P;\
  F->bk = P;\
  P->fd = F;\
  P->bk = B;\
}
```

Definition at line 3572 of file dl_malloc.c.

7.17.1.72 internal_free

```
#define internal_free(
    m,
    mem ) mspace_free(m, mem);
```

Definition at line 3804 of file dl_malloc.c.

7.17.1.73 internal_malloc

```
#define internal_malloc(
    m,
    b ) mspace_malloc(m, b)
```

Definition at line 3803 of file dl_malloc.c.

7.17.1.74 INUSE_BITS

```
#define INUSE_BITS (PINUSE_BIT|CINUSE_BIT)
```

Definition at line 2252 of file dl_malloc.c.

7.17.1.75 is_aligned

```
#define is_aligned(  
    A ) (((size_t)(A) & (CHUNK_ALIGN_MASK)) == 0)
```

Definition at line 1621 of file dl_malloc.c.

7.17.1.76 is_extern_segment

```
#define is_extern_segment(  
    S ) ((S)->sflags & EXTERN_BIT)
```

Definition at line 2480 of file dl_malloc.c.

7.17.1.77 is_granularity_aligned

```
#define is_granularity_aligned(  
    S ) (((size_t)(S) & (mparams.granularity - SIZE_T_ONE)) == 0)
```

Definition at line 2693 of file dl_malloc.c.

7.17.1.78 is_initialized

```
#define is_initialized(  
    M ) ((M)->top != 0)
```

Definition at line 2641 of file dl_malloc.c.

7.17.1.79 is_inuse

```
#define is_inuse(  
    p ) ((p)->head & INUSE_BITS) != PINUSE_BIT)
```

Definition at line 2262 of file dl_malloc.c.

7.17.1.80 is_mmapped

```
#define is_mmapped(  
    p ) ((p)->head & INUSE_BITS) == 0)
```

Definition at line 2263 of file dl_malloc.c.

7.17.1.81 is_mmapped_segment

```
#define is_mmapped_segment(  
    S ) ((S)->sflags & USE_MMAP_BIT)
```

Definition at line 2479 of file dl_malloc.c.

7.17.1.82 is_page_aligned

```
#define is_page_aligned(  
    S ) (((size_t)(S) & (mparams.page_size - SIZE_T_ONE)) == 0)
```

Definition at line 2691 of file dl_malloc.c.

7.17.1.83 is_small

```
#define is_small(  
    S ) (((S) >> SMALLBIN_SHIFT) < NSMALLBINS)
```

Definition at line 2829 of file dl_malloc.c.

7.17.1.84 least_bit

```
#define least_bit(  
    x ) ((x) & -(x))
```

Definition at line 2932 of file dl_malloc.c.

7.17.1.85 left_bits

```
#define left_bits(  
    x ) ((x<<1) | -(x<<1))
```

Definition at line 2935 of file dl_malloc.c.

7.17.1.86 leftmost_child

```
#define leftmost_child(  
    t ) ((t)->child[0] != 0? (t)->child[0] : (t)->child[1])
```

Definition at line 2413 of file dl_malloc.c.

7.17.1.87 leftshift_for_tree_index

```
#define leftshift_for_tree_index(  
    i )
```

Value:

```
((i == NTREEBINS-1)? 0 : \  
 ((SIZE_T_BITSIZE-SIZE_T_ONE) - ((i) >> 1) +  
  TREEBIN_SHIFT - 2))
```

Definition at line 2907 of file dl_malloc.c.

7.17.1.88 LOCK_AT_FORK

```
#define LOCK_AT_FORK 0
```

Definition at line 1531 of file dl_malloc.c.

7.17.1.89 M_GRANULARITY

```
#define M_GRANULARITY (-2)
```

Definition at line 731 of file dl_malloc.c.

7.17.1.90 M_MMAP_THRESHOLD

```
#define M_MMAP_THRESHOLD (-3)
```

Definition at line 732 of file dl_malloc.c.

7.17.1.91 M_TRIM_THRESHOLD

```
#define M_TRIM_THRESHOLD (-1)
```

Definition at line 730 of file dl_malloc.c.

7.17.1.92 MALLINFO_FIELD_TYPE

```
#define MALLINFO_FIELD_TYPE size_t
```

Definition at line 714 of file dl_malloc.c.

7.17.1.93 MALLOC_ALIGNMENT

```
#define MALLOC_ALIGNMENT ((size_t)(2 * sizeof(void *)))
```

Definition at line 621 of file dl_malloc.c.

7.17.1.94 MALLOC_FAILURE_ACTION

```
#define MALLOC_FAILURE_ACTION errno = ENOMEM;
```

Definition at line 659 of file dl_malloc.c.

7.17.1.95 malloc_getpagesize

```
#define malloc_getpagesize ((size_t)4096U)
```

Definition at line 1589 of file dl_malloc.c.

7.17.1.96 MALLOC_INSPECT_ALL

```
#define MALLOC_INSPECT_ALL 0
```

Definition at line 640 of file dl_malloc.c.

7.17.1.97 mark_inuse_foot

```
#define mark_inuse_foot(  
    M,  
    P,  
    S )
```

Definition at line 3051 of file dl_malloc.c.

7.17.1.98 mark_smallmap

```
#define mark_smallmap(  
    M,  
    i ) ((M)->smallmap |= idx2bit(i))
```

Definition at line 2923 of file dl_malloc.c.

7.17.1.99 mark_treemap

```
#define mark_treemap(  
    M,  
    i ) ((M)->treemap |= idx2bit(i))
```

Definition at line 2927 of file dl_malloc.c.

7.17.1.100 MAX_RELEASE_CHECK_RATE

```
#define MAX_RELEASE_CHECK_RATE 4095
```

Definition at line 699 of file dl_malloc.c.

7.17.1.101 MAX_REQUEST

```
#define MAX_REQUEST ((-MIN_CHUNK_SIZE) << 2)
```

Definition at line 2227 of file dl_malloc.c.

7.17.1.102 MAX_SIZE_T

```
#define MAX_SIZE_T (~(size_t)0)
```

Definition at line 588 of file dl_malloc.c.

7.17.1.103 MAX_SMALL_REQUEST

```
#define MAX_SMALL_REQUEST (MAX_SMALL_SIZE - CHUNK_ALIGN_MASK - CHUNK_OVERHEAD)
```

Definition at line 2580 of file dl_malloc.c.

7.17.1.104 MAX_SMALL_SIZE

```
#define MAX_SMALL_SIZE (MIN_LARGE_SIZE - SIZE_T_ONE)
```

Definition at line 2579 of file dl_malloc.c.

7.17.1.105 MCHUNK_SIZE

```
#define MCHUNK_SIZE (sizeof(mchunk))
```

Definition at line 2203 of file dl_malloc.c.

7.17.1.106 mem2chunk

```
#define mem2chunk(  
    mem ) ((mchunkptr)((char*)(mem) - TWO_SIZE_T_SIZES))
```

Definition at line 2222 of file dl_malloc.c.

7.17.1.107 MFAIL

```
#define MFAIL ((void*)(MAX_SIZE_T))
```

Definition at line 1638 of file dl_malloc.c.

7.17.1.108 MIN_CHUNK_SIZE

```
#define MIN_CHUNK_SIZE ((MCHUNK_SIZE + CHUNK_ALIGN_MASK) & ~CHUNK_ALIGN_MASK)
```

Definition at line 2217 of file dl_malloc.c.

7.17.1.109 MIN_LARGE_SIZE

```
#define MIN_LARGE_SIZE (SIZE_T_ONE << TREEBIN_SHIFT)
```

Definition at line 2578 of file dl_malloc.c.

7.17.1.110 MIN_REQUEST

```
#define MIN_REQUEST (MIN_CHUNK_SIZE - CHUNK_OVERHEAD - SIZE_T_ONE)
```

Definition at line 2228 of file dl_malloc.c.

7.17.1.111 MIN_SMALL_INDEX

```
#define MIN_SMALL_INDEX (small_index(MIN_CHUNK_SIZE))
```

Definition at line 2832 of file dl_malloc.c.

7.17.1.112 minsize_for_tree_index

```
#define minsize_for_tree_index(  
    i )
```

Value:

```
((SIZE_T_ONE << (((i) >> 1) + TREEBIN_SHIFT)) | \  
 ((size_t)((i) & SIZE_T_ONE) << (((i) >> 1) + TREEBIN_SHIFT - 1)))
```

Definition at line 2912 of file dl_malloc.c.

7.17.1.113 MLOCK_T

```
#define MLOCK_T pthread_mutex_t
```

Definition at line 2006 of file dl_malloc.c.

7.17.1.114 mmap_align

```
#define mmap_align(  
    S ) page_align(S)
```

Definition at line 2685 of file dl_malloc.c.

7.17.1.115 MMAP_CHUNK_OVERHEAD

```
#define MMAP_CHUNK_OVERHEAD (TWO_SIZE_T_SIZES)
```

Definition at line 2212 of file dl_malloc.c.

7.17.1.116 MMAP_CLEARS

```
#define MMAP_CLEARS 1
```

Definition at line 646 of file dl_malloc.c.

7.17.1.117 MMAP_DEFAULT

```
#define MMAP_DEFAULT(  
    s )
```

Value:

```
((dev_zero_fd < 0) ? \  
    (dev_zero_fd = open("/dev/zero", O_RDWR), \  
    mmap(0, (s), MMAP_PROT, MMAP_FLAGS, dev_zero_fd, 0)) : \  
    mmap(0, (s), MMAP_PROT, MMAP_FLAGS, dev_zero_fd, 0))
```

Definition at line 1659 of file dl_malloc.c.

7.17.1.118 MMAP_FLAGS

```
#define MMAP_FLAGS (MAP_PRIVATE)
```

Definition at line 1657 of file dl_malloc.c.

7.17.1.119 MMAP_FOOT_PAD

```
#define MMAP_FOOT_PAD (FOUR_SIZE_T_SIZES)
```

Definition at line 2214 of file dl_malloc.c.

7.17.1.120 MMAP_PROT

```
#define MMAP_PROT (PROT_READ|PROT_WRITE)
```

Definition at line 1645 of file dl_malloc.c.

7.17.1.121 MORECORE_CONTIGUOUS

```
#define MORECORE_CONTIGUOUS 0
```

Definition at line 669 of file dl_malloc.c.

7.17.1.122 MSPACES

```
#define MSPACES 1
```

Definition at line 615 of file dl_malloc.c.

7.17.1.123 MUNMAP_DEFAULT

```
#define MUNMAP_DEFAULT(  
    a,  
    s ) munmap((a), (s))
```

Definition at line 1644 of file dl_malloc.c.

7.17.1.124 next_chunk

```
#define next_chunk(  
    p ) ((mchunkptr) ( ((char*) (p)) + ((p)->head & ~FLAG_BITS)))
```

Definition at line 2276 of file dl_malloc.c.

7.17.1.125 next_pinuse

```
#define next_pinuse(  
    p ) ((next_chunk(p)->head) & PINUSE_BIT)
```

Definition at line 2280 of file dl_malloc.c.

7.17.1.126 NO_MALLINFO

```
#define NO_MALLINFO 0
```

Definition at line 711 of file dl_malloc.c.

7.17.1.127 NO_MALLOC_STATS

```
#define NO_MALLOC_STATS 0
```

Definition at line 717 of file dl_malloc.c.

7.17.1.128 NO_SEGMENT_TRAVERSAL

```
#define NO_SEGMENT_TRAVERSAL 0
```

Definition at line 720 of file dl_malloc.c.

7.17.1.129 NOINLINE

```
#define NOINLINE
```

Definition at line 802 of file dl_malloc.c.

7.17.1.130 NSMALLBINS

```
#define NSMALLBINS (32U)
```

Definition at line 2573 of file dl_malloc.c.

7.17.1.131 NTREEBINS

```
#define NTREEBINS (32U)
```

Definition at line 2574 of file dl_malloc.c.

7.17.1.132 ok_address

```
#define ok_address(  
    M,  
    a ) ((char*)(a) >= (M)->least_addr)
```

Definition at line 3014 of file dl_malloc.c.

7.17.1.133 ok_inuse

```
#define ok_inuse(  
    p ) is_inuse(p)
```

Definition at line 3018 of file dl_malloc.c.

7.17.1.134 ok_magic

```
#define ok_magic(  
    M ) (1)
```

Definition at line 3033 of file dl_malloc.c.

7.17.1.135 ok_next

```
#define ok_next(  
    p,  
    n ) ((char*)(p) < (char*)(n))
```

Definition at line 3016 of file dl_malloc.c.

7.17.1.136 ok_pinuse

```
#define ok_pinuse(  
    p ) pinuse(p)
```

Definition at line 3020 of file dl_malloc.c.

7.17.1.137 ONLY_MSPACES

```
#define ONLY_MSPACES 1
```

Definition at line 2 of file dl_malloc.c.

7.17.1.138 overhead_for

```
#define overhead_for(  
    p ) (is_mmapped(p)? MMAP_CHUNK_OVERHEAD : CHUNK_OVERHEAD)
```

Definition at line 2295 of file dl_malloc.c.

7.17.1.139 pad_request

```
#define pad_request(  
    req ) (((req) + CHUNK_OVERHEAD + CHUNK_ALIGN_MASK) & ~CHUNK_ALIGN_MASK)
```

Definition at line 2231 of file dl_malloc.c.

7.17.1.140 page_align

```
#define page_align(  
    S ) (((S) + (mparams.page_size - SIZE_T_ONE)) & ~(mparams.page_size - SIZE_T_O↵  
NE))
```

Definition at line 2672 of file dl_malloc.c.

7.17.1.141 pinuse

```
#define pinuse(  
    p ) ((p)->head & PINUSE_BIT)
```

Definition at line 2260 of file dl_malloc.c.

7.17.1.142 PINUSE_BIT

```
#define PINUSE_BIT (SIZE_T_ONE)
```

Definition at line 2249 of file dl_malloc.c.

7.17.1.143 POSTACTION

```
#define POSTACTION(  
    M ) { if (use_lock(M)) RELEASE_LOCK(&(M)->mutex); }
```

Definition at line 2749 of file dl_malloc.c.

7.17.1.144 PREACTION

```
#define PREACTION(  
    M ) ((use_lock(M)) ? ACQUIRE_LOCK(&(M)->mutex) : 0)
```

Definition at line 2748 of file dl_malloc.c.

7.17.1.145 prev_chunk

```
#define prev_chunk(  
    p ) ((mchunkptr) ( ((char*)(p)) - ((p)->prev_foot) ))
```

Definition at line 2277 of file dl_malloc.c.

7.17.1.146 PROCEED_ON_ERROR

```
#define PROCEED_ON_ERROR 0
```

Definition at line 633 of file dl_malloc.c.

7.17.1.147 RELEASE_LOCK

```
#define RELEASE_LOCK(  
    lk ) pthread_mutex_unlock(lk)
```

Definition at line 2008 of file dl_malloc.c.

7.17.1.148 RELEASE_MALLOC_GLOBAL_LOCK

```
#define RELEASE_MALLOC_GLOBAL_LOCK( ) RELEASE_LOCK(&malloc_global_mutex);
```

Definition at line 2045 of file dl_malloc.c.

7.17.1.149 `replace_dv`

```
#define replace_dv(  
    M,  
    P,  
    S )
```

Value:

```
{\  
    size_t DVS = M->dvsiz;\  
    assert(is_small(DVS));\  
    if (DVS != 0) {\  
        mchunkptr DV = M->dv;\  
        insert_small_chunk(M, DV, DVS);\  
    }\  
    M->dvsiz = S;\  
    M->dv = P;\  
}
```

Definition at line 3636 of file `dl_malloc.c`.

7.17.1.150 `request2size`

```
#define request2size(  
    req ) (((req) < MIN_REQUEST)? MIN_CHUNK_SIZE : pad_request(req))
```

Definition at line 2235 of file `dl_malloc.c`.

7.17.1.151 `RTCHECK`

```
#define RTCHECK(  
    e ) (e)
```

Definition at line 3041 of file `dl_malloc.c`.

7.17.1.152 `same_or_left_bits`

```
#define same_or_left_bits(  
    x ) ((x) | -(x))
```

Definition at line 2938 of file `dl_malloc.c`.

7.17.1.153 segment_holds

```
#define segment_holds(  
    S,  
    A ) ((char*)(A) >= S->base && (char*)(A) < S->base + S->size)
```

Definition at line 2697 of file dl_malloc.c.

7.17.1.154 set_flag4

```
#define set_flag4(  
    p ) ((p)->head |= FLAG4_BIT)
```

Definition at line 2268 of file dl_malloc.c.

7.17.1.155 set_foot

```
#define set_foot(  
    p,  
    s ) (((mchunkptr)((char*)(p) + (s)))->prev_foot = (s))
```

Definition at line 2284 of file dl_malloc.c.

7.17.1.156 set_free_with_pinuse

```
#define set_free_with_pinuse(  
    p,  
    s,  
    n ) (clear_pinuse(n), set_size_and_pinuse_of_free_chunk(p, s))
```

Definition at line 2291 of file dl_malloc.c.

7.17.1.157 set_inuse

```
#define set_inuse(  
    M,  
    p,  
    s )
```

Value:

```
((p)->head = (((p)->head & PINUSE_BIT) | s | CINUSE_BIT), \  
 ((mchunkptr)((char*)(p) + (s)))->head |= PINUSE_BIT)
```

Definition at line 3056 of file dl_malloc.c.

7.17.1.158 set_inuse_and_pinuse

```
#define set_inuse_and_pinuse(
    M,
    P,
    S )
```

Value:

```
((p)->head = (s|PINUSE_BIT|CINUSE_BIT),\
  ((mchunkptr)((char*)(p) + (s)))->head |= PINUSE_BIT)
```

Definition at line 3061 of file dl_malloc.c.

7.17.1.159 set_lock

```
#define set_lock(
    M,
    L )
```

Value:

```
((M)->mflags = (L)?\
  ((M)->mflags | USE_LOCK_BIT) :\
  ((M)->mflags & ~USE_LOCK_BIT))
```

Definition at line 2666 of file dl_malloc.c.

7.17.1.160 set_size_and_pinuse_of_free_chunk

```
#define set_size_and_pinuse_of_free_chunk(
    P,
    S ) ((p)->head = (s|PINUSE_BIT), set_foot(p, s))
```

Definition at line 2287 of file dl_malloc.c.

7.17.1.161 set_size_and_pinuse_of_inuse_chunk

```
#define set_size_and_pinuse_of_inuse_chunk(
    M,
    P,
    S ) ((p)->head = (s|PINUSE_BIT|CINUSE_BIT))
```

Definition at line 3066 of file dl_malloc.c.

7.17.1.162 should_trim

```
#define should_trim(  
    M,  
    s ) ((s) > (M)->trim_check)
```

Definition at line 2725 of file dl_malloc.c.

7.17.1.163 SIX_SIZE_T_SIZES

```
#define SIX_SIZE_T_SIZES (FOUR_SIZE_T_SIZES+TWO_SIZE_T_SIZES)
```

Definition at line 1614 of file dl_malloc.c.

7.17.1.164 SIZE_T_BITSIZE

```
#define SIZE_T_BITSIZE (sizeof(size_t) << 3)
```

Definition at line 1604 of file dl_malloc.c.

7.17.1.165 SIZE_T_FOUR

```
#define SIZE_T_FOUR ((size_t)4)
```

Definition at line 1611 of file dl_malloc.c.

7.17.1.166 SIZE_T_ONE

```
#define SIZE_T_ONE ((size_t)1)
```

Definition at line 1609 of file dl_malloc.c.

7.17.1.167 SIZE_T_SIZE

```
#define SIZE_T_SIZE (sizeof(size_t))
```

Definition at line 1603 of file dl_malloc.c.

7.17.1.168 SIZE_T_TWO

```
#define SIZE_T_TWO ((size_t)2)
```

Definition at line 1610 of file dl_malloc.c.

7.17.1.169 SIZE_T_ZERO

```
#define SIZE_T_ZERO ((size_t)0)
```

Definition at line 1608 of file dl_malloc.c.

7.17.1.170 small_index

```
#define small_index(  
    s ) (bindex_t)((s) >> SMALLBIN_SHIFT)
```

Definition at line 2830 of file dl_malloc.c.

7.17.1.171 small_index2size

```
#define small_index2size(  
    i ) ((i) << SMALLBIN_SHIFT)
```

Definition at line 2831 of file dl_malloc.c.

7.17.1.172 smallbin_at

```
#define smallbin_at(  
    M,  
    i ) ((sbinptr)((char*)&((M)->smallbins[(i)<<1])))
```

Definition at line 2835 of file dl_malloc.c.

7.17.1.173 SMALLBIN_SHIFT

```
#define SMALLBIN_SHIFT (3U)
```

Definition at line 2575 of file dl_malloc.c.

7.17.1.174 SMALLBIN_WIDTH

```
#define SMALLBIN_WIDTH (SIZE_T_ONE << SMALLBIN_SHIFT)
```

Definition at line 2576 of file dl_malloc.c.

7.17.1.175 smallmap_is_marked

```
#define smallmap_is_marked(  
    M,  
    i ) ((M)->smallmap & idx2bit(i))
```

Definition at line 2925 of file dl_malloc.c.

7.17.1.176 STRUCT_MALLINFO_DECLARED

```
#define STRUCT_MALLINFO_DECLARED 1
```

Definition at line 767 of file dl_malloc.c.

7.17.1.177 SYS_ALLOC_PADDING

```
#define SYS_ALLOC_PADDING (TOP_FOOT_SIZE + MALLOC_ALIGNMENT)
```

Definition at line 2689 of file dl_malloc.c.

7.17.1.178 TOP_FOOT_SIZE

```
#define TOP_FOOT_SIZE (align_offset(chunk2mem(0))+pad_request(sizeof(struct malloc_segment))+M↔  
IN_CHUNK_SIZE)
```

Definition at line 2735 of file dl_malloc.c.

7.17.1.179 treebin_at

```
#define treebin_at(  
    M,  
    i ) (&((M)->treebins[i]))
```

Definition at line 2836 of file dl_malloc.c.

7.17.1.180 TREEBIN_SHIFT

```
#define TREEBIN_SHIFT (8U)
```

Definition at line 2577 of file dl_malloc.c.

7.17.1.181 treemap_is_marked

```
#define treemap_is_marked(  
    M,  
    i ) ((M)->treemap & idx2bit(i))
```

Definition at line 2929 of file dl_malloc.c.

7.17.1.182 TRY_LOCK

```
#define TRY_LOCK(  
    lk ) (!pthread_mutex_trylock(lk))
```

Definition at line 2009 of file dl_malloc.c.

7.17.1.183 TWO_SIZE_T_SIZES

```
#define TWO_SIZE_T_SIZES (SIZE_T_SIZE<<1)
```

Definition at line 1612 of file dl_malloc.c.

7.17.1.184 unlink_chunk

```
#define unlink_chunk(  
    M,  
    P,  
    S )
```

Value:

```
if (is_small(S)) unlink_small_chunk(M, P, S)\  
else { tchunkptr TP = (tchunkptr)(P); unlink_large_chunk(M, TP); }
```

Definition at line 3795 of file dl_malloc.c.

7.17.1.185 unlink_first_small_chunk

```
#define unlink_first_small_chunk(
    M,
    B,
    P,
    I )
```

Value:

```
{\
  mchunkptr F = P->fd;\
  assert(P != B);\
  assert(P != F);\
  assert(chunksize(P) == small_index2size(I));\
  if (B == F) {\
    clear_smallmap(M, I);\
  }\
  else if (RTCHECK(ok_address(M, F) && F->bk == P)) {\
    F->bk = B;\
    B->fd = F;\
  }\
  else {\
    CORRUPTION_ERROR_ACTION(M);\
  }\
}
```

Definition at line 3617 of file dl_malloc.c.

7.17.1.186 unlink_large_chunk

```
#define unlink_large_chunk(
    M,
    X )
```

Definition at line 3718 of file dl_malloc.c.

7.17.1.187 unlink_small_chunk

```
#define unlink_small_chunk(
    M,
    P,
    S )
```

Value:

```
{\
  mchunkptr F = P->fd;\
  mchunkptr B = P->bk;\
  bindex_t I = small_index(S);\
  assert(P != B);\
  assert(P != F);\
  assert(chunksize(P) == small_index2size(I));\
  if (RTCHECK(F == smallbin_at(M,I) || (ok_address(M, F) && F->bk == P))) { \
    if (B == F) {\
      clear_smallmap(M, I);\
    }\
    else if (RTCHECK(B == smallbin_at(M,I) || \
```

```

        (ok_address(M, B) && B->fd == P)) {\
    F->bk = B;\
    B->fd = F;\
}\
else {\
    CORRUPTION_ERROR_ACTION(M);\
}\
}\
else {\
    CORRUPTION_ERROR_ACTION(M);\
}\
}

```

Definition at line 3591 of file dl_malloc.c.

7.17.1.188 USAGE_ERROR_ACTION

```

#define USAGE_ERROR_ACTION(
    m,
    p ) ABORT

```

Definition at line 2788 of file dl_malloc.c.

7.17.1.189 USE_BUILTIN_FFS

```

#define USE_BUILTIN_FFS 0

```

Definition at line 705 of file dl_malloc.c.

7.17.1.190 USE_DEV_RANDOM

```

#define USE_DEV_RANDOM 0

```

Definition at line 708 of file dl_malloc.c.

7.17.1.191 use_lock

```

#define use_lock(
    M ) ((M)->mflags & USE_LOCK_BIT)

```

Definition at line 2647 of file dl_malloc.c.

7.17.1.192 USE_LOCK_BIT

```
#define USE_LOCK_BIT (2U)
```

Definition at line 2038 of file dl_malloc.c.

7.17.1.193 USE_LOCKS

```
#define USE_LOCKS
```

Value:

```
((defined(USE_SPIN_LOCKS) && USE_SPIN_LOCKS != 0) || \
 (defined(USE_RECURSIVE_LOCKS) && USE_RECURSIVE_LOCKS != 0))
```

Definition at line 591 of file dl_malloc.c.

7.17.1.194 use_mmap

```
#define use_mmap(  
    M ) ((M)->mflags & USE_MMMap_BIT)
```

Definition at line 2655 of file dl_malloc.c.

7.17.1.195 USE_MMMap_BIT

```
#define USE_MMMap_BIT (SIZE_T_ONE)
```

Define CALL_MMMap/CALL_MUNMMMap/CALL_DIRECT_MMMap

Definition at line 1729 of file dl_malloc.c.

7.17.1.196 use_noncontiguous

```
#define use_noncontiguous(  
    M ) ((M)->mflags & USE_NONCONTIGUOUS_BIT)
```

Definition at line 2663 of file dl_malloc.c.

7.17.1.197 USE_NONCONTIGUOUS_BIT

```
#define USE_NONCONTIGUOUS_BIT (4U)
```

Definition at line 1771 of file dl_malloc.c.

7.17.1.198 USE_SPIN_LOCKS

```
#define USE_SPIN_LOCKS 0
```

Definition at line 607 of file dl_malloc.c.

7.17.2 Typedef Documentation

7.17.2.1 bindex_t

```
typedef unsigned int bindex_t
```

Definition at line 2197 of file dl_malloc.c.

7.17.2.2 binmap_t

```
typedef unsigned int binmap_t
```

Definition at line 2198 of file dl_malloc.c.

7.17.2.3 flag_t

```
typedef unsigned int flag_t
```

Definition at line 2199 of file dl_malloc.c.

7.17.2.4 mchunk

```
typedef struct malloc_chunk mchunk
```

Definition at line 2194 of file dl_malloc.c.

7.17.2.5 mchunkptr

```
typedef struct malloc_chunk* mchunkptr
```

Definition at line 2195 of file dl_malloc.c.

7.17.2.6 msegment

```
typedef struct malloc_segment msegment
```

Definition at line 2482 of file dl_malloc.c.

7.17.2.7 msegmentptr

```
typedef struct malloc_segment* msegmentptr
```

Definition at line 2483 of file dl_malloc.c.

7.17.2.8 mspace

```
typedef void* mspace
```

Definition at line 1275 of file dl_malloc.c.

7.17.2.9 mstate

```
typedef struct malloc_state* mstate
```

Definition at line 2607 of file dl_malloc.c.

7.17.2.10 sbinptr

```
typedef struct malloc_chunk* sbinptr
```

Definition at line 2196 of file dl_malloc.c.

7.17.2.11 tbinptr

```
typedef struct malloc_tree_chunk* tbinptr
```

Definition at line 2410 of file dl_malloc.c.

7.17.2.12 tchunk

```
typedef struct malloc_tree_chunk tchunk
```

Definition at line 2408 of file dl_malloc.c.

7.17.2.13 tchunkptr

```
typedef struct malloc_tree_chunk* tchunkptr
```

Definition at line 2409 of file dl_malloc.c.

7.17.3 Function Documentation

7.17.3.1 create_mspace()

```
mspace create_mspace (  
    size_t capacity,  
    int locked )
```

Definition at line 5424 of file dl_malloc.c.

7.17.3.2 create_mspace_with_base()

```
mspace create_mspace_with_base (  
    void * base,  
    size_t capacity,  
    int locked )
```

Definition at line 5443 of file dl_malloc.c.

7.17.3.3 destroy_mspace()

```
size_t destroy_mspace (
    mspace msp )
```

Definition at line 5474 of file dl_malloc.c.

7.17.3.4 mspace_bulk_free()

```
size_t mspace_bulk_free (
    mspace msp,
    void * array[],
    size_t nelem )
```

Definition at line 5846 of file dl_malloc.c.

7.17.3.5 mspace_calloc()

```
void * mspace_calloc (
    mspace msp,
    size_t n_elements,
    size_t elem_size )
```

Definition at line 5717 of file dl_malloc.c.

7.17.3.6 mspace_footprint()

```
size_t mspace_footprint (
    mspace msp )
```

Definition at line 5897 of file dl_malloc.c.

7.17.3.7 mspace_footprint_limit()

```
size_t mspace_footprint_limit (
    mspace msp )
```

Definition at line 5921 of file dl_malloc.c.

7.17.3.8 mspace_free()

```
void mspace_free (
    mspace msp,
    void * mem )
```

Definition at line 5616 of file dl_malloc.c.

7.17.3.9 mspace_independent_calloc()

```
void ** mspace_independent_calloc (
    mspace msp,
    size_t n_elements,
    size_t elem_size,
    void * chunks[] )
```

Definition at line 5825 of file dl_malloc.c.

7.17.3.10 mspace_independent_comalloc()

```
void ** mspace_independent_comalloc (
    mspace msp,
    size_t n_elements,
    size_t sizes[],
    void * chunks[] )
```

Definition at line 5836 of file dl_malloc.c.

7.17.3.11 mspace_mallinfo()

```
struct mallinfo mspace_mallinfo (
    mspace msp )
```

Definition at line 5953 of file dl_malloc.c.

7.17.3.12 mspace_malloc()

```
void * mspace_malloc (
    mspace msp,
    size_t bytes )
```

Definition at line 5502 of file dl_malloc.c.

7.17.3.13 mspace_malloc_stats()

```
void mspace_malloc_stats (  
    mspace msp )
```

Definition at line 5886 of file dl_malloc.c.

7.17.3.14 mspace_mallocopt()

```
int mspace_mallocopt (  
    int param_number,  
    int value )
```

Definition at line 5971 of file dl_malloc.c.

7.17.3.15 mspace_max_footprint()

```
size_t mspace_max_footprint (  
    mspace msp )
```

Definition at line 5909 of file dl_malloc.c.

7.17.3.16 mspace_memalign()

```
void * mspace_memalign (  
    mspace msp,  
    size_t alignment,  
    size_t bytes )
```

Definition at line 5814 of file dl_malloc.c.

7.17.3.17 mspace_realloc()

```
void * mspace_realloc (  
    mspace msp,  
    void * mem,  
    size_t newsize )
```

Definition at line 5737 of file dl_malloc.c.

7.17.3.18 `mSPACE_realloc_in_place()`

```
void* mSPACE_realloc_in_place (
    mSPACE msp,
    void * oldmem,
    size_t bytes )
```

Definition at line 5782 of file `dl_malloc.c`.

7.17.3.19 `mSPACE_set_footprint_limit()`

```
size_t mSPACE_set_footprint_limit (
    mSPACE msp,
    size_t bytes )
```

Definition at line 5934 of file `dl_malloc.c`.

7.17.3.20 `mSPACE_track_large_chunks()`

```
int mSPACE_track_large_chunks (
    mSPACE msp,
    int enable )
```

Definition at line 5457 of file `dl_malloc.c`.

7.17.3.21 `mSPACE_trim()`

```
int mSPACE_trim (
    mSPACE msp,
    size_t pad )
```

Definition at line 5870 of file `dl_malloc.c`.

7.17.3.22 `mSPACE_usable_size()`

```
size_t mSPACE_usable_size (
    const void * mem )
```

Definition at line 5962 of file `dl_malloc.c`.

7.18 dl_malloc.h File Reference

```
#include <stddef.h>
```

Classes

- struct [mallinfo](#)

Macros

- #define [ONLY_MSPACES](#) 1
- #define [ONLY_MSPACES](#) 1
- #define [NO_MALLINFO](#) 0
- #define [MSPACES](#) 1
- #define [MALLINFO_FIELD_TYPE](#) size_t
- #define [STRUCT_MALLINFO_DECLARED](#) 1

Typedefs

- typedef void * [mspace](#)

Functions

- size_t [dlmalloc_usable_size](#) (const void *)
- [mspace create_mspace](#) (size_t capacity, int locked)
- size_t [destroy_mspace](#) ([mspace](#) msp)
- [mspace create_mspace_with_base](#) (void *base, size_t capacity, int locked)
- int [mspace_track_large_chunks](#) ([mspace](#) msp, int enable)
- struct [mallinfo](#) [mspace_mallinfo](#) ([mspace](#) msp)
- int [mspace_mallopt](#) (int, int)
- void * [mspace_malloc](#) ([mspace](#) msp, size_t bytes)
- void [mspace_free](#) ([mspace](#) msp, void *mem)
- void * [mspace_calloc](#) ([mspace](#) msp, size_t n_elements, size_t elem_size)
- void * [mspace_realloc](#) ([mspace](#) msp, void *mem, size_t newsize)
- void * [mspace_realloc_in_place](#) ([mspace](#) msp, void *mem, size_t newsize)
- void * [mspace_memalign](#) ([mspace](#) msp, size_t alignment, size_t bytes)
- void ** [mspace_independent_calloc](#) ([mspace](#) msp, size_t n_elements, size_t elem_size, void *chunks[])
- void ** [mspace_independent_comalloc](#) ([mspace](#) msp, size_t n_elements, size_t sizes[], void *chunks[])
- size_t [mspace_bulk_free](#) ([mspace](#) msp, void **, size_t n_elements)
- size_t [mspace_usable_size](#) (const void *mem)
- void [mspace_malloc_stats](#) ([mspace](#) msp)
- int [mspace_trim](#) ([mspace](#) msp, size_t pad)
- size_t [mspace_footprint](#) ([mspace](#) msp)
- size_t [mspace_max_footprint](#) ([mspace](#) msp)
- size_t [mspace_footprint_limit](#) ([mspace](#) msp)
- size_t [mspace_set_footprint_limit](#) ([mspace](#) msp, size_t bytes)
- void [mspace_inspect_all](#) ([mspace](#) msp, void(*handler)(void *, void *, size_t, void *), void *arg)

7.18.1 Macro Definition Documentation

7.18.1.1 MALLINFO_FIELD_TYPE

```
#define MALLINFO_FIELD_TYPE size_t
```

Definition at line 57 of file dl_malloc.h.

7.18.1.2 MSPACES

```
#define MSPACES 1
```

Definition at line 46 of file dl_malloc.h.

7.18.1.3 NO_MALLINFO

```
#define NO_MALLINFO 0
```

Definition at line 41 of file dl_malloc.h.

7.18.1.4 ONLY_MSPACES [1/2]

```
#define ONLY_MSPACES 1
```

Definition at line 38 of file dl_malloc.h.

7.18.1.5 ONLY_MSPACES [2/2]

```
#define ONLY_MSPACES 1
```

Definition at line 38 of file dl_malloc.h.

7.18.1.6 STRUCT_MALLINFO_DECLARED

```
#define STRUCT_MALLINFO_DECLARED 1
```

Definition at line 60 of file dl_malloc.h.

7.18.2 Typedef Documentation

7.18.2.1 mspace

```
typedef void* mspace
```

Definition at line 533 of file dl_malloc.h.

7.18.3 Function Documentation

7.18.3.1 create_mspace()

```
mspace create_mspace (  
    size_t capacity,  
    int locked )
```

Definition at line 5424 of file dl_malloc.c.

7.18.3.2 create_mspace_with_base()

```
mspace create_mspace_with_base (  
    void * base,  
    size_t capacity,  
    int locked )
```

Definition at line 5443 of file dl_malloc.c.

7.18.3.3 destroy_mspace()

```
size_t destroy_mspace (  
    mspace msp )
```

Definition at line 5474 of file dl_malloc.c.

7.18.3.4 dmalloc_usable_size()

```
size_t dmalloc_usable_size (
    const void * )
```

7.18.3.5 mspace_bulk_free()

```
size_t mspace_bulk_free (
    mspace msp,
    void ** ,
    size_t n_elements )
```

7.18.3.6 mspace_calloc()

```
void* mspace_calloc (
    mspace msp,
    size_t n_elements,
    size_t elem_size )
```

Definition at line 5717 of file dl_malloc.c.

7.18.3.7 mspace_footprint()

```
size_t mspace_footprint (
    mspace msp )
```

Definition at line 5897 of file dl_malloc.c.

7.18.3.8 mspace_footprint_limit()

```
size_t mspace_footprint_limit (
    mspace msp )
```

Definition at line 5921 of file dl_malloc.c.

7.18.3.9 mspace_free()

```
void mspace_free (
    mspace msp,
    void * mem )
```

Definition at line 5616 of file dl_malloc.c.

7.18.3.10 mspace_independent_calloc()

```
void** mspace_independent_calloc (
    mspace msp,
    size_t n_elements,
    size_t elem_size,
    void * chunks[] )
```

Definition at line 5825 of file dl_malloc.c.

7.18.3.11 mspace_independent_comalloc()

```
void** mspace_independent_comalloc (
    mspace msp,
    size_t n_elements,
    size_t sizes[],
    void * chunks[] )
```

Definition at line 5836 of file dl_malloc.c.

7.18.3.12 mspace_inspect_all()

```
void mspace_inspect_all (
    mspace msp,
    void(*) (void *, void *, size_t, void *) handler,
    void * arg )
```

7.18.3.13 mspace_mallinfo()

```
struct mallinfo mspace_mallinfo (
    mspace msp )
```

Definition at line 5953 of file dl_malloc.c.

7.18.3.14 mspace_malloc()

```
void* mspace_malloc (
    mspace msp,
    size_t bytes )
```

Definition at line 5502 of file dl_malloc.c.

7.18.3.15 mspace_malloc_stats()

```
void mspace_malloc_stats (
    mspace msp )
```

Definition at line 5886 of file dl_malloc.c.

7.18.3.16 mspace_mallopt()

```
int mspace_mallopt (
    int ,
    int )
```

Definition at line 5971 of file dl_malloc.c.

7.18.3.17 mspace_max_footprint()

```
size_t mspace_max_footprint (
    mspace msp )
```

Definition at line 5909 of file dl_malloc.c.

7.18.3.18 mspace_memalign()

```
void* mspace_memalign (
    mspace msp,
    size_t alignment,
    size_t bytes )
```

Definition at line 5814 of file dl_malloc.c.

7.18.3.19 mspace_realloc()

```
void* mspace_realloc (
    mspace msp,
    void * mem,
    size_t newsize )
```

Definition at line 5737 of file dl_malloc.c.

7.18.3.20 mspace_realloc_in_place()

```
void* mspace_realloc_in_place (
    mspace msp,
    void * mem,
    size_t newsize )
```

Definition at line 5782 of file dl_malloc.c.

7.18.3.21 mspace_set_footprint_limit()

```
size_t mspace_set_footprint_limit (
    mspace msp,
    size_t bytes )
```

Definition at line 5934 of file dl_malloc.c.

7.18.3.22 mspace_track_large_chunks()

```
int mspace_track_large_chunks (
    mspace msp,
    int enable )
```

Definition at line 5457 of file dl_malloc.c.

7.18.3.23 mspace_trim()

```
int mspace_trim (
    mspace msp,
    size_t pad )
```

Definition at line 5870 of file dl_malloc.c.

7.18.3.24 mspace_usable_size()

```
size_t mspace_usable_size (
    const void * mem )
```

Definition at line 5962 of file dl_malloc.c.

7.19 future.hpp File Reference

```
#include <upcxx/future/core.hpp>
#include <upcxx/future/future1.hpp>
#include <upcxx/future/apply.hpp>
#include <upcxx/future/make_future.hpp>
#include <upcxx/future/promise.hpp>
#include <upcxx/future/then.hpp>
#include <upcxx/future/when_all.hpp>
```

7.20 future/apply.hpp File Reference

```
#include <upcxx/future/core.hpp>
#include <upcxx/future/make_future.hpp>
```

Classes

- struct [upcxx::detail::apply_tupled_as_future_impl< Return >](#)
- struct [upcxx::detail::apply_tupled_as_future_impl< void >](#)
- struct [upcxx::detail::apply_tupled_as_future_impl< Return >](#)
- struct [upcxx::detail::apply_tupled_as_future_impl< future1< Kind, T... > >](#)
- struct [upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, ArgTupDecayed >](#)
- struct [upcxx::detail::apply_tupled_as_future_help< Fn, ArgTup, std::tuple< T... > >](#)
- struct [upcxx::detail::apply_tupled_as_future< Fn, ArgTup >](#)
- struct [upcxx::detail::apply_futured_as_future< Fn\(future1< Kind, T... >\)>](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- template<typename Fn , typename ArgTup >
auto [upcxx::apply_tupled_as_future](#) (Fn &&fn, ArgTup &&argtup) -> decltype(detail::apply_tupled_as_↵
future< Fn, ArgTup >)(std::forward< Fn >(fn), std::forward< ArgTup >(argtup)))

7.21 future/body_pure.hpp File Reference

```
#include <upcxx/future/core.hpp>
```

Classes

- struct [upcxx::detail::future_body_pure](#)< future1 < Kind, T... > >

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

7.22 future/core.cpp File Reference

```
#include <upcxx/future/core.hpp>
```

Typedefs

- template<typename ... T>
using [future_header_result](#) = [upcxx::detail::future_header_result](#)< T... >

7.22.1 Typedef Documentation

7.22.1.1 future_header_result

```
template<typename ... T>  
using future\_header\_result = upcxx::detail::future\_header\_result<T...>
```

Definition at line 11 of file core.cpp.

7.23 future/core.hpp File Reference

```
#include <upcxx/diagnostic.hpp>  
#include <upcxx/utility.hpp>
```

Classes

- struct [upcxx::detail::future_header_result< T >](#)
- struct [upcxx::detail::future_dependency< FuArg >](#)
- struct [upcxx::detail::future_body_proxy< T >](#)
- struct [upcxx::detail::future_body_pure< FuArg >](#)
- struct [upcxx::detail::future_body_then< FuArg, Fn >](#)
- struct [upcxx::detail::future_impl_shref< HeaderOps, T >](#)
- struct [upcxx::detail::future_impl_result< T >](#)
- struct [upcxx::detail::future_impl_when_all< ArgTuple, T >](#)
- struct [upcxx::detail::future_impl_mapped< FuArg, Fn, T >](#)
- struct [upcxx::detail::future_kind_shref< HeaderOps >](#)
- struct [upcxx::detail::future_kind_result](#)
- struct [upcxx::detail::future_kind_when_all< FuArg >](#)
- struct [upcxx::detail::future_kind_mapped< FuArg, Fn >](#)
- struct [upcxx::future1< Kind, T >](#)
- struct [upcxx::future_is_trivially_ready< FutureOrKind >](#)
- struct [upcxx::detail::future_from_tuple< Kind, Tup >](#)
- struct [upcxx::detail::future_from_tuple< Kind, std::tuple< T... > >](#)
- struct [upcxx::detail::future_then< Arg, Fn, FnRet, arg_trivial >](#)
- struct [upcxx::detail::future_then_pure< Arg, Fn, FnRet, arg_trivial, fnret_trivial >](#)
- struct [upcxx::detail::future_header](#)
- struct [upcxx::detail::future_header::dependency_link](#)
- struct [upcxx::detail::future_header_dependent](#)
- struct [upcxx::detail::future_header_ops_general](#)
- struct [upcxx::detail::future_header_ops_result](#)
- struct [upcxx::detail::future_header_ops_result_ready](#)
- struct [upcxx::detail::future_body](#)
- struct [upcxx::detail::future_body_proxy_](#)
- struct [upcxx::detail::future_body_proxy< T >](#)
- struct [upcxx::detail::future_header_result< T >](#)
- struct [upcxx::detail::future_header_result<>](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Typedefs

- `template<typename ... T>`
`using upcxx::future = future1< detail::future_kind_shref< detail::future_header_ops_general >, T... >`
- `template<typename App >`
`using upcxx::detail::apply_futred_as_future_return_t = typename apply_futred_as_future< App >↔
::return_type`
- `template<typename Kind, typename Tup >`
`using upcxx::detail::future_from_tuple_t = typename future_from_tuple< Kind, Tup >::type`

7.24 future/future1.hpp File Reference

```
#include <upcxx/future/core.hpp>
```


Classes

- struct [upcxx::future1](#)< Kind, T >

Namespaces

- [upcxx](#)

7.25 future/impl_mapped.hpp File Reference

```
#include <upcxx/future/core.hpp>
#include <upcxx/future/apply.hpp>
#include <upcxx/future/body_pure.hpp>
```

Classes

- struct [upcxx::future_is_trivially_ready](#)< future1< detail::future_kind_mapped< Arg, Fn >, T... > >
- struct [upcxx::detail::future_impl_mapped](#)< FuArg, Fn, T >
- struct [upcxx::detail::future_impl_mapped](#)< FuArg, Fn, T >::result_lrefs_function
- struct [upcxx::detail::future_dependency](#)< future1< future_kind_mapped< FuArg, Fn >, T... > >

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

7.26 future/impl_result.hpp File Reference

```
#include <upcxx/future/core.hpp>
```

Classes

- struct [upcxx::future_is_trivially_ready](#)< future1< detail::future_kind_result, T... > >
- struct [upcxx::detail::future_impl_result](#)< T >
- struct [upcxx::detail::future_impl_result](#)<>
- struct [upcxx::detail::future_dependency](#)< future1< future_kind_result, T... > >
- struct [upcxx::detail::future_dependency](#)< future1< future_kind_result > >

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

7.27 future/impl_shref.hpp File Reference

```
#include <upcxx/future/core.hpp>
```

Classes

- struct [upcxx::future_is_trivially_ready](#)< future1< detail::future_kind_shref< HeaderOps >, T... > >
- struct [upcxx::detail::future_impl_shref](#)< HeaderOps, T >
- struct [upcxx::detail::future_dependency_shref_base](#)< is_trivially_ready_result >
- struct [upcxx::detail::future_dependency_shref_base](#)< false >
- struct [upcxx::detail::future_dependency_shref_base](#)< true >
- struct [upcxx::detail::future_dependency](#)< future1< future_kind_shref< HeaderOps >, T... > >

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

7.28 future/impl_when_all.hpp File Reference

```
#include <upcxx/future/core.hpp>
#include <upcxx/future/body_pure.hpp>
```

Classes

- struct [upcxx::future_is_trivially_ready](#)< future1< detail::future_kind_when_all< Arg... >, T... > >
- struct [upcxx::detail::future_impl_when_all](#)< ArgTuple, T >
- struct [upcxx::detail::future_impl_when_all](#)< std::tuple< FuArg... >, T... >
- struct [upcxx::detail::future_dependency_when_all_arg](#)< i, Arg >
- struct [upcxx::detail::future_dependency_when_all_base](#)< AllArg, lxFSeq >
- struct [upcxx::detail::future_dependency_when_all_base](#)< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... > >
- struct [upcxx::detail::future_dependency](#)< future1< future_kind_when_all< Arg... >, T... > >

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

7.29 future/make_future.hpp File Reference

```
#include <upcxx/future/impl_result.hpp>
#include <upcxx/future/impl_shref.hpp>
```

Classes

- struct [upcxx::detail::make_future_< trivial, T >](#)
- struct [upcxx::detail::make_future_< true, T... >](#)
- struct [upcxx::detail::make_future_< false, T... >](#)
- struct [upcxx::detail::make_future< T >](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- template<typename ... T>
auto [upcxx::make_future](#) (T ...values) -> decltype(detail::make_future< T... >()(std::declval< T >()...))
- template<typename T >
auto [upcxx::to_future](#) (T x) -> decltype(make_future(std::forward< T >(x)))
- template<typename Kind, typename ... T>
future1< Kind, T... > [upcxx::to_future](#) (future1< Kind, T... > x)

7.30 future/promise.hpp File Reference

```
#include <upcxx/future/core.hpp>
#include <upcxx/future/impl_shref.hpp>
#include <cstdint>
```

Classes

- class [upcxx::promise< T >](#)
- struct [upcxx::detail::promise_like< Fu >](#)
- struct [upcxx::detail::promise_like< future1< Kind, T... > >](#)
- class [upcxx::promise< T >](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Typedefs

- template<typename Fu >
using [upcxx::detail::promise_like_t](#) = typename promise_like< Fu >::type

7.31 future/then.hpp File Reference

```
#include <upcxx/future/core.hpp>
#include <upcxx/future/apply.hpp>
#include <upcxx/future/impl_mapped.hpp>
```

Classes

- struct [upcxx::detail::future_body_then_base](#)
- struct [upcxx::detail::future_body_then< FuArg, Fn >](#)
- struct [upcxx::detail::future_body_then_pure< FuArg, Fn >](#)
- struct [upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, false >](#)
- struct [upcxx::detail::future_then< Arg, Fn, future1< FnRetKind, FnRetT... >, true >](#)
- struct [upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, false >](#)
- struct [upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, true, fnret_trivial >](#)
- struct [upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, true >](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- `template<typename ArgKind , typename Fn1 , typename ... ArgT>`
`auto upcxx::operator>> (future1< ArgKind, ArgT... > arg, Fn1 &&fn) -> decltype(detail::future_then<`
`future1< ArgKind, ArgT... >, typename std::decay< Fn1 >::type >()(std::move(arg), std::forward< Fn1`
`>(fn)))`
- `template<typename Fn1 , typename ... ArgT>`
`future< ArgT... > & upcxx::operator>>= (future< ArgT... > &arg, Fn1 &&fn)`

7.32 future/when_all.hpp File Reference

```
#include <upcxx/future/core.hpp>
#include <upcxx/future/impl_when_all.hpp>
```

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Typedefs

- `template<typename ... Arg>`
`using upcxx::detail::when_all_return_t = future_from_tuple_t< future_kind_when_all< Arg... >`
`, decltype(std::tuple_cat(std::declval< typename Arg::results_type >()...)) >`

Functions

- `template<typename ... Arg>`
`detail::when_all_return_t< Arg... >` [upcxx::when_all](#) (Arg ...args)

7.33 global_ptr.hpp File Reference

```
#include <upcxx/backend.hpp>
#include <upcxx/diagnostic.hpp>
#include <cassert>
#include <cstddef>
#include <cstdint>
#include <iostream>
#include <type_traits>
```

Classes

- struct [upcxx::detail::global_ptr_ctor_internal](#)
- class [upcxx::global_ptr< T >](#)
- struct [std::less< upcxx::global_ptr< T > >](#)
- struct [std::less_equal< upcxx::global_ptr< T > >](#)
- struct [std::greater< upcxx::global_ptr< T > >](#)
- struct [std::greater_equal< upcxx::global_ptr< T > >](#)
- struct [std::hash< upcxx::global_ptr< T > >](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)
- [std](#)

Functions

- bool [upcxx::is_memory_shared_with](#) (inrank_t r)
- void * [upcxx::pshm_remote_addr2local](#) (inrank_t r, void *addr)
- void * [upcxx::pshm_local_addr2remote](#) (void *addr, inrank_t &rank_out)
- `template<typename T >`
`global_ptr< T >` [upcxx::operator+](#) (std::ptrdiff_t diff, `global_ptr< T >` ptr)
- `template<typename T, typename U >`
`global_ptr< T >` [upcxx::reinterpret_pointer_cast](#) (`global_ptr< U >` ptr)
- `template<typename T >`
`std::ostream &` [upcxx::operator<<](#) (std::ostream &os, `global_ptr< T >` ptr)

7.34 nobsrule.py File Reference

Namespaces

- [nobsrule](#)

Functions

- def [nobsrule.requires_upcxx_backend](#) (cxt, src)

7.35 os_env.hpp File Reference

```
#include <upcxx/diagnostic.hpp>
#include <cstdlib>
#include <iostream>
#include <sstream>
#include <string>
```

Classes

- struct [upcxx::detail::os_env_parse< T >](#)
- struct [upcxx::detail::os_env_parse< std::string >](#)
- struct [upcxx::detail::os_env_parse< T >](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- template<class T >
T [upcxx::os_env](#) (const std::string &name)
- template<class T >
T [upcxx::os_env](#) (const std::string &name, const T &otherwise)

7.36 packing.cpp File Reference

```
#include <upcxx/packing.hpp>
```

7.37 packing.hpp File Reference

```
#include <upcxx/diagnostic.hpp>
#include <upcxx/reflection.hpp>
#include <upcxx/utility.hpp>
#include <array>
#include <cstdint>
#include <stdint>
#include <cstring>
#include <string>
#include <tuple>
#include <type_traits>
#include <unordered_map>
#include <unordered_set>
#include <utility>
#include <vector>
```

Classes

- class [upcxx::parcel_layout](#)
- struct [upcxx::parcel_writer](#)
- class [upcxx::parcel_reader](#)
- struct [upcxx::packing_not_supported< T >](#)
- struct [upcxx::packing_is_trivial< T, false_ >](#)
- struct [upcxx::packing_is_trivial< T, std::integral_constant< bool, false &packing< T >::is_trivial > >](#)
- struct [upcxx::packing_empty< T, is_default_constructible >](#)
- struct [upcxx::packing_empty< T, true >](#)
- struct [upcxx::packing_empty< T, false >](#)
- struct [upcxx::packing_trivial< T >](#)
- struct [upcxx::packing_opaque< T, is_empty, is_trivially_copyable >](#)
- struct [upcxx::packing_opaque< T, true, is_trivially_copyable >](#)
- struct [upcxx::packing_opaque< T, false, true >](#)
- struct [upcxx::packing_opaque< T, false, false >](#)
- struct [upcxx::packing_ubound_reflector](#)
- struct [upcxx::packing_pack_reflector](#)
- struct [upcxx::packing_unpack_reflector< member_assignment_not_construction >](#)
- struct [upcxx::packing_unpack_reflector< true >](#)
- struct [upcxx::packing_unpack_reflector< false >](#)
- struct [upcxx::packing_reflected< T, is_default_constructible >](#)
- struct [upcxx::packing_reflected< T, true >](#)
- struct [upcxx::packing_reflected< T, false >](#)
- struct [upcxx::packing_function_pointer< T >](#)
- struct [upcxx::packing_specializer< T, is_empty, is_funptr, is_scalar >](#)
- struct [upcxx::packing_specializer< T, true, is_funptr, is_scalar >](#)
- struct [upcxx::packing_specializer< T, false, true, is_scalar >](#)
- struct [upcxx::packing_specializer< T, false, false, true >](#)
- struct [upcxx::packing_specializer< T, false, false, false >](#)
- struct [upcxx::packing< T >](#)
- struct [upcxx::packing< T const >](#)
- struct [upcxx::packing< T volatile >](#)
- struct [upcxx::packing< T & >](#)
- struct [upcxx::packing< T && >](#)
- struct [upcxx::detail::packing_tuple_each< n, i, T >](#)
- struct [upcxx::detail::packing_tuple_each< n, n, T... >](#)
- struct [upcxx::packing< std::tuple< T... > >](#)
- struct [upcxx::packing< std::pair< A, B > >](#)
- struct [upcxx::packing< std::array< T, n > >](#)
- struct [upcxx::packing< std::string >](#)
- struct [upcxx::packing< std::vector< T > >](#)
- struct [upcxx::packing< std::unordered_set< T > >](#)
- struct [upcxx::packing< std::unordered_map< K, V > >](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- void [upcxx::detail::packing_funptr_basis \(\)](#)

7.38 reflection.hpp File Reference

```
#include <array>
#include <cstdint>
#include <utility>
#include <iostream>
#include <tuple>
#include <type_traits>
```

Classes

- struct [upcxx::reflection< Subject >](#)
- struct [upcxx::fast_hasher](#)
- struct [upcxx::hasher_reflector< Hasher >](#)
- struct [upcxx::fast_hashing< T >](#)
- struct [upcxx::integer_golden_ratio_bits< bit_n >](#)
- struct [upcxx::integer_golden_ratio_bits< 32 >](#)
- struct [upcxx::integer_golden_ratio_bits< 64 >](#)
- struct [upcxx::integer_golden_ratio< T >](#)
- struct [upcxx::mod2n_hashing< T >](#)
- struct [upcxx::reflection_tuple< Tup, i, n >](#)
- struct [upcxx::reflection_tuple< Tup, n, n >](#)
- struct [upcxx::reflection< std::tuple< Ts... > >](#)
- struct [upcxx::reflection< std::pair< A, B > >](#)
- struct [upcxx::reflection< std::array< T, n > >](#)
- struct [upcxx::print_reflector](#)
- struct [upcxx::print_proxy< T >](#)

Namespaces

- [upcxx](#)

Functions

- template<typename Reflector , typename Subject >
void [reflect](#) (Reflector &re, Subject &&x)
- template<typename Reflector , typename Subject >
void [upcxx::reflect_upon](#) (Reflector &re, Subject &&subj)
- template<typename T >
std::size_t [upcxx::fast_hash](#) (const T &x, std::size_t salt=0)
- template<typename T >
std::size_t [upcxx::mod2n_hash](#) (const T &x, int bits)
- template<typename T >
print_proxy< T > [upcxx::print](#) (T const &subject)

7.38.1 Function Documentation

7.38.1.1 reflect()

```
template<typename Reflector , typename Subject >
void reflect (
    Reflector & re,
    Subject && x ) [inline]
```

Definition at line 33 of file reflection.hpp.

7.39 rget.hpp File Reference

```
#include <upcxx/backend.hpp>
#include <upcxx/completion.hpp>
#include <upcxx/global_ptr.hpp>
```

Classes

- struct [upcxx::detail::rget_byval< T >](#)
- struct [upcxx::detail::rget_state_operxn< Mode, Cx >](#)
- struct [upcxx::detail::rget_state_remote< Cx >](#)
- struct [upcxx::detail::rget_state_operxn< rget_byref, nil_cx >](#)
- struct [upcxx::detail::rget_state_operxn< rget_byval< T >, nil_cx >](#)
- struct [upcxx::detail::rget_state_operxn< rget_byref, future_cx< ordinal > >](#)
- struct [upcxx::detail::rget_state_operxn< rget_byval< T >, future_cx< ordinal > >](#)
- struct [upcxx::detail::rget_state_operxn< rget_byref, promise_cx<> >](#)
- struct [upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T > >](#)
- struct [upcxx::detail::rget_state_remote< nil_cx >](#)
- struct [upcxx::detail::rget_state_remote< rpc_cx< Fn > >](#)
- struct [upcxx::detail::rget_states< Mode, R, O >](#)
- struct [upcxx::detail::rget_futures_of< Mode, R, O >](#)
- struct [upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > >](#)
- struct [upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > >](#)

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- template<typename T , typename R = nil_cx, typename O = future_cx<0>>
[detail::rget_futures_of< detail::rget_byval< T >, R, O >::return_type upcxx::rget](#) (global_ptr< T > gp_s,
 completions< nil_cx, R, O > cxs=completions< nil_cx, R, O >{ })
- template<typename T , typename R = nil_cx, typename O = future_cx<0>>
[detail::rget_futures_of< detail::rget_byref, R, O >::return_type upcxx::rget](#) (global_ptr< T > gp_s, T *buf_d,
 std::size_t n, completions< nil_cx, R, O > cxs=completions< nil_cx, R, O >{ })

7.40 rpc.hpp File Reference

```
#include <upcxx/backend.hpp>
#include <upcxx/bind.hpp>
#include <upcxx/future.hpp>
```

Classes

- struct `upcxx::detail::rpc_recipient_after< Pro >`
- struct `upcxx::detail::rpc_return< ValidType, Fn, Args >`

Namespaces

- `upcxx`
- `upcxx::detail`

Functions

- template<typename Fn , typename ... Args>
void `upcxx::rpc_ff` (inrank_t recipient, Fn &&fn, Args &&...args)
- template<typename Fn , typename ... Args>
auto `upcxx::rpc` (inrank_t recipient, Fn &&fn, Args &&...args) -> typename detail::rpc_return< decltype(fn(args...)), Fn, Args... >::type
- template<typename Fn , typename ... T, typename ... Args>
void `upcxx::rpc` (inrank_t recipient, promise< T... > &prom, Fn &&fn, Args &&...args)

7.41 rput.hpp File Reference

```
#include <upcxx/backend.hpp>
#include <upcxx/completion.hpp>
#include <upcxx/global_ptr.hpp>
```

Classes

- struct `upcxx::detail::rput_byval< T >`
- struct `upcxx::detail::rput_state_here< Cx >`
- struct `upcxx::detail::rput_state_remote< Cx >`
- struct `upcxx::detail::rput_state_here< nil_cx >`
- struct `upcxx::detail::rput_state_here< future_cx< ordinal > >`
- struct `upcxx::detail::rput_state_here< promise_cx<> >`
- struct `upcxx::detail::rput_state_remote< nil_cx >`
- struct `upcxx::detail::rput_state_remote< rpc_cx< Fn > >`
- struct `upcxx::detail::rput_states< Mode, S, R, O >`
- struct `upcxx::detail::rput_states< rput_byref, S, R, O >`
- struct `upcxx::detail::rput_states< rput_byval< T >, S, R, O >`
- struct `upcxx::detail::rput_futures_of< S, R, O >`
- struct `upcxx::detail::rput_futures_of< future_cx< S_ord >, R, O >`
- struct `upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > >`
- struct `upcxx::detail::rput_futures_of< future_cx< S_ord >, R, future_cx< O_ord > >`

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Functions

- `template<typename T, typename S = nil_cx, typename R = nil_cx, typename O = future_cx<0>>
detail::rput_futures_of< S, R, O >::return_type upcxx::rput (T value_s, global_ptr< T > gp_d, completions< S, R, O > cxs=completions< S, R, O >{})`
- `template<typename T, typename S = nil_cx, typename R = nil_cx, typename O = future_cx<0>>
detail::rput_futures_of< S, R, O >::return_type upcxx::rput (T const *buf_s, std::size_t n, global_ptr< T > gp_d, completions< S, R, O > cxs=completions< S, R, O >{})`

7.42 upcxx.cpp File Reference

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

7.43 upcxx.hpp File Reference

```
#include <upcxx/allocate.hpp>  
#include <upcxx/backend.hpp>  
#include <upcxx/dist_object.hpp>  
#include <upcxx/future.hpp>  
#include <upcxx/global_ptr.hpp>  
#include <upcxx/rget.hpp>  
#include <upcxx/rput.hpp>  
#include <upcxx/wait.hpp>  
#include <upcxx/atomic.hpp>  
#include <upcxx/broadcast.hpp>  
#include <upcxx/allreduce.hpp>
```

7.44 utility.hpp File Reference

```
#include <upcxx/diagnostic.hpp>  
#include <array>  
#include <cstdint>  
#include <functional>  
#include <iostream>  
#include <sstream>  
#include <tuple>  
#include <utility>
```

Classes

- struct `upcxx::nop_function< Sig >`
- struct `upcxx::nop_function< Ret(Arg...)>`
- struct `upcxx::nop_function< void(Arg...)>`
- struct `upcxx::constant_function< T >`
- class `upcxx::function_ref< Sig >`
- class `upcxx::function_ref< Ret(Arg...)>`
- struct `upcxx::trait_forall< Test, T >`
- struct `upcxx::trait_forall< Test >`
- struct `upcxx::trait_forall< Test, T, Ts... >`
- struct `upcxx::trait_forall_tupled< Test, Tuple >`
- struct `upcxx::trait_forall_tupled< Test, std::tuple< T... > >`
- struct `upcxx::trait_any< Tr >`
- struct `upcxx::trait_any<>`
- struct `upcxx::trait_any< Tr0, Trs... >`
- struct `upcxx::trait_any< Tr0, Trs... >::type< T >`
- struct `upcxx::trait_all< Tr >`
- struct `upcxx::trait_all<>`
- struct `upcxx::trait_all< Tr0, Trs... >`
- struct `upcxx::trait_all< Tr0, Trs... >::type< T >`
- struct `upcxx::tuple_types_into< Tuple, Into >`
- struct `upcxx::tuple_types_into< std::tuple< T... >, Into >`
- struct `upcxx::index_sequence<... >`
- struct `upcxx::detail::make_index_sequence< n, s >`
- struct `upcxx::detail::make_index_sequence< 0, s... >`
- struct `upcxx::add_lref_if_nonref< T >`
- struct `upcxx::add_lref_if_nonref< T & >`
- struct `upcxx::add_lref_if_nonref< T && >`
- struct `upcxx::decay_tupled< Tup >`
- struct `upcxx::decay_tupled< std::tuple< T... > >`
- struct `upcxx::detail::tuple_get_or_void< i, TupRef, in_range >`
- struct `upcxx::detail::tuple_get_or_void< i, TupRef, false >`
- struct `upcxx::detail::tuple_rvals_get< i, Tup >`
- struct `upcxx::detail::tuple_rvals_get< Tup &, i, Ti & >`
- struct `upcxx::detail::tuple_rvals_get< Tup &, i, Ti && >`
- struct `upcxx::detail::tuple_rvals_get< Tup &, i, Ti >`
- struct `upcxx::detail::tuple_rvals_get< Tup const &, i, Ti & >`
- struct `upcxx::detail::tuple_rvals_get< Tup const &, i, Ti && >`
- struct `upcxx::detail::tuple_rvals_get< Tup const &, i, Ti >`
- struct `upcxx::detail::tuple_rvals_get< Tup &&, i, Ti & >`
- struct `upcxx::detail::tuple_rvals_get< Tup &&, i, Ti && >`
- struct `upcxx::detail::tuple_rvals_get< Tup &&, i, Ti >`

Namespaces

- [upcxx](#)
- [upcxx::detail](#)

Typedefs

- `template<typename Tuple, template< typename... > class Into>`
`using upcxx::tuple_types_into_t = typename tuple_types_into< Tuple, Into >::type`
- `template<int n>`
`using upcxx::make_index_sequence = typename detail::make_index_sequence< n >::type`

Functions

- `template<typename Sig >`
`nop_function< Sig > upcxx::nop ()`
- `template<typename T >`
`constant_function< T > upcxx::constant (T value)`
- `template<int i, typename Tup >`
`auto upcxx::get_or_void (Tup &&tup) -> decltype(detail::tuple_get_or_void< i, Tup >()(std::forward< Tup >(tup)))`
- `template<typename Tup , int ... i>`
`auto upcxx::detail::tuple_rvals (Tup &&tup, index_sequence< i... >) -> std::tuple< decltype(tuple_rvals_↵
get< Tup &&, i >()(tup))... >`
- `template<typename Tup >`
`auto upcxx::tuple_rvals (Tup &&tup) -> decltype(detail::tuple_rvals(std::forward< Tup >(tup), make_index_↵
_sequence< std::tuple_size< typename std::decay< Tup >::type >::value >()))`
- `template<typename Fn , typename Tup , int ... i>`
`auto upcxx::detail::apply_tupled (Fn &&fn, Tup &&args, index_sequence< i... >) -> decltype(fn(std::get< i
>(args)...))`
- `template<typename Fn , typename Tup >`
`auto upcxx::apply_tupled (Fn &&fn, Tup &&args) -> decltype(detail::apply_tupled(std::forward< Fn >(fn),
std::forward< Tup >(args), make_index_sequence< std::tuple_size< Tup >::value >()))`

7.45 wait.hpp File Reference

```
#include <upcxx/backend.hpp>
#include <upcxx/future.hpp>
```

Namespaces

- [upcxx](#)

Functions

- `template<typename Kind , typename ... T>`
`auto upcxx::wait (future1< Kind, T... > const &f) -> decltype(f.result())`

Index

`_STRUCT_MALLINFO`
 `dl_malloc.c`, [356](#)
`~dist_object`
 `upcxx::dist_object`, [98](#)
`~future1`
 `upcxx::future1`, [106](#)
`~future_impl_shref`
 `upcxx::detail::future_impl_shref`, [164](#)
`~parcel_reader`
 `upcxx::parcel_reader`, [259](#)

a
 `upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator`, [60](#)
`ABORT_ON_ASSERT_FAILURE`
 `dl_malloc.c`, [357](#)
`ABORT`
 `dl_malloc.c`, [356](#)
`ACQUIRE_LOCK`
 `dl_malloc.c`, [357](#)
`ACQUIRE_MALLOC_GLOBAL_LOCK`
 `dl_malloc.c`, [357](#)
`accum`
 `upcxx::detail::allreduce_state`, [58](#)
`action_impl`
 `upcxx::backend::gasnet1_seq::action_impl`, [54](#)
`active_next`
 `upcxx::detail::future_header_dependent`, [144](#)
`add_bytes`
 `upcxx::parcel_layout`, [256](#)
`add_trivial_aligned`
 `upcxx::parcel_layout`, [256](#)
`add_trivial_unaligned`
 `upcxx::parcel_layout`, [256](#)
`align_as_chunk`
 `dl_malloc.c`, [357](#)
`align_offset`
 `dl_malloc.c`, [357](#)
`alignment`
 `upcxx::parcel_layout`, [257](#)
 `upcxx::parcel_writer`, [262](#)
`allocate`
 `upcxx`, [24](#)
`allocate.hpp`, [339](#)
`allreduce`
 `upcxx`, [25](#)
`allreduce.hpp`, [340](#)
`am_size_rdzv_cutover`
 `upcxx::backend::gasnet1_seq`, [45](#)
`answer`
 `upcxx::detail::allreduce_state`, [59](#)
`apply_`
 `upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >`, [79](#)
 `upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator`, [60](#)
 `upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true >`, [81](#)
`apply_futured_as_future_return_t`
 `upcxx::detail`, [49](#)
`apply_tupled`
 `upcxx`, [25](#)
 `upcxx::detail`, [49](#)
`apply_tupled_as_future`
 `upcxx`, [25](#)
`arena`
 `mallinfo`, [203](#)
`arg_`
 `upcxx::detail::future_impl_mapped`, [157](#)
`args_`
 `upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >`, [169](#)
`assert`
 `dl_malloc.c`, [357](#)
`assert_failed`
 `upcxx`, [25](#)
`atomic.hpp`, [340](#)
`atomic_fetch_add`
 `upcxx`, [25](#)
`atomic_get`
 `upcxx`, [26](#)
`atomic_put`
 `upcxx`, [26](#)

b
 `upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator`, [61](#)
`b_`
 `upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >`, [80](#)
 `upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true >`, [81](#)
`backend.hpp`, [342](#)
 `gasnet1_seq`, [344](#)
`backend/gasnet1_seq/backend.cpp`, [341](#)
`backend/gasnet1_seq/backend.hpp`, [341](#)
`barrier`
 `upcxx`, [26](#)
`base`
 `malloc_segment`, [207](#)

- base_type
 - upcxx::bound_function, 77
- bind
 - upcxx, 26
- bind.hpp, 344
- bind_last
 - upcxx, 27
 - upcxx::detail, 50
- bindex_t
 - dl_malloc.c, 394
- binmap_t
 - dl_malloc.c, 394
- bit_for_tree_index
 - dl_malloc.c, 358
- bk
 - malloc_chunk, 205
 - malloc_tree_chunk, 212
- body_
 - upcxx::detail::future_header, 140
- bound_function
 - upcxx::bound_function, 78
- broadcast
 - upcxx, 27
 - upcxx::detail::allreduce_state, 58
- broadcast.hpp, 345
- broadcast_receive
 - upcxx::detail, 50
- buf_
 - upcxx::parcel_writer, 264
- buffer
 - upcxx::detail::rput_states< rput_byval< T >, S, R, O >, 322
 - upcxx::parcel_reader, 259
 - upcxx::parcel_writer, 262
- CALL_DIRECT_MMAP
 - dl_malloc.c, 358
- CALL_MMAP
 - dl_malloc.c, 358
- CALL_MORECORE
 - dl_malloc.c, 358
- CALL_MREMAP
 - dl_malloc.c, 358
- CALL_MUNMAP
 - dl_malloc.c, 359
- CHUNK_ALIGN_MASK
 - dl_malloc.c, 360
- CHUNK_OVERHEAD
 - dl_malloc.c, 361
- CINUSE_BIT
 - dl_malloc.c, 361
- CMFAIL
 - dl_malloc.c, 362
- COMPARATOR
 - dist_object.hpp, 351
- CORRUPTION_ERROR_ACTION
 - dl_malloc.c, 363
- calloc_must_clear
 - dl_malloc.c, 359
- check_free_chunk
 - dl_malloc.c, 359
- check_inuse_chunk
 - dl_malloc.c, 359
- check_malloc_state
 - dl_malloc.c, 359
- check_malloced_chunk
 - dl_malloc.c, 360
- check_mmapped_chunk
 - dl_malloc.c, 360
- check_top_chunk
 - dl_malloc.c, 360
- child
 - malloc_tree_chunk, 213
- chunk2mem
 - dl_malloc.c, 360
- chunk_minus_offset
 - dl_malloc.c, 361
- chunk_plus_offset
 - dl_malloc.c, 361
- chunksizes
 - dl_malloc.c, 361
- cinuse
 - dl_malloc.c, 361
- cleanup_early
 - upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >, 123
 - upcxx::detail::future_dependency< future1< future_kind_result > >, 125
 - upcxx::detail::future_dependency< future1< future_kind_result, T... > >, 127
- cleanup_ready
 - upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >, 123
 - upcxx::detail::future_dependency< future1< future_kind_result > >, 125
 - upcxx::detail::future_dependency< future1< future_kind_result, T... > >, 127
- cleanup_ready_get_header
 - upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >, 123
 - upcxx::detail::future_dependency< future1< future_kind_result > >, 125
 - upcxx::detail::future_dependency< future1< future_kind_result, T... > >, 127
- clear_flag4
 - dl_malloc.c, 362
- clear_pinuse
 - dl_malloc.c, 362
- clear_smallmap
 - dl_malloc.c, 362
- clear_treemap
 - dl_malloc.c, 362
- comma
 - upcxx::print_reflector, 266

- command.hpp, 345
- command_execute
 - upcxx, 27
- command_executor
 - upcxx::detail, 50
- command_pack
 - upcxx, 27, 28
- command_size_ubound
 - upcxx, 28
- completed
 - upcxx::detail::rget_state_operxn< rget_byref, future_cx< ordinal > >, 283
 - upcxx::detail::rget_state_operxn< rget_byref, nil_cx >, 284
 - upcxx::detail::rget_state_operxn< rget_byref, promise_cx<> >, 285
 - upcxx::detail::rget_state_operxn< rget_byval< T >, future_cx< ordinal > >, 287
 - upcxx::detail::rget_state_operxn< rget_byval< T >, nil_cx >, 288
 - upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T > >, 289
 - upcxx::detail::rget_state_remote< nil_cx >, 291
 - upcxx::detail::rget_state_remote< rpc_cx< Fn > >, 292
 - upcxx::detail::rput_state_here< future_cx< ordinal > >, 313
 - upcxx::detail::rput_state_here< nil_cx >, 314
 - upcxx::detail::rput_state_here< promise_cx<> >, 315
 - upcxx::detail::rput_state_remote< nil_cx >, 317
 - upcxx::detail::rput_state_remote< rpc_cx< Fn > >, 318
- completion.hpp, 346
- compute_bit2idx
 - dl_malloc.c, 363
- compute_tree_index
 - dl_malloc.c, 363
- constant
 - upcxx, 28
- constant_function
 - upcxx::constant_function, 85
- construct_results
 - upcxx::detail::future_header_result, 150
 - upcxx::detail::future_header_result<>, 154
- contributed
 - upcxx::detail::allreduce_state, 58
- core.cpp
 - future_header_result, 409
- create_mspace
 - dl_malloc.c, 396
 - dl_malloc.h, 403
- create_mspace_with_base
 - dl_malloc.c, 396
 - dl_malloc.h, 403
- DEBUG
 - dl_malloc.c, 364
- DEFAULT_GRANULARITY
 - dl_malloc.c, 364
- DEFAULT_MMAP_THRESHOLD
 - dl_malloc.c, 364
- DEFAULT_TRIM_THRESHOLD
 - dl_malloc.c, 364
- DESTROY_LOCK
 - dl_malloc.c, 364
- DIRECT_MMAP_DEFAULT
 - dl_malloc.c, 364
- DLMALLOC_EXPORT
 - dl_malloc.c, 365
- DLMALLOC_VERSION
 - dl_malloc.c, 365
- dbgbrk
 - upcxx, 28
- dbgbrk_spin
 - upcxx, 39
- dbgbrk_spin_init
 - upcxx, 39
- deallocate
 - upcxx, 29
- decref_header
 - upcxx::detail::future_header_ops_general, 145
 - upcxx::detail::future_header_ops_result, 146
 - upcxx::detail::future_header_ops_result_ready, 148
- default_mflags
 - malloc_params, 206
- delete_
 - upcxx, 29
- delete_array
 - upcxx, 29
- delete_header
 - upcxx::detail::future_header_ops_general, 145
 - upcxx::detail::future_header_ops_result, 147
 - upcxx::detail::future_header_ops_result_ready, 148
- delete_me
 - upcxx::detail::future_header_result, 151
 - upcxx::detail::future_header_result<>, 154
- delete_me_ready
 - upcxx::detail::future_header_result, 151
 - upcxx::detail::future_header_result<>, 154
- dep
 - upcxx::detail::future_header::dependency_link, 88
- dep_
 - upcxx::detail::future_body_pure< future1< Kind, T... > >, 116
 - upcxx::detail::future_body_then, 117
 - upcxx::detail::future_body_then_pure, 121
 - upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >, 124
 - upcxx::detail::future_dependency_when_all_arg, 134
- destroy_mspace
 - dl_malloc.c, 396
 - dl_malloc.h, 403

- destruct
 - upcxx::detail::packing_tuple_each, 248
 - upcxx::detail::packing_tuple_each< n, n, T... >, 250
- destruct_early
 - upcxx::detail::future_body, 110
 - upcxx::detail::future_body_proxy, 112
 - upcxx::detail::future_body_pure< future1< Kind, T... > >, 115
 - upcxx::detail::future_body_then_pure, 120
- diagnostic.cpp, 347
- diagnostic.hpp, 347
 - UPCXX_ASSERT_1, 348
 - UPCXX_ASSERT_2, 348
 - UPCXX_ASSERT_ALWAYS, 349
 - UPCXX_ASSERT_DISPATCH, 349
 - UPCXX_ASSERT, 348
 - UPCXX_INVOKE_UB, 349
- dig_
 - upcxx::dist_id, 96
- digest.cpp, 349
- digest.hpp, 350
- disable_contiguous
 - dl_malloc.c, 365
- disable_lock
 - dl_malloc.c, 365
- disable_mmap
 - dl_malloc.c, 365
- dist_master_id_bump
 - upcxx::detail, 52
- dist_master_promises
 - upcxx::detail, 52
- dist_object
 - upcxx::dist_object, 97, 98
- dist_object.cpp, 350
- dist_object.hpp, 350
 - COMPARATOR, 351
- dist_promise
 - upcxx::detail, 50
- dl_malloc.c, 351
 - _STRUCT_MALLINFO, 356
 - ABORT_ON_ASSERT_FAILURE, 357
 - ABORT, 356
 - ACQUIRE_LOCK, 357
 - ACQUIRE_MALLOC_GLOBAL_LOCK, 357
 - align_as_chunk, 357
 - align_offset, 357
 - assert, 357
 - bindex_t, 394
 - binmap_t, 394
 - bit_for_tree_index, 358
 - CALL_DIRECT_MMAP, 358
 - CALL_MMAP, 358
 - CALL_MORECORE, 358
 - CALL_MREMAP, 358
 - CALL_MUNMAP, 359
 - CHUNK_ALIGN_MASK, 360
 - CHUNK_OVERHEAD, 361
 - CINUSE_BIT, 361
 - CMFAIL, 362
 - CORRUPTION_ERROR_ACTION, 363
 - calloc_must_clear, 359
 - check_free_chunk, 359
 - check_inuse_chunk, 359
 - check_malloc_state, 359
 - check_mallocated_chunk, 360
 - check_mmapped_chunk, 360
 - check_top_chunk, 360
 - chunk2mem, 360
 - chunk_minus_offset, 361
 - chunk_plus_offset, 361
 - chunksizes, 361
 - cinuse, 361
 - clear_flag4, 362
 - clear_pinuse, 362
 - clear_smallmap, 362
 - clear_treemap, 362
 - compute_bit2idx, 363
 - compute_tree_index, 363
 - create_mspace, 396
 - create_mspace_with_base, 396
 - DEBUG, 364
 - DEFAULT_GRANULARITY, 364
 - DEFAULT_MMAP_THRESHOLD, 364
 - DEFAULT_TRIM_THRESHOLD, 364
 - DESTROY_LOCK, 364
 - DIRECT_MMAP_DEFAULT, 364
 - DLMALLOC_EXPORT, 365
 - DLMALLOC_VERSION, 365
 - destroy_mspace, 396
 - disable_contiguous, 365
 - disable_lock, 365
 - disable_mmap, 365
 - EXTERN_BIT, 366
 - enable_lock, 365
 - enable_mmap, 366
 - ensure_initialization, 366
 - FENCEPOST_HEAD, 366
 - FLAG4_BIT, 366
 - FLAG_BITS, 367
 - FOOTERS, 367
 - FORCEINLINE, 367
 - FOUR_SIZE_T_SIZES, 367
 - flag4inuse, 366
 - flag_t, 394
 - get_foot, 367
 - granularity_align, 367
 - HALF_MAX_SIZE_T, 368
 - HAVE_MMAP, 368
 - HAVE_MORECORE, 368
 - HAVE_MREMAP, 368
 - INITIAL_LOCK, 369
 - INSECURE, 369
 - INUSE_BITS, 370
 - idx2bit, 368
 - insert_chunk, 369

insert_large_chunk, 369
insert_small_chunk, 369
internal_free, 370
internal_malloc, 370
is_aligned, 370
is_extern_segment, 371
is_granularity_aligned, 371
is_initialized, 371
is_inuse, 371
is_mmapped, 371
is_mmapped_segment, 371
is_page_aligned, 372
is_small, 372
LOCK_AT_FORK, 373
least_bit, 372
left_bits, 372
leftmost_child, 372
leftshift_for_tree_index, 372
M_GRANULARITY, 373
M_MMAP_THRESHOLD, 373
M_TRIM_THRESHOLD, 373
MALLINFO_FIELD_TYPE, 373
MALLOC_ALIGNMENT, 374
MALLOC_FAILURE_ACTION, 374
MALLOC_INSPECT_ALL, 374
MAX_RELEASE_CHECK_RATE, 375
MAX_REQUEST, 375
MAX_SIZE_T, 375
MAX_SMALL_REQUEST, 375
MAX_SMALL_SIZE, 375
MCHUNK_SIZE, 376
MFAIL, 376
MIN_CHUNK_SIZE, 376
MIN_LARGE_SIZE, 376
MIN_REQUEST, 376
MIN_SMALL_INDEX, 377
MLOCK_T, 377
MMAP_CHUNK_OVERHEAD, 377
MMAP_CLEARS, 378
MMAP_DEFAULT, 378
MMAP_FLAGS, 378
MMAP_FOOT_PAD, 378
MMAP_PROT, 378
MORECORE_CONTIGUOUS, 379
MSPACES, 379
MUNMAP_DEFAULT, 379
malloc_getpagesize, 374
mark_inuse_foot, 374
mark_smallmap, 374
mark_treemap, 375
mchunk, 394
mchunkptr, 394
mem2chunk, 376
minsize_for_tree_index, 377
mmap_align, 377
msegment, 395
msegmentptr, 395
mspace, 395
mspace_bulk_free, 397
mspace_calloc, 397
mspace_footprint, 397
mspace_footprint_limit, 397
mspace_free, 397
mspace_independent_calloc, 398
mspace_independent_comalloc, 398
mspace_mallinfo, 398
mspace_malloc, 398
mspace_malloc_stats, 398
mspace_mallopt, 399
mspace_max_footprint, 399
mspace_memalign, 399
mspace_realloc, 399
mspace_realloc_in_place, 399
mspace_set_footprint_limit, 400
mspace_track_large_chunks, 400
mspace_trim, 400
mspace_usable_size, 400
mstate, 395
NO_MALLINFO, 379
NO_MALLOC_STATS, 380
NO_SEGMENT_TRAVERSAL, 380
NOINLINE, 380
NSMALLBINS, 380
NTREEBINS, 380
next_chunk, 379
next_pinuse, 379
ONLY_MSPACES, 381
ok_address, 380
ok_inuse, 381
ok_magic, 381
ok_next, 381
ok_pinuse, 381
overhead_for, 382
PINUSE_BIT, 382
POSTACTION, 382
PREACTION, 383
PROCEED_ON_ERROR, 383
pad_request, 382
page_align, 382
pinuse, 382
prev_chunk, 383
RELEASE_LOCK, 383
RELEASE_MALLOC_GLOBAL_LOCK, 383
RTCHECK, 384
replace_dv, 383
request2size, 384
SIX_SIZE_T_SIZES, 387
SIZE_T_BITSIZE, 387
SIZE_T_FOUR, 387
SIZE_T_ONE, 387
SIZE_T_SIZE, 387
SIZE_T_TWO, 387
SIZE_T_ZERO, 388
SMALLBIN_SHIFT, 388
SMALLBIN_WIDTH, 388
STRUCT_MALLINFO_DECLARED, 389

- SYS_ALLOC_PADDING, 389
- same_or_left_bits, 384
- sbinptr, 395
- segment_holds, 384
- set_flag4, 385
- set_foot, 385
- set_free_with_pinuse, 385
- set_inuse, 385
- set_inuse_and_pinuse, 385
- set_lock, 386
- set_size_and_pinuse_of_free_chunk, 386
- set_size_and_pinuse_of_inuse_chunk, 386
- should_trim, 386
- small_index, 388
- small_index2size, 388
- smallbin_at, 388
- smallmap_is_marked, 389
- TOP_FOOT_SIZE, 389
- TREEBIN_SHIFT, 389
- TRY_LOCK, 390
- TWO_SIZE_T_SIZES, 390
- tbinptr, 395
- tchunk, 396
- tchunkptr, 396
- treebin_at, 389
- treemap_is_marked, 390
- USAGE_ERROR_ACTION, 392
- USE_BUILTIN_FFS, 392
- USE_DEV_RANDOM, 392
- USE_LOCK_BIT, 392
- USE_LOCKS, 393
- USE_MMAP_BIT, 393
- USE_NONCONTIGUOUS_BIT, 393
- USE_SPIN_LOCKS, 394
- unlink_chunk, 390
- unlink_first_small_chunk, 390
- unlink_large_chunk, 391
- unlink_small_chunk, 391
- use_lock, 392
- use_mmap, 393
- use_noncontiguous, 393
- dl_malloc.h, 401
 - create_mspace, 403
 - create_mspace_with_base, 403
 - destroy_mspace, 403
 - dlmalloc_usable_size, 403
 - MALLINFO_FIELD_TYPE, 402
 - MSPACES, 402
 - mspace, 403
 - mspace_bulk_free, 404
 - mspace_calloc, 404
 - mspace_footprint, 404
 - mspace_footprint_limit, 404
 - mspace_free, 404
 - mspace_independent_calloc, 405
 - mspace_independent_comalloc, 405
 - mspace_inspect_all, 405
 - mspace_mallinfo, 405
 - mspace_malloc, 405
 - mspace_malloc_stats, 406
 - mspace_mallopt, 406
 - mspace_max_footprint, 406
 - mspace_memalign, 406
 - mspace_realloc, 406
 - mspace_realloc_in_place, 407
 - mspace_set_footprint_limit, 407
 - mspace_track_large_chunks, 407
 - mspace_trim, 407
 - mspace_usable_size, 407
 - NO_MALLINFO, 402
 - ONLY_MSPACES, 402
 - STRUCT_MALLINFO_DECLARED, 402
- dlmalloc_usable_size
 - dl_malloc.h, 403
- drop_for_proxied
 - upcxx::detail::future_header, 139
- drop_for_result
 - upcxx::detail::future_header, 139
- during_level
 - upcxx::backend, 40, 41
- during_user
 - upcxx::backend, 41
- dv
 - malloc_state, 209
- dvsize
 - malloc_state, 209
- EXTERN_BIT
 - dl_malloc.c, 366
- eat
 - upcxx::digest, 89
- element_type
 - upcxx::global_ptr, 181
- embed
 - upcxx::parcel_layout, 257
- enable_lock
 - dl_malloc.c, 365
- enable_mmap
 - dl_malloc.c, 366
- ensure_initialization
 - dl_malloc.c, 366
- enter_proxying
 - upcxx::detail::future_header_dependent, 143
- enter_ready
 - upcxx::detail::future_header, 139
- entered_active
 - upcxx::detail::future_header_dependent, 143
- execute
 - upcxx::commanding, 82
- extp
 - malloc_state, 209
- exts
 - malloc_state, 209
- FENCEPOST_HEAD
 - dl_malloc.c, 366
- FLAG4_BIT

- dl_malloc.c, [366](#)
- FLAG_BITS
 - dl_malloc.c, [367](#)
- FOOTERS
 - dl_malloc.c, [367](#)
- FORCEINLINE
 - dl_malloc.c, [367](#)
- FOUR_SIZE_T_SIZES
 - dl_malloc.c, [367](#)
- fast_hash
 - upcxx, [29](#)
- fast_hasher
 - upcxx::fast_hasher, [99](#)
- fd
 - malloc_chunk, [205](#)
 - malloc_tree_chunk, [213](#)
- fetch
 - upcxx::dist_object, [98](#)
- finalize
 - upcxx, [30](#)
- fire_and_delete
 - upcxx::backend::gasnet1_seq::action, [53](#)
 - upcxx::backend::gasnet1_seq::action_impl, [55](#)
 - upcxx::backend::gasnet1_seq::rma_cb, [294](#)
- flag4inuse
 - dl_malloc.c, [366](#)
- flag_t
 - dl_malloc.c, [394](#)
- fn
 - upcxx::backend::gasnet1_seq::action_impl, [55](#)
 - upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator, [61](#)
 - upcxx::detail::rget_state_remote< rpc_cx< Fn > >, [292](#)
 - upcxx::detail::rput_state_remote< rpc_cx< Fn > >, [318](#)
- fn_
 - upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >, [80](#)
 - upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true >, [81](#)
 - upcxx::detail::future_body_then, [117](#)
 - upcxx::detail::future_body_then_pure, [121](#)
 - upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >, [124](#)
 - upcxx::detail::future_impl_mapped, [158](#)
 - upcxx::rpc_cx, [302](#)
- fn_return_getter_t
 - upcxx::detail::future_impl_mapped::result_lrefs_↔ function, [276](#)
- fn_return_t
 - upcxx::detail::future_impl_mapped::result_lrefs_↔ function, [274](#)
- footprint
 - malloc_state, [209](#)
- footprint_limit
 - malloc_state, [209](#)
- fordblks
 - mallinfo, [203](#)
- fsmblocks
 - mallinfo, [203](#)
- function_ref
 - upcxx::function_ref< Ret(Arg...)>, [102](#), [103](#)
- fut_
 - upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > >, [280](#)
 - upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > >, [282](#)
 - upcxx::detail::rput_futures_of< future_cx< S_ord >, R, O >, [310](#)
 - upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > >, [312](#)
- future
 - upcxx, [23](#)
- future.hpp, [408](#)
- future/apply.hpp, [408](#)
- future/body_pure.hpp, [409](#)
- future/core.cpp, [409](#)
- future/core.hpp, [409](#)
- future/future1.hpp, [410](#)
- future/impl_mapped.hpp, [411](#)
- future/impl_result.hpp, [411](#)
- future/impl_shref.hpp, [412](#)
- future/impl_when_all.hpp, [412](#)
- future/make_future.hpp, [412](#)
- future/promise.hpp, [413](#)
- future/then.hpp, [414](#)
- future/when_all.hpp, [414](#)
- future1
 - upcxx::future1, [105–107](#)
- future_body
 - upcxx::detail::future_body, [110](#)
- future_body_proxy
 - upcxx::detail::future_body_proxy, [112](#)
- future_body_proxy_
 - upcxx::detail::future_body_proxy_, [113](#)
- future_body_pure
 - upcxx::detail::future_body_pure< future1< Kind, T... > >, [115](#)
- future_body_then
 - upcxx::detail::future_body_then, [117](#)
- future_body_then_base
 - upcxx::detail::future_body_then_base, [118](#)
- future_body_then_pure
 - upcxx::detail::future_body_then_pure, [120](#)
- future_dependency
 - upcxx::detail::future_dependency< future1< future_kind_mapped< FuArg, Fn >, T... > >, [123](#)
 - upcxx::detail::future_dependency< future1< future_kind_result > >, [125](#)
 - upcxx::detail::future_dependency< future1< future_kind_result, T... > >, [127](#)
 - upcxx::detail::future_dependency< future1< future_kind_shref< HeaderOps >, T... >

- >, 129
- upcxx::detail::future_dependency< future1< future_kind_when_all< Arg... >, T... > >, 129
- future_dependency_shref_base
 - upcxx::detail::future_dependency_shref_base< false >, 131
 - upcxx::detail::future_dependency_shref_base< true >, 132
- future_dependency_when_all_arg
 - upcxx::detail::future_dependency_when_all_arg, 134
- future_dependency_when_all_base
 - upcxx::detail::future_dependency_when_all_↔ base< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... > >, 136
- future_from_tuple_t
 - upcxx::detail, 49
- future_header_dependent
 - upcxx::detail::future_header_dependent, 143
- future_header_result
 - core.cpp, 409
 - upcxx::detail::future_header_result, 150
 - upcxx::detail::future_header_result<>, 154
- future_impl_mapped
 - upcxx::detail::future_impl_mapped, 156
- future_impl_result
 - upcxx::detail::future_impl_result, 159
- future_impl_shref
 - upcxx::detail::future_impl_shref, 163, 164
- future_impl_when_all
 - upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >, 168
- future_n
 - upcxx::completions, 84
- gasnet1_seq
 - backend.hpp, 344
- get_foot
 - dl_malloc.c, 367
- get_or_void
 - upcxx, 30
- getter_
 - upcxx::detail::future_impl_mapped::result_lrefs_↔ function, 276
- global_ptr
 - upcxx::global_ptr, 182
- global_ptr.hpp, 415
- granularity
 - malloc_params, 206
- granularity_align
 - dl_malloc.c, 367
- h
 - upcxx::hasher_reflector, 193
- HALF_MAX_SIZE_T
 - dl_malloc.c, 368
- HAVE_MMAP
 - dl_malloc.c, 368
- HAVE_MORECORE
 - dl_malloc.c, 368
- HAVE_MREMAP
 - dl_malloc.c, 368
- handle
 - upcxx::backend::gasnet1_seq::rma_cb, 295
- hblkhd
 - mallinfo, 203
- hblks
 - mallinfo, 203
- hdr_
 - upcxx::detail::future_dependency_shref_base< true >, 133
 - upcxx::detail::future_impl_shref, 166
- head
 - malloc_chunk, 205
 - malloc_tree_chunk, 213
- header_
 - upcxx::detail::future_dependency_shref_base< false >, 131
 - upcxx::detail::future_dependency_shref_base< true >, 133
- header_ops
 - upcxx::detail::future_impl_mapped, 156
 - upcxx::detail::future_impl_result, 159
 - upcxx::detail::future_impl_result<>, 161
 - upcxx::detail::future_impl_shref, 163
 - upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >, 167
- here
 - upcxx::dist_id, 96
- INITIAL_LOCK
 - dl_malloc.c, 369
- INSECURE
 - dl_malloc.c, 369
- INUSE_BITS
 - dl_malloc.c, 370
- id
 - upcxx::dist_object, 98
- idx2bit
 - dl_malloc.c, 368
- impl_
 - upcxx::future1, 109
- impl_type
 - upcxx::future1, 105
- in_user_progress_
 - upcxx::backend::gasnet1_seq, 45
- incoming
 - upcxx::detail::allreduce_state, 59
- index
 - malloc_tree_chunk, 213
- init
 - upcxx, 30
- initiator_
 - upcxx::detail::rpc_recipient_after, 304
- insert_chunk
 - dl_malloc.c, 369

- insert_large_chunk
 - dl_malloc.c, [369](#)
- insert_small_chunk
 - dl_malloc.c, [369](#)
- internal
 - upcxx, [24](#)
- internal_free
 - dl_malloc.c, [370](#)
- internal_malloc
 - dl_malloc.c, [370](#)
- inrank_t
 - upcxx, [23](#)
- is_aligned
 - dl_malloc.c, [370](#)
- is_extern_segment
 - dl_malloc.c, [371](#)
- is_granularity_aligned
 - dl_malloc.c, [371](#)
- is_initialized
 - dl_malloc.c, [371](#)
- is_inuse
 - dl_malloc.c, [371](#)
- is_local
 - upcxx::global_ptr, [182](#)
- is_memory_shared_with
 - upcxx, [30](#)
- is_mmapped
 - dl_malloc.c, [371](#)
- is_mmapped_segment
 - dl_malloc.c, [371](#)
- is_null
 - upcxx::global_ptr, [183](#)
- is_page_aligned
 - dl_malloc.c, [372](#)
- is_small
 - dl_malloc.c, [372](#)
- is_trivial
 - upcxx::binding, [71](#)
 - upcxx::packing_empty< T, false >, [232](#)
 - upcxx::packing_empty< T, true >, [233](#)
 - upcxx::packing_function_pointer, [235](#)
 - upcxx::packing_trivial, [247](#)
- is_trivially_ready_result
 - upcxx::detail::future_header_ops_general, [145](#)
 - upcxx::detail::future_header_ops_result, [147](#)
 - upcxx::detail::future_header_ops_result_ready, [148](#)
- keepcost
 - mallinfo, [203](#)
- kind_type
 - upcxx::future1, [105](#)
- LOCK_AT_FORK
 - dl_malloc.c, [373](#)
- lay_
 - upcxx::parcel_writer, [264](#)
- layout
 - upcxx::parcel_reader, [259](#)
 - upcxx::parcel_writer, [262](#)
- least_addr
 - malloc_state, [210](#)
- least_bit
 - dl_malloc.c, [372](#)
- leave_active
 - upcxx::detail::future_body, [110](#)
 - upcxx::detail::future_body_proxy_, [113](#)
 - upcxx::detail::future_body_pure< future1< Kind, T... > >, [115](#)
 - upcxx::detail::future_body_then, [117](#)
 - upcxx::detail::future_body_then_pure, [120](#)
- leave_active_into_proxy
 - upcxx::detail::future_body_then_base, [119](#)
- left_bits
 - dl_malloc.c, [372](#)
- leftmost_child
 - dl_malloc.c, [372](#)
- leftshift_for_tree_index
 - dl_malloc.c, [372](#)
- link_
 - upcxx::detail::future_body_proxy_, [113](#)
 - upcxx::detail::future_dependency_shref_base< false >, [132](#)
- local
 - upcxx::global_ptr, [183](#)
- M_GRANULARITY
 - dl_malloc.c, [373](#)
- M_MMAP_THRESHOLD
 - dl_malloc.c, [373](#)
- M_TRIM_THRESHOLD
 - dl_malloc.c, [373](#)
- MALLINFO_FIELD_TYPE
 - dl_malloc.c, [373](#)
 - dl_malloc.h, [402](#)
- MALLOC_ALIGNMENT
 - dl_malloc.c, [374](#)
- MALLOC_FAILURE_ACTION
 - dl_malloc.c, [374](#)
- MALLOC_INSPECT_ALL
 - dl_malloc.c, [374](#)
- MAX_RELEASE_CHECK_RATE
 - dl_malloc.c, [375](#)
- MAX_REQUEST
 - dl_malloc.c, [375](#)
- MAX_SIZE_T
 - dl_malloc.c, [375](#)
- MAX_SMALL_REQUEST
 - dl_malloc.c, [375](#)
- MAX_SMALL_SIZE
 - dl_malloc.c, [375](#)
- MCHUNK_SIZE
 - dl_malloc.c, [376](#)
- MFAIL
 - dl_malloc.c, [376](#)
- MIN_CHUNK_SIZE
 - dl_malloc.c, [376](#)
- MIN_LARGE_SIZE

- dl_malloc.c, 376
- MIN_REQUEST
 - dl_malloc.c, 376
- MIN_SMALL_INDEX
 - dl_malloc.c, 377
- MLOCK_T
 - dl_malloc.c, 377
- MMAP_CHUNK_OVERHEAD
 - dl_malloc.c, 377
- MMAP_CLEAR
 - dl_malloc.c, 378
- MMAP_DEFAULT
 - dl_malloc.c, 378
- MMAP_FLAGS
 - dl_malloc.c, 378
- MMAP_FOOT_PAD
 - dl_malloc.c, 378
- MMAP_PROT
 - dl_malloc.c, 378
- MORECORE_CONTIGUOUS
 - dl_malloc.c, 379
- MSPACES
 - dl_malloc.c, 379
 - dl_malloc.h, 402
- MUNMAP_DEFAULT
 - dl_malloc.c, 379
- magic
 - malloc_params, 206
 - malloc_state, 210
- make_future
 - upcxx, 30
- make_index_sequence
 - upcxx, 23
- make_rma_get_cb
 - upcxx::backend, 42
- make_rma_put_cb
 - upcxx::backend, 42
- mallinfo, 202
 - arena, 203
 - fordblks, 203
 - fsmblocks, 203
 - hblkhd, 203
 - hblks, 203
 - keepcost, 203
 - ordblks, 204
 - smblocks, 204
 - uordblks, 204
 - usmblocks, 204
- malloc_chunk, 204
 - bk, 205
 - fd, 205
 - head, 205
 - prev_foot, 205
- malloc_getpagesize
 - dl_malloc.c, 374
- malloc_params, 206
 - default_mflags, 206
 - granularity, 206
 - magic, 206
 - mmap_threshold, 206
 - page_size, 207
 - trim_threshold, 207
- malloc_segment, 207
 - base, 207
 - next, 208
 - sflags, 208
 - size, 208
- malloc_state, 208
 - dv, 209
 - dvsizes, 209
 - extp, 209
 - exts, 209
 - footprint, 209
 - footprint_limit, 209
 - least_addr, 210
 - magic, 210
 - max_footprint, 210
 - mflags, 210
 - mutex, 210
 - release_checks, 210
 - seg, 211
 - smallbins, 211
 - smallmap, 211
 - top, 211
 - topsize, 211
 - treebins, 211
 - treemap, 212
 - trim_check, 212
- malloc_tree_chunk, 212
 - bk, 212
 - child, 213
 - fd, 213
 - head, 213
 - index, 213
 - parent, 213
 - prev_foot, 213
- mark_inuse_foot
 - dl_malloc.c, 374
- mark_smallmap
 - dl_malloc.c, 374
- mark_treemap
 - dl_malloc.c, 375
- max_footprint
 - malloc_state, 210
- mchunk
 - dl_malloc.c, 394
- mchunkptr
 - dl_malloc.c, 394
- mem2chunk
 - dl_malloc.c, 376
- mflags
 - malloc_state, 210
- minsize_for_tree_index
 - dl_malloc.c, 377
- mmap_align
 - dl_malloc.c, 377

- mmap_threshold
 - malloc_params, 206
- mod2n_hash
 - upcxx, 31
- move_from_storage
 - upcxx::packing< std::tuple< T... > >, 224
- msegment
 - dl_malloc.c, 395
- msegmentptr
 - dl_malloc.c, 395
- mspace
 - dl_malloc.c, 395
 - dl_malloc.h, 403
- mspace_bulk_free
 - dl_malloc.c, 397
 - dl_malloc.h, 404
- mspace_calloc
 - dl_malloc.c, 397
 - dl_malloc.h, 404
- mspace_footprint
 - dl_malloc.c, 397
 - dl_malloc.h, 404
- mspace_footprint_limit
 - dl_malloc.c, 397
 - dl_malloc.h, 404
- mspace_free
 - dl_malloc.c, 397
 - dl_malloc.h, 404
- mspace_independent_calloc
 - dl_malloc.c, 398
 - dl_malloc.h, 405
- mspace_independent_comalloc
 - dl_malloc.c, 398
 - dl_malloc.h, 405
- mspace_inspect_all
 - dl_malloc.h, 405
- mspace_mallinfo
 - dl_malloc.c, 398
 - dl_malloc.h, 405
- mspace_malloc
 - dl_malloc.c, 398
 - dl_malloc.h, 405
- mspace_malloc_stats
 - dl_malloc.c, 398
 - dl_malloc.h, 406
- mspace_mallopt
 - dl_malloc.c, 399
 - dl_malloc.h, 406
- mspace_max_footprint
 - dl_malloc.c, 399
 - dl_malloc.h, 406
- mspace_memalign
 - dl_malloc.c, 399
 - dl_malloc.h, 406
- mspace_realloc
 - dl_malloc.c, 399
 - dl_malloc.h, 406
- mspace_realloc_in_place
 - dl_malloc.c, 399
 - dl_malloc.h, 407
- mspace_set_footprint_limit
 - dl_malloc.c, 400
 - dl_malloc.h, 407
- mspace_track_large_chunks
 - dl_malloc.c, 400
 - dl_malloc.h, 407
- mspace_trim
 - dl_malloc.c, 400
 - dl_malloc.h, 407
- mspace_usable_size
 - dl_malloc.c, 400
 - dl_malloc.h, 407
- mstate
 - dl_malloc.c, 395
- mutex
 - malloc_state, 210
- NO_MALLINFO
 - dl_malloc.c, 379
 - dl_malloc.h, 402
- NO_MALLOC_STATS
 - dl_malloc.c, 380
- NO_SEGMENT_TRAVERSAL
 - dl_malloc.c, 380
- NOINLINE
 - dl_malloc.c, 380
- NSMALLBINS
 - dl_malloc.c, 380
- NTREEBINS
 - dl_malloc.c, 380
- new_
 - upcxx, 31
 - upcxx::detail, 51
- new_array
 - upcxx, 31
 - upcxx::detail, 51
- next
 - malloc_segment, 208
 - upcxx::backend::gasnet1_seq::rma_cb, 295
- next_
 - upcxx::backend::gasnet1_seq::action, 54
- next_chunk
 - dl_malloc.c, 379
- next_pinuse
 - dl_malloc.c, 379
- nobsrule, 17
 - requires_upcxx_backend, 17
- nobsrule.py, 415
- nop
 - upcxx, 32
- o
 - upcxx::detail::rget_states, 293
 - upcxx::detail::rput_states< rput_byref, S, R, O >, 320
 - upcxx::print_reflector, 267
- o_

- upcxx::detail::rput_futures_of< future_cx< S_ord
>, R, future_cx< O_ord > >, 308
- ONLY_MSPACES
 - dl_malloc.c, 381
 - dl_malloc.h, 402
- off_wire
 - upcxx::binding, 70
 - upcxx::binding< dist_object< T > & >, 72
- off_wire_type
 - upcxx::binding, 70
 - upcxx::binding< dist_object< T > & >, 72
- ok_address
 - dl_malloc.c, 380
- ok_inuse
 - dl_malloc.c, 381
- ok_magic
 - dl_malloc.c, 381
- ok_next
 - dl_malloc.c, 381
- ok_pinuse
 - dl_malloc.c, 381
- on_wire
 - upcxx::binding, 70
 - upcxx::binding< dist_object< T > & >, 72
- on_wire_type
 - upcxx::binding, 70
 - upcxx::binding< dist_object< T > & >, 72
- op
 - upcxx::detail::allreduce_state, 59
- op_cx
 - upcxx::backend::gasnet1_seq::rma_get_cb_impl,
297
 - upcxx::backend::gasnet1_seq::rma_put_cb_impl,
300
- opaque
 - upcxx::hasher_reflector, 193
 - upcxx::packing_pack_reflector, 240
 - upcxx::packing_ubound_reflector, 251
 - upcxx::packing_unpack_reflector< false >, 253
 - upcxx::packing_unpack_reflector< true >, 254
 - upcxx::print_reflector, 266
- operator!=
 - upcxx::digest, 90
 - upcxx::global_ptr, 183
- operator<
 - upcxx::digest, 90
 - upcxx::global_ptr, 185
- operator<<
 - upcxx, 32
 - upcxx::global_ptr, 186
 - upcxx::parcel_layout, 257
 - upcxx::print_proxy, 265
- operator<=
 - upcxx::digest, 90
 - upcxx::global_ptr, 185
- operator>
 - upcxx::digest, 90
 - upcxx::global_ptr, 185
- operator>>
 - upcxx, 33
- operator>>=
 - upcxx, 33
- operator>=
 - upcxx::digest, 90
 - upcxx::global_ptr, 185
- operator*
 - upcxx::dist_object, 98
- operator()
 - std::greater< upcxx::global_ptr< T > >, 188
 - std::greater_equal< upcxx::global_ptr< T > >,
189
 - std::hash< upcxx::digest >, 190
 - std::hash< upcxx::dist_id< T > >, 191
 - std::hash< upcxx::global_ptr< T > >, 191
 - std::less< upcxx::global_ptr< T > >, 198
 - std::less_equal< upcxx::global_ptr< T > >, 198
 - upcxx::bound_function, 78
 - upcxx::constant_function, 85
 - upcxx::detail::apply_futered_as_future< Fn(future1< Kind, T... >)>, 62
 - upcxx::detail::apply_tupled_as_future_impl, 65
 - upcxx::detail::apply_tupled_as_future_impl< future1< Kind, T... > >, 66
 - upcxx::detail::apply_tupled_as_future_impl< void >,
67
 - upcxx::detail::bind, 68
 - upcxx::detail::bind< bound_function< Fn0, B0... > >,
69
 - upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator, 60
 - upcxx::detail::disjoin_cx< A, nil_cx, B_ord_bump >,
92
 - upcxx::detail::disjoin_cx< nil_cx, B, B_ord_bump >,
93
 - upcxx::detail::disjoin_cx< nil_cx, future_cx< B_← ord_old >, B_ord_bump >, 94
 - upcxx::detail::disjoin_cx< nil_cx, nil_cx, B_ord0 >,
95
 - upcxx::detail::future_impl_mapped::result_lrefs_← function, 275
 - upcxx::detail::future_then< Arg, Fn, future1< Fn_← RetKind, FnRetT... >, false >, 176
 - upcxx::detail::future_then< Arg, Fn, future1< Fn_← RetKind, FnRetT... >, true >, 177
 - upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, false >, 178
 - upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, false, true >, 179
 - upcxx::detail::future_then_pure< Arg, Fn, future1< FnRetKind, FnRetT... >, true, fnret_trivial >,
180
 - upcxx::detail::make_future_< false, T... >, 200
 - upcxx::detail::make_future_< true, T... >, 201
 - upcxx::detail::os_env_parse, 217
 - upcxx::detail::os_env_parse< std::string >, 218
 - upcxx::detail::rget_futures_of, 278

- upcxx::detail::rget_futures_of< rget_byref, R, future_cx< 0 > >, 280
- upcxx::detail::rget_futures_of< rget_byval< T >, R, future_cx< 0 > >, 281
- upcxx::detail::rpc_recipient_after, 303
- upcxx::detail::rput_futures_of, 306
- upcxx::detail::rput_futures_of< future_cx< S_ord >, R, future_cx< O_ord > >, 308
- upcxx::detail::rput_futures_of< future_cx< S_ord >, R, O >, 310
- upcxx::detail::rput_futures_of< S, R, future_cx< O_ord > >, 311
- upcxx::detail::tuple_get_or_void, 328
- upcxx::detail::tuple_get_or_void< i, TupRef, false >, 329
- upcxx::detail::tuple_rvals_get< Tup &, i, Ti >, 333
- upcxx::detail::tuple_rvals_get< Tup &, i, Ti & >, 332
- upcxx::detail::tuple_rvals_get< Tup &, i, Ti && >, 333
- upcxx::detail::tuple_rvals_get< Tup &&, i, Ti >, 331
- upcxx::detail::tuple_rvals_get< Tup &&, i, Ti & >, 330
- upcxx::detail::tuple_rvals_get< Tup &&, i, Ti && >, 331
- upcxx::detail::tuple_rvals_get< Tup const &, i, Ti >, 335
- upcxx::detail::tuple_rvals_get< Tup const &, i, Ti & >, 334
- upcxx::detail::tuple_rvals_get< Tup const &, i, Ti && >, 335
- upcxx::fast_hasher, 100
- upcxx::fast_hashing, 101
- upcxx::function_ref< Ret(Arg...)>, 103
- upcxx::hasher_reflector, 193
- upcxx::mod2n_hashing, 214
- upcxx::nop_function< Ret(Arg...)>, 215
- upcxx::nop_function< void(Arg...)>, 216
- upcxx::packing_pack_reflector, 240
- upcxx::packing_ubound_reflector, 251
- upcxx::packing_unpack_reflector< false >, 253
- upcxx::packing_unpack_reflector< true >, 254
- upcxx::print_reflector, 266
- upcxx::reflection, 270
- upcxx::reflection< std::array< T, n > >, 270
- upcxx::reflection< std::pair< A, B > >, 271
- upcxx::reflection< std::tuple< Ts... > >, 272
- upcxx::reflection_tuple, 272
- upcxx::reflection_tuple< Tup, n, n >, 273
- operator+
 - upcxx, 32
 - upcxx::global_ptr, 183
- operator++
 - upcxx::global_ptr, 183, 184
- operator-
 - upcxx::global_ptr, 184
- operator->
 - upcxx::dist_object, 99
- operator--
 - upcxx::global_ptr, 184
- operator=
 - upcxx::detail::future_impl_mapped::result_lrefs_← function, 275, 276
 - upcxx::detail::future_impl_shref, 165
 - upcxx::function_ref< Ret(Arg...)>, 103
 - upcxx::future1, 107
- operator==
 - upcxx::digest, 90
 - upcxx::global_ptr, 185
- operator |
 - upcxx, 33
- operxn
 - upcxx::completions, 84
- operxn_cx_as_future
 - upcxx, 39
- operxn_cx_as_promise
 - upcxx, 33
- ordblks
 - mallinfo, 204
- os_env
 - upcxx, 34
- os_env.hpp, 416
- overhead_for
 - dl_malloc.c, 382
- owner
 - upcxx::detail::rget_states, 293
- PINUSE_BIT
 - dl_malloc.c, 382
- POSTACTION
 - dl_malloc.c, 382
- PREACTION
 - dl_malloc.c, 383
- PROCEED_ON_ERROR
 - dl_malloc.c, 383
- pack
 - upcxx::commanding, 82
 - upcxx::detail::packing_tuple_each, 248
 - upcxx::detail::packing_tuple_each< n, n, T... >, 250
 - upcxx::packing< bound_function< Fn, B... > >, 219
 - upcxx::packing< std::array< T, n > >, 220
 - upcxx::packing< std::pair< A, B > >, 221
 - upcxx::packing< std::string >, 222
 - upcxx::packing< std::tuple< T... > >, 224
 - upcxx::packing< std::unordered_map< K, V > >, 225
 - upcxx::packing< std::unordered_set< T > >, 226
 - upcxx::packing< std::vector< T > >, 227
 - upcxx::packing_empty< T, false >, 231
 - upcxx::packing_empty< T, true >, 233
 - upcxx::packing_function_pointer, 234
 - upcxx::packing_reflected< T, false >, 241
 - upcxx::packing_reflected< T, true >, 242
 - upcxx::packing_trivial, 246
- packing.cpp, 416

- packing.hpp, 416
- packing_funptr_basis
 - upcxx::detail, 51
- pad_request
 - dl_malloc.c, 382
- page_align
 - dl_malloc.c, 382
- page_size
 - malloc_params, 207
- parcel_layout
 - upcxx::parcel_layout, 256
- parcel_reader
 - upcxx::parcel_reader, 258
- parcel_writer
 - upcxx::parcel_writer, 261
- parent
 - malloc_tree_chunk, 213
- pinuse
 - dl_malloc.c, 382
- pop
 - upcxx::parcel_reader, 259
- pop_char
 - upcxx::parcel_reader, 259
- pop_int8
 - upcxx::parcel_reader, 260
- pop_trivial_aligned
 - upcxx::parcel_reader, 260
- pop_trivial_unaligned
 - upcxx::parcel_reader, 260
- pop_uint8
 - upcxx::parcel_reader, 260
- prev_chunk
 - dl_malloc.c, 383
- prev_foot
 - malloc_chunk, 205
 - malloc_tree_chunk, 213
- print
 - upcxx, 34
- pro
 - upcxx::detail::rget_state_operxn< rget_byref, future_cx< ordinal > >, 283
 - upcxx::detail::rget_state_operxn< rget_byref, promise_cx<> >, 286
 - upcxx::detail::rget_state_operxn< rget_byval< T >, future_cx< ordinal > >, 287
 - upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T > >, 289
 - upcxx::detail::rput_state_here< future_cx< ordinal > >, 313
 - upcxx::detail::rput_state_here< promise_cx<> >, 316
- pro_
 - upcxx::detail::rpc_recipient_after, 304
 - upcxx::promise_cx, 268
- progress
 - upcxx, 34
- progress_level
 - upcxx, 24
- promise_like_t
 - upcxx::detail, 49
- pshm_local_addr2remote
 - upcxx, 34
- pshm_remote_addr2local
 - upcxx, 35
- put
 - upcxx::parcel_writer, 262
- put_char
 - upcxx::parcel_writer, 262
- put_int8
 - upcxx::parcel_writer, 263
- put_trivial_aligned
 - upcxx::parcel_writer, 263
- put_trivial_unaligned
 - upcxx::parcel_writer, 263
- put_uint8
 - upcxx::parcel_writer, 264
- r
 - upcxx::detail::rget_states, 293
 - upcxx::detail::rput_states< rput_byref, S, R, O >, 320
 - upcxx::packing_unpack_reflector< false >, 253
 - upcxx::packing_unpack_reflector< true >, 255
- RELEASE_LOCK
 - dl_malloc.c, 383
- RELEASE_MALLOC_GLOBAL_LOCK
 - dl_malloc.c, 383
- RTCHECK
 - dl_malloc.c, 384
- rank_
 - upcxx::global_ptr, 187
- rank_me
 - upcxx, 35
 - upcxx::backend, 43
- rank_n
 - upcxx, 35
 - upcxx::backend, 43
- raw_ptr_
 - upcxx::global_ptr, 187
- readify
 - upcxx::detail::future_header_result, 151
 - upcxx::detail::future_header_result<>, 155
- ready
 - upcxx::detail::future_impl_mapped, 157
 - upcxx::detail::future_impl_result, 159
 - upcxx::detail::future_impl_result<>, 161
 - upcxx::detail::future_impl_shref, 165
 - upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >, 168
 - upcxx::future1, 108
- ref_n_
 - upcxx::detail::future_header, 140
- reflect
 - reflection.hpp, 418
- reflect_upon
 - upcxx, 35
- reflection.hpp, 418

- reflect, [418](#)
- refs_add
 - upcxx::detail::future_header, [140](#)
 - upcxx::detail::future_header_dependent, [143](#)
- refs_drop
 - upcxx::detail::future_header, [140](#)
 - upcxx::detail::future_header_dependent, [144](#)
- reinterpret_pointer_cast
 - upcxx, [35](#)
 - upcxx::global_ptr, [186](#)
- release_checks
 - malloc_state, [210](#)
- remote
 - upcxx::completions, [84](#)
- remote_cx_as_rpc
 - upcxx, [36](#)
- replace_dv
 - dl_malloc.c, [383](#)
- request2size
 - dl_malloc.c, [384](#)
- requires_upcxx_backend
 - nobsrule, [17](#)
- result
 - upcxx::fast_hasher, [100](#)
 - upcxx::future1, [108](#)
- result_
 - upcxx::detail::future_header, [140](#)
 - upcxx::detail::future_impl_mapped::result_lrefs_↔
function, [276](#)
- result_lrefs_function
 - upcxx::detail::future_dependency< future1<
future_kind_mapped< FuArg, Fn >, T... >>, [122](#)
 - upcxx::detail::future_impl_mapped::result_lrefs_↔
function, [275](#)
- result_lrefs_getter
 - upcxx::detail::future_dependency< future1<
future_kind_mapped< FuArg, Fn >, T... >>, [124](#)
 - upcxx::detail::future_dependency< future1<
future_kind_result >>, [126](#)
 - upcxx::detail::future_dependency< future1<
future_kind_result, T... >>, [127](#)
 - upcxx::detail::future_dependency_when_all_↔
base< future1< future_kind_when_all<
Arg... >, T... >, upcxx::index_sequence<
i... >>, [136](#)
 - upcxx::detail::future_impl_mapped, [157](#)
 - upcxx::detail::future_impl_result, [159](#)
 - upcxx::detail::future_impl_result<>, [161](#)
 - upcxx::detail::future_impl_shref, [165](#)
 - upcxx::detail::future_impl_when_all< std::tuple<
FuArg... >, T... >, [168](#)
- result_moved
 - upcxx::future1, [108](#)
- result_rvals
 - upcxx::detail::future_impl_mapped, [157](#)
 - upcxx::detail::future_impl_result, [159](#)
 - upcxx::detail::future_impl_result<>, [161](#)
 - upcxx::detail::future_impl_shref, [165](#)
 - upcxx::detail::future_impl_when_all< std::tuple<
FuArg... >, T... >, [168](#)
- results
 - upcxx::future1, [108](#)
- results_
 - upcxx::detail::future_dependency< future1<
future_kind_result, T... >>, [128](#)
 - upcxx::detail::future_header_result, [152](#)
 - upcxx::detail::future_impl_result, [160](#)
- results_moved
 - upcxx::future1, [108](#)
- results_of
 - upcxx::detail::future_header_result, [151](#)
 - upcxx::detail::future_header_result<>, [155](#)
- results_type
 - upcxx::future1, [105](#)
- return_type
 - upcxx::detail::apply_futered_as_future< Fn(future1<
Kind, T... >>)>, [62](#)
 - upcxx::detail::apply_tupled_as_future_impl, [65](#)
 - upcxx::detail::apply_tupled_as_future_impl<
future1< Kind, T... >>, [66](#)
 - upcxx::detail::apply_tupled_as_future_impl< void
>, [67](#)
 - upcxx::detail::rget_futures_of, [278](#)
 - upcxx::detail::rget_futures_of< rget_byref, R,
future_cx< 0 >>, [279](#)
 - upcxx::detail::rget_futures_of< rget_byval< T >,
R, future_cx< 0 >>, [281](#)
 - upcxx::detail::rput_futures_of, [306](#)
 - upcxx::detail::rput_futures_of< future_cx< S_ord
>, R, future_cx< O_ord >>, [307](#)
 - upcxx::detail::rput_futures_of< future_cx< S_ord
>, R, O >, [309](#)
 - upcxx::detail::rput_futures_of< S, R, future_cx<
O_ord >>, [311](#)
- rget
 - upcxx, [36](#)
- rget.hpp, [419](#)
- rget_futures_of
 - upcxx::detail::rget_futures_of, [278](#)
 - upcxx::detail::rget_futures_of< rget_byref, R,
future_cx< 0 >>, [279](#)
 - upcxx::detail::rget_futures_of< rget_byval< T >,
R, future_cx< 0 >>, [281](#)
- rget_state_operxn
 - upcxx::detail::rget_state_operxn< rget_byref,
future_cx< ordinal >>, [283](#)
 - upcxx::detail::rget_state_operxn< rget_byref, nil_↔
_cx >, [284](#)
 - upcxx::detail::rget_state_operxn< rget_byref,
promise_cx<>>, [285](#)
 - upcxx::detail::rget_state_operxn< rget_byval< T
>, future_cx< ordinal >>, [286](#)
 - upcxx::detail::rget_state_operxn< rget_byval< T
>, nil_cx >, [288](#)

- upcxx::detail::rget_state_operxn< rget_byval< T >, promise_cx< T >>, 289
- rget_state_remote
 - upcxx::detail::rget_state_remote< nil_cx >, 290
 - upcxx::detail::rget_state_remote< rpc_cx< Fn >>, 291
- rget_states
 - upcxx::detail::rget_states, 293
- rma_get
 - upcxx::backend, 42
- rma_get_cb_impl
 - upcxx::backend::gasnet1_seq::rma_get_cb_impl, 296
- rma_get_cb_wstate
 - upcxx::backend::rma_get_cb_wstate, 298
- rma_put
 - upcxx::backend, 42
- rma_put_cb_impl
 - upcxx::backend::gasnet1_seq::rma_put_cb_impl, 300
- rma_put_cb_wstate
 - upcxx::backend::rma_put_cb_wstate, 301
- rpc
 - upcxx, 36, 37
- rpc.hpp, 420
- rpc_cx
 - upcxx::rpc_cx, 302
- rpc_ff
 - upcxx, 37
- rput
 - upcxx, 37
- rput.hpp, 420
- rput_futures_of
 - upcxx::detail::rput_futures_of, 306
 - upcxx::detail::rput_futures_of< future_cx< S_ord >, R, future_cx< O_ord >>, 307
 - upcxx::detail::rput_futures_of< future_cx< S_ord >, R, O >, 309
 - upcxx::detail::rput_futures_of< S, R, future_cx< O_ord >>, 311
- rput_state_here
 - upcxx::detail::rput_state_here< future_cx< ordinal >>, 313
 - upcxx::detail::rput_state_here< nil_cx >, 314
 - upcxx::detail::rput_state_here< promise_cx<>>, 315
- rput_state_remote
 - upcxx::detail::rput_state_remote< nil_cx >, 317
 - upcxx::detail::rput_state_remote< rpc_cx< Fn >>, 318
- rput_states
 - upcxx::detail::rput_states< rput_byref, S, R, O >, 320
 - upcxx::detail::rput_states< rput_byval< T >, S, R, O >, 321
- s
 - upcxx::detail::rput_states< rput_byref, S, R, O >, 320
 - upcxx::fast_hasher, 100
 - s_
 - upcxx::detail::rput_futures_of< future_cx< S_ord >, R, future_cx< O_ord >>, 308
 - SIX_SIZE_T_SIZES
 - dl_malloc.c, 387
 - SIZE_T_BITSIZE
 - dl_malloc.c, 387
 - SIZE_T_FOUR
 - dl_malloc.c, 387
 - SIZE_T_ONE
 - dl_malloc.c, 387
 - SIZE_T_SIZE
 - dl_malloc.c, 387
 - SIZE_T_TWO
 - dl_malloc.c, 387
 - SIZE_T_ZERO
 - dl_malloc.c, 388
 - SMALLBIN_SHIFT
 - dl_malloc.c, 388
 - SMALLBIN_WIDTH
 - dl_malloc.c, 388
 - STRUCT_MALLINFO_DECLARED
 - dl_malloc.c, 389
 - dl_malloc.h, 402
 - SYS_ALLOC_PADDING
 - dl_malloc.c, 389
 - same_or_left_bits
 - dl_malloc.c, 384
 - sbinptr
 - dl_malloc.c, 395
 - seg
 - malloc_state, 211
 - segment_holds
 - dl_malloc.c, 384
 - send_am
 - upcxx::backend, 42
 - send_am_eager_queued
 - upcxx::backend::gasnet1_seq, 44
 - send_am_eager_restricted
 - upcxx::backend::gasnet1_seq, 44
 - send_am_rdzv
 - upcxx::backend::gasnet1_seq, 44
 - send_am_restricted
 - upcxx::backend::gasnet1_seq, 44
 - set_flag4
 - dl_malloc.c, 385
 - set_foot
 - dl_malloc.c, 385
 - set_free_with_pinuse
 - dl_malloc.c, 385
 - set_inuse
 - dl_malloc.c, 385
 - set_inuse_and_pinuse
 - dl_malloc.c, 385
 - set_lock
 - dl_malloc.c, 386
 - set_size_and_pinuse_of_free_chunk

- dl_malloc.c, 386
- set_size_and_pinuse_of_inuse_chunk
 - dl_malloc.c, 386
- sflags
 - malloc_segment, 208
- should_trim
 - dl_malloc.c, 386
- size
 - malloc_segment, 208
 - upcxx::parcel_layout, 257
 - upcxx::parcel_writer, 264
- size_aligned
 - upcxx::parcel_layout, 257
- size_ubound
 - upcxx::commanding, 83
 - upcxx::detail::packing_tuple_each, 249
 - upcxx::detail::packing_tuple_each< n, n, T... >, 250
 - upcxx::packing< bound_function< Fn, B... > >, 219
 - upcxx::packing< std::array< T, n > >, 220
 - upcxx::packing< std::pair< A, B > >, 221
 - upcxx::packing< std::string >, 223
 - upcxx::packing< std::tuple< T... > >, 224
 - upcxx::packing< std::unordered_map< K, V > >, 225
 - upcxx::packing< std::unordered_set< T > >, 226
 - upcxx::packing< std::vector< T > >, 228
 - upcxx::packing_empty< T, false >, 231
 - upcxx::packing_empty< T, true >, 233
 - upcxx::packing_function_pointer, 234
 - upcxx::packing_not_supported, 237
 - upcxx::packing_reflected< T, false >, 241
 - upcxx::packing_reflected< T, true >, 243
 - upcxx::packing_trivial, 246
- small_index
 - dl_malloc.c, 388
- small_index2size
 - dl_malloc.c, 388
- smallbin_at
 - dl_malloc.c, 388
- smallbins
 - malloc_state, 211
- smallmap
 - malloc_state, 211
- smallmap_is_marked
 - dl_malloc.c, 389
- smbkls
 - mallinfo, 204
- source
 - upcxx::completions, 84
- source_cx_as_future
 - upcxx, 39
- source_cx_as_promise
 - upcxx, 38
- src_cx
 - upcxx::backend::gasnet1_seq::rma_put_cb_impl, 300
- state
 - upcxx::backend::rma_get_cb_wstate, 298
 - upcxx::backend::rma_put_cb_wstate, 301
- status_
 - upcxx::detail::future_header, 141
- status_active
 - upcxx::detail::future_header, 141
- status_proxying
 - upcxx::detail::future_header, 141
- status_proxying_active
 - upcxx::detail::future_header, 141
- status_ready
 - upcxx::detail::future_header, 141
- status_results_no
 - upcxx::detail::future_header_result, 152
- status_results_yes
 - upcxx::detail::future_header_result, 152
- std, 17
- std::greater< global_ptr< T > >
 - upcxx::global_ptr, 186
- std::greater< upcxx::global_ptr< T > >, 188
 - operator(), 188
- std::greater_equal< global_ptr< T > >
 - upcxx::global_ptr, 186
- std::greater_equal< upcxx::global_ptr< T > >, 189
 - operator(), 189
- std::hash< global_ptr< T > >
 - upcxx::global_ptr, 187
- std::hash< upcxx::digest >, 190
 - operator(), 190
- std::hash< upcxx::dist_id< T > >, 190
 - operator(), 191
- std::hash< upcxx::global_ptr< T > >, 191
 - operator(), 191
- std::less< global_ptr< T > >
 - upcxx::global_ptr, 187
- std::less< upcxx::global_ptr< T > >, 197
 - operator(), 198
- std::less_equal< global_ptr< T > >
 - upcxx::global_ptr, 187
- std::less_equal< upcxx::global_ptr< T > >, 198
 - operator(), 198
- steal_header
 - upcxx::detail::future_impl_mapped, 157
 - upcxx::detail::future_impl_result, 160
 - upcxx::detail::future_impl_result<>, 161
 - upcxx::detail::future_impl_shref, 166
 - upcxx::detail::future_impl_when_all< std::tuple< FuArg... >, T... >, 169
- storage_
 - upcxx::detail::future_body, 111
- subject
 - upcxx::print_proxy, 265
- suc
 - upcxx::detail::future_header::dependency_link, 88
- sucs_head_
 - upcxx::detail::future_header, 141
- sucs_next

- upcxx::detail::future_header::dependency_link, 88
- TOP_FOOT_SIZE
 - dl_malloc.c, 389
- TREEBIN_SHIFT
 - dl_malloc.c, 389
- TRY_LOCK
 - dl_malloc.c, 390
- TWO_SIZE_T_SIZES
 - dl_malloc.c, 390
- target
 - upcxx::detail::rput_states< rput_byref, S, R, O >, 320
- tbinptr
 - dl_malloc.c, 395
- tchunk
 - dl_malloc.c, 396
- tchunkptr
 - dl_malloc.c, 396
- the_always
 - upcxx::detail::future_header_result<>, 155
- the_nil
 - upcxx::detail::future_header, 142
- then
 - upcxx::future1, 109
- then_pure
 - upcxx::future1, 109
- this_t
 - upcxx::detail::future_dependency_when_all_↔ base< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... >>, 136
- Ti
 - upcxx::detail::packing_tuple_each, 248
- to_future
 - upcxx, 38
- top
 - malloc_state, 211
- topsize
 - malloc_state, 211
- treebin_at
 - dl_malloc.c, 389
- treebins
 - malloc_state, 211
- treemap
 - malloc_state, 212
- treemap_is_marked
 - dl_malloc.c, 390
- trim_check
 - malloc_state, 212
- trim_threshold
 - malloc_params, 207
- tuple_rvals
 - upcxx, 38
 - upcxx::detail, 51
- tuple_types_into_t
 - upcxx, 23
- tupled_impl
 - upcxx::detail::apply_futered_as_future< Fn(future1< Kind, T... >>)>, 62
- type
 - upcxx::add_lref_if_nonref, 56
 - upcxx::add_lref_if_nonref< T & >, 56
 - upcxx::add_lref_if_nonref< T && >, 57
 - upcxx::decay_tupled< std::tuple< T... > >, 87
 - upcxx::detail::disjoin_cx< A, nil_cx, B_ord_bump >, 92
 - upcxx::detail::disjoin_cx< nil_cx, B, B_ord_bump >, 93
 - upcxx::detail::disjoin_cx< nil_cx, future_cx< B_↔ ord_old >, B_ord_bump >, 94
 - upcxx::detail::disjoin_cx< nil_cx, nil_cx, B_ord0 >, 95
 - upcxx::detail::future_from_tuple< Kind, std::tuple< T... > >, 138
 - upcxx::detail::make_index_sequence< 0, s... >, 202
 - upcxx::detail::promise_like< future1< Kind, T... > >, 269
 - upcxx::detail::rpc_return, 304
 - upcxx::trait_all<>, 323
 - upcxx::trait_any<>, 325
 - upcxx::tuple_types_into< std::tuple< T... >, Into >, 336
- UPCXX_ASSERT_1
 - diagnostic.hpp, 348
- UPCXX_ASSERT_2
 - diagnostic.hpp, 348
- UPCXX_ASSERT_ALWAYS
 - diagnostic.hpp, 349
- UPCXX_ASSERT_DISPATCH
 - diagnostic.hpp, 349
- UPCXX_ASSERT
 - diagnostic.hpp, 348
- UPCXX_INVOKE_UB
 - diagnostic.hpp, 349
- USAGE_ERROR_ACTION
 - dl_malloc.c, 392
- USE_BUILTIN_FFS
 - dl_malloc.c, 392
- USE_DEV_RANDOM
 - dl_malloc.c, 392
- USE_LOCK_BIT
 - dl_malloc.c, 392
- USE_LOCKS
 - dl_malloc.c, 393
- USE_MMAP_BIT
 - dl_malloc.c, 393
- USE_NONCONTIGUOUS_BIT
 - dl_malloc.c, 393
- USE_SPIN_LOCKS
 - dl_malloc.c, 394
- ub
 - upcxx::packing_ubound_reflector, 252
- uinrank_t
 - upcxx, 24

- unlink
 - upcxx::detail::future_header::dependency_link, 87
- unlink_
 - upcxx::detail::future_dependency_shref_base< false >, 131
 - upcxx::detail::future_dependency_shref_base< true >, 133
- unlink_chunk
 - dl_malloc.c, 390
- unlink_first_small_chunk
 - dl_malloc.c, 390
- unlink_large_chunk
 - dl_malloc.c, 391
- unlink_small_chunk
 - dl_malloc.c, 391
- unpack
 - upcxx::packing< bound_function< Fn, B... > >, 219
 - upcxx::packing< std::array< T, n > >, 220
 - upcxx::packing< std::pair< A, B > >, 222
 - upcxx::packing< std::string >, 223
 - upcxx::packing< std::tuple< T... > >, 224
 - upcxx::packing< std::unordered_map< K, V > >, 225
 - upcxx::packing< std::unordered_set< T > >, 227
 - upcxx::packing< std::vector< T > >, 228
 - upcxx::packing_empty< T, false >, 231
 - upcxx::packing_empty< T, true >, 233
 - upcxx::packing_function_pointer, 235
 - upcxx::packing_reflected< T, false >, 242
 - upcxx::packing_reflected< T, true >, 243
 - upcxx::packing_trivial, 247
- unpack_into
 - upcxx::detail::packing_tuple_each, 249
 - upcxx::detail::packing_tuple_each< n, n, T... >, 250
- uordblks
 - mallinfo, 204
- upcxx, 18
 - allocate, 24
 - allreduce, 25
 - apply_tupled, 25
 - apply_tupled_as_future, 25
 - assert_failed, 25
 - atomic_fetch_add, 25
 - atomic_get, 26
 - atomic_put, 26
 - barrier, 26
 - bind, 26
 - bind_last, 27
 - broadcast, 27
 - command_execute, 27
 - command_pack, 27, 28
 - command_size_ubound, 28
 - constant, 28
 - dbgbrk, 28
 - dbgbrk_spin, 39
 - dbgbrk_spin_init, 39
 - deallocate, 29
 - delete_, 29
 - delete_array, 29
 - fast_hash, 29
 - finalize, 30
 - future, 23
 - get_or_void, 30
 - init, 30
 - internal, 24
 - inrank_t, 23
 - is_memory_shared_with, 30
 - make_future, 30
 - make_index_sequence, 23
 - mod2n_hash, 31
 - new_, 31
 - new_array, 31
 - nop, 32
 - operator<<, 32
 - operator>>, 33
 - operator>>=, 33
 - operator+, 32
 - operator|, 33
 - operxn_cx_as_future, 39
 - operxn_cx_as_promise, 33
 - os_env, 34
 - print, 34
 - progress, 34
 - progress_level, 24
 - pshm_local_addr2remote, 34
 - pshm_remote_addr2local, 35
 - rank_me, 35
 - rank_n, 35
 - reflect_upon, 35
 - reinterpret_pointer_cast, 35
 - remote_cx_as_rpc, 36
 - rget, 36
 - rpc, 36, 37
 - rpc_ff, 37
 - rput, 37
 - source_cx_as_future, 39
 - source_cx_as_promise, 38
 - to_future, 38
 - tuple_rvals, 38
 - tuple_types_into_t, 23
 - uinrank_t, 24
 - user, 24
 - wait, 38
 - when_all, 39
- upcxx.cpp, 421
- upcxx.hpp, 421
- upcxx::add_lref_if_nonref
 - type, 56
- upcxx::add_lref_if_nonref< T >, 55
- upcxx::add_lref_if_nonref< T & >, 56
 - type, 56
- upcxx::add_lref_if_nonref< T && >, 57
 - type, 57
- upcxx::backend, 40

- during_level, 40, 41
- during_user, 41
- make_rma_get_cb, 42
- make_rma_put_cb, 42
- rank_me, 43
- rank_n, 43
- rma_get, 42
- rma_put, 42
- send_am, 42
- upcxx::backend::gasnet1_seq, 43
 - am_size_rdzv_cutover, 45
 - in_user_progress_, 45
 - send_am_eager_queued, 44
 - send_am_eager_restricted, 44
 - send_am_rdzv, 44
 - send_am_restricted, 44
 - user_actions_head_, 45
 - user_actions_tailp_, 45
- upcxx::backend::gasnet1_seq::action, 53
 - fire_and_delete, 53
 - next_, 54
- upcxx::backend::gasnet1_seq::action_impl
 - action_impl, 54
 - fire_and_delete, 55
 - fn, 55
- upcxx::backend::gasnet1_seq::action_impl< Fn >, 54
- upcxx::backend::gasnet1_seq::rma_cb, 294
 - fire_and_delete, 294
 - handle, 295
 - next, 295
- upcxx::backend::gasnet1_seq::rma_get_cb_impl
 - op_cx, 297
 - rma_get_cb_impl, 296
- upcxx::backend::gasnet1_seq::rma_get_cb_impl< State, OpCx >, 296
- upcxx::backend::gasnet1_seq::rma_put_cb_impl
 - op_cx, 300
 - rma_put_cb_impl, 300
 - src_cx, 300
- upcxx::backend::gasnet1_seq::rma_put_cb_impl< State, SrcCx, OpCx >, 299
- upcxx::backend::rma_get_cb, 295
- upcxx::backend::rma_get_cb_wstate
 - rma_get_cb_wstate, 298
 - state, 298
- upcxx::backend::rma_get_cb_wstate< State >, 297
- upcxx::backend::rma_put_cb, 298
- upcxx::backend::rma_put_cb_wstate
 - rma_put_cb_wstate, 301
 - state, 301
- upcxx::backend::rma_put_cb_wstate< State >, 301
- upcxx::binding
 - is_trivial, 71
 - off_wire, 70
 - off_wire_type, 70
 - on_wire, 70
 - on_wire_type, 70
- upcxx::binding< dist_object< T > & >, 71
 - off_wire, 72
 - off_wire_type, 72
 - on_wire, 72
 - on_wire_type, 72
- upcxx::binding< dist_object< T > && >, 73
- upcxx::binding< T >, 69
- upcxx::binding< T & >, 73
- upcxx::binding< T && >, 74
- upcxx::binding< T const >, 74
- upcxx::binding< T volatile >, 75
- upcxx::binding_is_trivial
 - value, 75
- upcxx::binding_is_trivial< T, std::integral_constant< bool, binding< T >::is_trivial > >, 76
 - value, 76
- upcxx::binding_is_trivial< T, trivial >, 75
- upcxx::bound_function
 - base_type, 77
 - bound_function, 78
 - operator(), 78
- upcxx::bound_function< Fn, B >, 77
- upcxx::commanding
 - execute, 82
 - pack, 82
 - size_ubound, 83
- upcxx::commanding< Fn >, 82
- upcxx::completions
 - future_n, 84
 - opern, 84
 - remote, 84
 - source, 84
- upcxx::completions< SourceCx, RemoteCx, OpnCx >, 83
- upcxx::constant_function
 - constant_function, 85
 - operator(), 85
 - value_, 86
- upcxx::constant_function< T >, 85
- upcxx::decay_tupled< std::tuple< T... > >, 86
 - type, 87
- upcxx::decay_tupled< Tup >, 86
- upcxx::detail, 46
 - apply_futered_as_future_return_t, 49
 - apply_tupled, 49
 - bind_last, 50
 - broadcast_receive, 50
 - command_executor, 50
 - dist_master_id_bump, 52
 - dist_master_promises, 52
 - dist_promise, 50
 - future_from_tuple_t, 49
 - new_, 51
 - new_array, 51
 - packing_funptr_basis, 51
 - promise_like_t, 49
 - tuple_rvals, 51
 - when_all_return_t, 49
- upcxx::detail::allreduce_state

- accum, [58](#)
- answer, [59](#)
- broadcast, [58](#)
- contributed, [58](#)
- incoming, [59](#)
- op, [59](#)
- upcxx::detail::allreduce_state< T, Op >, [57](#)
- upcxx::detail::apply_futered_as_future< Fn(future1< Kind, T... >)>, [61](#)
 - operator(), [62](#)
 - return_type, [62](#)
 - tupled_impl, [62](#)
- upcxx::detail::apply_tupled_as_future< Fn, ArgTup >, [63](#)
- upcxx::detail::apply_tupled_as_future_help< Fn, Arg← Tup, ArgTupDecayed >, [63](#)
- upcxx::detail::apply_tupled_as_future_help< Fn, Arg← Tup, std::tuple< T... > >, [64](#)
- upcxx::detail::apply_tupled_as_future_impl
 - operator(), [65](#)
 - return_type, [65](#)
- upcxx::detail::apply_tupled_as_future_impl< future1< Kind, T... > >, [65](#)
 - operator(), [66](#)
 - return_type, [66](#)
- upcxx::detail::apply_tupled_as_future_impl< Return >, [64](#)
- upcxx::detail::apply_tupled_as_future_impl< void >, [67](#)
 - operator(), [67](#)
 - return_type, [67](#)
- upcxx::detail::bind
 - operator(), [68](#)
- upcxx::detail::bind< bound_function< Fn0, B0... > >, [68](#)
 - operator(), [69](#)
- upcxx::detail::bind< FnDecayed >, [68](#)
- upcxx::detail::bound_function_base< Fn, BndTup, all← _trivial >, [78](#)
- upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >, [79](#)
 - apply_, [79](#)
 - b_, [80](#)
 - fn_, [80](#)
- upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator
 - a, [60](#)
 - apply_, [60](#)
 - b, [61](#)
 - fn, [61](#)
 - operator(), [60](#)
- upcxx::detail::bound_function_base< Fn, std::tuple< B... >, false >::applicator< Arg >, [59](#)
- upcxx::detail::bound_function_base< Fn, std::tuple< B... >, true >, [80](#)
 - apply_, [81](#)
 - b_, [81](#)
 - fn_, [81](#)
- upcxx::detail::disjoin_cx< A, B, B_ord_bump >, [91](#)
- upcxx::detail::disjoin_cx< A, nil_cx, B_ord_bump >, [92](#)
 - operator(), [92](#)
 - type, [92](#)
- upcxx::detail::disjoin_cx< nil_cx, B, B_ord_bump >, [93](#)
 - operator(), [93](#)
 - type, [93](#)
- upcxx::detail::disjoin_cx< nil_cx, future_cx< B_ord_old >, B_ord_bump >, [94](#)
 - operator(), [94](#)
 - type, [94](#)
- upcxx::detail::disjoin_cx< nil_cx, nil_cx, B_ord0 >, [95](#)
 - operator(), [95](#)
 - type, [95](#)
- upcxx::detail::future_body, [110](#)
 - destruct_early, [110](#)
 - future_body, [110](#)
 - leave_active, [110](#)
 - storage_, [111](#)
- upcxx::detail::future_body_proxy
 - destruct_early, [112](#)
 - future_body_proxy, [112](#)
- upcxx::detail::future_body_proxy< T >, [111](#)
- upcxx::detail::future_body_proxy_, [112](#)
 - future_body_proxy_, [113](#)
 - leave_active, [113](#)
 - link_, [113](#)
- upcxx::detail::future_body_pure< FuArg >, [114](#)
- upcxx::detail::future_body_pure< future1< Kind, T... > >, [114](#)
 - dep_, [116](#)
 - destruct_early, [115](#)
 - future_body_pure, [115](#)
 - leave_active, [115](#)
- upcxx::detail::future_body_then
 - dep_, [117](#)
 - fn_, [117](#)
 - future_body_then, [117](#)
 - leave_active, [117](#)
- upcxx::detail::future_body_then< FuArg, Fn >, [116](#)
- upcxx::detail::future_body_then_base, [118](#)
 - future_body_then_base, [118](#)
 - leave_active_into_proxy, [119](#)
- upcxx::detail::future_body_then_pure
 - dep_, [121](#)
 - destruct_early, [120](#)
 - fn_, [121](#)
 - future_body_then_pure, [120](#)
 - leave_active, [120](#)
- upcxx::detail::future_body_then_pure< FuArg, Fn >, [119](#)
- upcxx::detail::future_dependency< FuArg >, [121](#)
- upcxx::detail::future_dependency< future1< future← kind_mapped< FuArg, Fn >, T... > >, [122](#)
 - cleanup_early, [123](#)
 - cleanup_ready, [123](#)
 - cleanup_ready_get_header, [123](#)
 - dep_, [124](#)
 - fn_, [124](#)

- future_dependency, 123
- result_lrefs_function, 122
- result_lrefs_getter, 124
- upcxx::detail::future_dependency< future1< future_↵
kind_result > >, 124
- cleanup_early, 125
- cleanup_ready, 125
- cleanup_ready_get_header, 125
- future_dependency, 125
- result_lrefs_getter, 126
- upcxx::detail::future_dependency< future1< future_↵
kind_result, T... > >, 126
- cleanup_early, 127
- cleanup_ready, 127
- cleanup_ready_get_header, 127
- future_dependency, 127
- result_lrefs_getter, 127
- results_, 128
- upcxx::detail::future_dependency< future1< future_↵
kind_shref< HeaderOps >, T... > >, 128
- future_dependency, 129
- upcxx::detail::future_dependency< future1< future_↵
kind_when_all< Arg... >, T... > >, 129
- future_dependency, 129
- upcxx::detail::future_dependency_shref_base< false >, 130
- future_dependency_shref_base, 131
- header_, 131
- link_, 132
- unlink_, 131
- upcxx::detail::future_dependency_shref_base< is_↵
trivially_ready_result >, 130
- upcxx::detail::future_dependency_shref_base< true >, 132
- future_dependency_shref_base, 132
- hdr_, 133
- header_, 133
- unlink_, 133
- upcxx::detail::future_dependency_when_all_arg
- dep_, 134
- future_dependency_when_all_arg, 134
- upcxx::detail::future_dependency_when_all_arg< i, Arg
>, 134
- upcxx::detail::future_dependency_when_all_base< AllArg, lxSeq >, 135
- upcxx::detail::future_dependency_when_all_base< future1< future_kind_when_all< Arg... >, T... >, upcxx::index_sequence< i... > >, 135
- future_dependency_when_all_base, 136
- result_lrefs_getter, 136
- this_t, 136
- upcxx::detail::future_from_tuple< Kind, std::tuple< T... > >, 137
- type, 138
- upcxx::detail::future_from_tuple< Kind, Tup >, 137
- upcxx::detail::future_header, 138
- body_, 140
- drop_for_proxied, 139
- drop_for_result, 139
- enter_ready, 139
- ref_n_, 140
- refs_add, 140
- refs_drop, 140
- result_, 140
- status_, 141
- status_active, 141
- status_proxying, 141
- status_proxying_active, 141
- status_ready, 141
- sucs_head_, 141
- the_nil, 142
- upcxx::detail::future_header::dependency_link, 87
- dep, 88
- suc, 88
- sucs_next, 88
- unlink, 87
- upcxx::detail::future_header_dependent, 142
- active_next_, 144
- enter_proxying, 143
- entered_active, 143
- future_header_dependent, 143
- refs_add, 143
- refs_drop, 144
- upcxx::detail::future_header_ops_general, 144
- decref_header, 145
- delete_header, 145
- is_trivially_ready_result, 145
- upcxx::detail::future_header_ops_result, 146
- decref_header, 146
- delete_header, 147
- is_trivially_ready_result, 147
- upcxx::detail::future_header_ops_result_ready, 147
- decref_header, 148
- delete_header, 148
- is_trivially_ready_result, 148
- upcxx::detail::future_header_result
- construct_results, 150
- delete_me, 151
- delete_me_ready, 151
- future_header_result, 150
- readify, 151
- results_, 152
- results_of, 151
- status_results_no, 152
- status_results_yes, 152
- upcxx::detail::future_header_result< T >, 149
- upcxx::detail::future_header_result<>, 152
- construct_results, 154
- delete_me, 154
- delete_me_ready, 154
- future_header_result, 154
- readify, 155
- results_of, 155
- the_always, 155
- upcxx::detail::future_impl_mapped
- arg_, 157

- fn_, 158
- future_impl_mapped, 156
- header_ops, 156
- ready, 157
- result_lrefs_getter, 157
- result_rvals, 157
- steal_header, 157
- upcxx::detail::future_impl_mapped< FuArg, Fn, T >, 155
- upcxx::detail::future_impl_mapped< FuArg, Fn, T >←
:result_lrefs_function, 274
- upcxx::detail::future_impl_mapped::result_lrefs_function
 - fn_return_getter_t, 276
 - fn_return_t, 274
 - getter_, 276
 - operator(), 275
 - operator=, 275, 276
 - result_, 276
 - result_lrefs_function, 275
- upcxx::detail::future_impl_result
 - future_impl_result, 159
 - header_ops, 159
 - ready, 159
 - result_lrefs_getter, 159
 - result_rvals, 159
 - results_, 160
 - steal_header, 160
- upcxx::detail::future_impl_result< T >, 158
- upcxx::detail::future_impl_result<>, 160
 - header_ops, 161
 - ready, 161
 - result_lrefs_getter, 161
 - result_rvals, 161
 - steal_header, 161
- upcxx::detail::future_impl_shref
 - ~future_impl_shref, 164
 - future_impl_shref, 163, 164
 - hdr_, 166
 - header_ops, 163
 - operator=, 165
 - ready, 165
 - result_lrefs_getter, 165
 - result_rvals, 166
 - steal_header, 166
- upcxx::detail::future_impl_shref< HeaderOps, T >, 162
- upcxx::detail::future_impl_when_all< ArgTuple, T >, 166
- upcxx::detail::future_impl_when_all< std::tuple< Fu←
Arg... >, T... >, 167
 - args_, 169
 - future_impl_when_all, 168
 - header_ops, 167
 - ready, 168
 - result_lrefs_getter, 168
 - result_rvals, 168
 - steal_header, 169
- upcxx::detail::future_kind_mapped
 - with_types, 173
- upcxx::detail::future_kind_mapped< FuArg, Fn >, 173
- upcxx::detail::future_kind_result, 174
 - with_types, 174
- upcxx::detail::future_kind_shref
 - with_types, 175
- upcxx::detail::future_kind_shref< HeaderOps >, 174
- upcxx::detail::future_kind_when_all
 - with_types, 175
- upcxx::detail::future_kind_when_all< FuArg >, 175
- upcxx::detail::future_then< Arg, Fn, FnRet, arg_trivial
>, 176
- upcxx::detail::future_then< Arg, Fn, future1< FnRet←
Kind, FnRetT... >, false >, 176
 - operator(), 176
- upcxx::detail::future_then< Arg, Fn, future1< FnRet←
Kind, FnRetT... >, true >, 177
 - operator(), 177
- upcxx::detail::future_then_pure< Arg, Fn, FnRet, arg←
_trivial, fnret_trivial >, 178
- upcxx::detail::future_then_pure< Arg, Fn, future1<
FnRetKind, FnRetT... >, false, false >, 178
 - operator(), 178
- upcxx::detail::future_then_pure< Arg, Fn, future1<
FnRetKind, FnRetT... >, false, true >, 179
 - operator(), 179
- upcxx::detail::future_then_pure< Arg, Fn, future1<
FnRetKind, FnRetT... >, true, fnret_trivial >, 180
 - operator(), 180
- upcxx::detail::global_ptr_ctor_internal, 188
- upcxx::detail::is_future_cx< Cx >, 196
- upcxx::detail::is_future_cx< future_cx< ordinal > >, 197
- upcxx::detail::make_future< T >, 199
- upcxx::detail::make_future_< false, T... >, 200
 - operator(), 200
- upcxx::detail::make_future_< trivial, T >, 199
- upcxx::detail::make_future_< true, T... >, 200
 - operator(), 201
- upcxx::detail::make_index_sequence< 0, s... >, 201
 - type, 202
- upcxx::detail::make_index_sequence< n, s >, 201
- upcxx::detail::os_env_parse
 - operator(), 217
- upcxx::detail::os_env_parse< std::string >, 217
 - operator(), 218
- upcxx::detail::os_env_parse< T >, 217
- upcxx::detail::packing_tuple_each
 - destruct, 248
 - pack, 248
 - size_ubound, 249
 - Ti, 248
 - unpack_into, 249
- upcxx::detail::packing_tuple_each< n, i, T >, 247
- upcxx::detail::packing_tuple_each< n, n, T... >, 249
 - destruct, 250
 - pack, 250
 - size_ubound, 250

unpack_into, 250
 upcxx::detail::promise_like< Fu >, 268
 upcxx::detail::promise_like< future1< Kind, T... > >, 268
 type, 269
 upcxx::detail::rget_byval< T >, 277
 upcxx::detail::rget_futures_of
 operator(), 278
 return_type, 278
 rget_futures_of, 278
 upcxx::detail::rget_futures_of< Mode, R, O >, 277
 upcxx::detail::rget_futures_of< rget_byref, R, future_←
 cx< 0 > >, 278
 fut_, 280
 operator(), 280
 return_type, 279
 rget_futures_of, 279
 upcxx::detail::rget_futures_of< rget_byval< T >, R,
 future_cx< 0 > >, 280
 fut_, 282
 operator(), 281
 return_type, 281
 rget_futures_of, 281
 upcxx::detail::rget_state_operxn< Mode, Cx >, 282
 upcxx::detail::rget_state_operxn< rget_byref, future_←
 cx< ordinal > >, 282
 completed, 283
 pro, 283
 rget_state_operxn, 283
 upcxx::detail::rget_state_operxn< rget_byref, nil_cx >, 284
 completed, 284
 rget_state_operxn, 284
 upcxx::detail::rget_state_operxn< rget_byref, promise_←
 _cx<> >, 285
 completed, 285
 pro, 286
 rget_state_operxn, 285
 upcxx::detail::rget_state_operxn< rget_byval< T >,
 future_cx< ordinal > >, 286
 completed, 287
 pro, 287
 rget_state_operxn, 286
 upcxx::detail::rget_state_operxn< rget_byval< T >,
 nil_cx >, 287
 completed, 288
 rget_state_operxn, 288
 upcxx::detail::rget_state_operxn< rget_byval< T >,
 promise_cx< T > >, 288
 completed, 289
 pro, 289
 rget_state_operxn, 289
 upcxx::detail::rget_state_remote< Cx >, 290
 upcxx::detail::rget_state_remote< nil_cx >, 290
 completed, 291
 rget_state_remote, 290
 upcxx::detail::rget_state_remote< rpc_cx< Fn > >, 291
 completed, 292
 fn, 292
 rget_state_remote, 291
 upcxx::detail::rget_states
 o, 293
 owner, 293
 r, 293
 rget_states, 293
 upcxx::detail::rget_states< Mode, R, O >, 292
 upcxx::detail::rpc_recipient_after
 initiator_, 304
 operator(), 303
 pro_, 304
 upcxx::detail::rpc_recipient_after< Pro >, 303
 upcxx::detail::rpc_return
 type, 304
 upcxx::detail::rpc_return< ValidType, Fn, Args >, 304
 upcxx::detail::rput_byval< T >, 305
 upcxx::detail::rput_futures_of
 operator(), 306
 return_type, 306
 rput_futures_of, 306
 upcxx::detail::rput_futures_of< future_cx< S_ord >, R,
 future_cx< O_ord > >, 307
 o_, 308
 operator(), 308
 return_type, 307
 rput_futures_of, 307
 s_, 308
 upcxx::detail::rput_futures_of< future_cx< S_ord >, R,
 O >, 309
 fut_, 310
 operator(), 310
 return_type, 309
 rput_futures_of, 309
 upcxx::detail::rput_futures_of< S, R, future_cx< O_ord
 > >, 310
 fut_, 312
 operator(), 311
 return_type, 311
 rput_futures_of, 311
 upcxx::detail::rput_futures_of< S, R, O >, 305
 upcxx::detail::rput_state_here< Cx >, 312
 upcxx::detail::rput_state_here< future_cx< ordinal >
 >, 312
 completed, 313
 pro, 313
 rput_state_here, 313
 upcxx::detail::rput_state_here< nil_cx >, 314
 completed, 314
 rput_state_here, 314
 upcxx::detail::rput_state_here< promise_cx<> >, 315
 completed, 315
 pro, 316
 rput_state_here, 315
 upcxx::detail::rput_state_remote< Cx >, 316
 upcxx::detail::rput_state_remote< nil_cx >, 316
 completed, 317

- rput_state_remote, 317
- upcxx::detail::rput_state_remote< rpc_cx< Fn > >, 317
 - completed, 318
 - fn, 318
 - rput_state_remote, 318
- upcxx::detail::rput_states< Mode, S, R, O >, 319
- upcxx::detail::rput_states< rput_byref, S, R, O >, 319
 - o, 320
 - r, 320
 - rput_states, 320
 - s, 320
 - target, 320
- upcxx::detail::rput_states< rput_byval< T >, S, R, O >, 321
 - buffer, 322
 - rput_states, 321
- upcxx::detail::tuple_get_or_void
 - operator(), 328
- upcxx::detail::tuple_get_or_void< i, TupRef, false >, 329
 - operator(), 329
- upcxx::detail::tuple_get_or_void< i, TupRef, in_range >, 328
- upcxx::detail::tuple_rvals_get< i, Tup >, 329
- upcxx::detail::tuple_rvals_get< Tup &, i, Ti >, 333
 - operator(), 333
- upcxx::detail::tuple_rvals_get< Tup &, i, Ti & >, 332
 - operator(), 332
- upcxx::detail::tuple_rvals_get< Tup &, i, Ti && >, 332
 - operator(), 333
- upcxx::detail::tuple_rvals_get< Tup &&, i, Ti >, 331
 - operator(), 331
- upcxx::detail::tuple_rvals_get< Tup &&, i, Ti & >, 330
 - operator(), 330
- upcxx::detail::tuple_rvals_get< Tup &&, i, Ti && >, 330
 - operator(), 331
- upcxx::detail::tuple_rvals_get< Tup const &, i, Ti >, 335
 - operator(), 335
- upcxx::detail::tuple_rvals_get< Tup const &, i, Ti & >, 334
 - operator(), 334
- upcxx::detail::tuple_rvals_get< Tup const &, i, Ti && >, 334
 - operator(), 335
- upcxx::digest, 88
 - eat, 89
 - operator!=, 90
 - operator<, 90
 - operator<=, 90
 - operator>, 90
 - operator>=, 90
 - operator==, 90
 - w0, 91
 - w1, 91
 - zero, 89
- upcxx::dist_id
 - dig_, 96
 - here, 96
 - when_here, 96
- upcxx::dist_id< T >, 96
- upcxx::dist_object
 - ~dist_object, 98
 - dist_object, 97, 98
 - fetch, 98
 - id, 98
 - operator*, 98
 - operator->, 99
- upcxx::dist_object< T >, 97
- upcxx::fast_hasher, 99
 - fast_hasher, 99
 - operator(), 100
 - result, 100
 - s, 100
- upcxx::fast_hashing
 - operator(), 101
- upcxx::fast_hashing< T >, 101
- upcxx::function_ref< Ret(Arg...)>, 102
 - function_ref, 102, 103
 - operator(), 103
 - operator=, 103
- upcxx::function_ref< Sig >, 101
- upcxx::future1
 - ~future1, 106
 - future1, 105–107
 - impl_, 109
 - impl_type, 105
 - kind_type, 105
 - operator=, 107
 - ready, 108
 - result, 108
 - result_moved, 108
 - results, 108
 - results_moved, 108
 - results_type, 105
 - then, 109
 - then_pure, 109
- upcxx::future1< Kind, T >, 104
- upcxx::future_cx< ordinal >, 121
- upcxx::future_is_trivially_ready
 - value, 170
- upcxx::future_is_trivially_ready< future1< detail←::future_kind_mapped< Arg, Fn >, T... > >, 170
 - value, 170
- upcxx::future_is_trivially_ready< future1< detail←::future_kind_result, T... > >, 171
 - value, 171
- upcxx::future_is_trivially_ready< future1< detail←::future_kind_shref< HeaderOps >, T... > >, 171
 - value, 172
- upcxx::future_is_trivially_ready< future1< detail←::future_kind_when_all< Arg... >, T... > >, 172
 - value, 172

- upcxx::future_is_trivially_ready< FutureOrKind >, 169
- upcxx::global_ptr
 - element_type, 181
 - global_ptr, 182
 - is_local, 182
 - is_null, 183
 - local, 183
 - operator!=, 183
 - operator<, 185
 - operator<<, 186
 - operator<=, 185
 - operator>, 185
 - operator>=, 185
 - operator+, 183
 - operator++, 183, 184
 - operator-, 184
 - operator--, 184
 - operator==, 185
 - rank_, 187
 - raw_ptr_, 187
 - reinterpret_pointer_cast, 186
 - std::greater< global_ptr< T > >, 186
 - std::greater_equal< global_ptr< T > >, 186
 - std::hash< global_ptr< T > >, 187
 - std::less< global_ptr< T > >, 187
 - std::less_equal< global_ptr< T > >, 187
 - where, 186
- upcxx::global_ptr< T >, 180
- upcxx::hasher_reflector
 - h, 193
 - opaque, 193
 - operator(), 193
- upcxx::hasher_reflector< Hasher >, 192
- upcxx::index_sequence<... >, 194
- upcxx::integer_golden_ratio
 - value, 194
- upcxx::integer_golden_ratio< T >, 194
- upcxx::integer_golden_ratio_bits< 32 >, 195
 - value, 195
- upcxx::integer_golden_ratio_bits< 64 >, 196
 - value, 196
- upcxx::integer_golden_ratio_bits< bit_n >, 195
- upcxx::mod2n_hashing
 - operator(), 214
- upcxx::mod2n_hashing< T >, 214
- upcxx::nil_cx, 214
- upcxx::nop_function< Ret(Arg...)>, 215
 - operator(), 215
- upcxx::nop_function< Sig >, 215
- upcxx::nop_function< void(Arg...)>, 216
 - operator(), 216
- upcxx::packing< bound_function< Fn, B... > >, 218
 - pack, 219
 - size_ubound, 219
 - unpack, 219
- upcxx::packing< std::array< T, n > >, 220
 - pack, 220
 - size_ubound, 220
- unpack, 220
- upcxx::packing< std::pair< A, B > >, 221
 - pack, 221
 - size_ubound, 221
 - unpack, 222
- upcxx::packing< std::string >, 222
 - pack, 222
 - size_ubound, 223
 - unpack, 223
- upcxx::packing< std::tuple< T... > >, 223
 - move_from_storage, 224
 - pack, 224
 - size_ubound, 224
 - unpack, 224
- upcxx::packing< std::unordered_map< K, V > >, 225
 - pack, 225
 - size_ubound, 225
 - unpack, 225
- upcxx::packing< std::unordered_set< T > >, 226
 - pack, 226
 - size_ubound, 226
 - unpack, 227
- upcxx::packing< std::vector< T > >, 227
 - pack, 227
 - size_ubound, 228
 - unpack, 228
- upcxx::packing< T >, 218
- upcxx::packing< T & >, 228
- upcxx::packing< T && >, 229
- upcxx::packing< T const >, 229
- upcxx::packing< T volatile >, 230
- upcxx::packing_empty< T, false >, 231
 - is_trivial, 232
 - pack, 231
 - size_ubound, 231
 - unpack, 231
- upcxx::packing_empty< T, is_default_constructible >, 230
- upcxx::packing_empty< T, true >, 232
 - is_trivial, 233
 - pack, 233
 - size_ubound, 233
 - unpack, 233
- upcxx::packing_function_pointer
 - is_trivial, 235
 - pack, 234
 - size_ubound, 234
 - unpack, 235
- upcxx::packing_function_pointer< T >, 234
- upcxx::packing_is_trivial
 - value, 236
- upcxx::packing_is_trivial< T, false >, 235
- upcxx::packing_is_trivial< T, std::integral_constant< bool, false & packing< T >::is_trivial > >, 236
 - value, 236
- upcxx::packing_not_supported
 - size_ubound, 237
- upcxx::packing_not_supported< T >, 237

- upcxx::packing_opaque< T, false, false >, 238
- upcxx::packing_opaque< T, false, true >, 238
- upcxx::packing_opaque< T, is_empty, is_trivially_copyable >, 237
- upcxx::packing_opaque< T, true, is_trivially_copyable >, 239
- upcxx::packing_pack_reflector, 239
 - opaque, 240
 - operator(), 240
 - w, 240
- upcxx::packing_reflected< T, false >, 241
 - pack, 241
 - size_ubound, 241
 - unpack, 242
- upcxx::packing_reflected< T, is_default_constructible >, 240
- upcxx::packing_reflected< T, true >, 242
 - pack, 242
 - size_ubound, 243
 - unpack, 243
- upcxx::packing_specializer< T, false, false, false >, 244
- upcxx::packing_specializer< T, false, false, true >, 244
- upcxx::packing_specializer< T, false, true, is_scalar >, 245
- upcxx::packing_specializer< T, is_empty, is_funptr, is_scalar >, 243
- upcxx::packing_specializer< T, true, is_funptr, is_scalar >, 245
- upcxx::packing_trivial
 - is_trivial, 247
 - pack, 246
 - size_ubound, 246
 - unpack, 247
- upcxx::packing_trivial< T >, 246
- upcxx::packing_ubound_reflector, 251
 - opaque, 251
 - operator(), 251
 - ub, 252
- upcxx::packing_unpack_reflector< false >, 252
 - opaque, 253
 - operator(), 253
 - r, 253
- upcxx::packing_unpack_reflector< member_assignment←_not_construction >, 252
- upcxx::packing_unpack_reflector< true >, 254
 - opaque, 254
 - operator(), 254
 - r, 255
- upcxx::parcel_layout, 255
 - add_bytes, 256
 - add_trivial_aligned, 256
 - add_trivial_unaligned, 256
 - alignment, 257
 - embed, 257
 - operator<<, 257
 - parcel_layout, 256
 - size, 257
 - size_aligned, 257
- upcxx::parcel_reader, 258
 - ~parcel_reader, 259
 - buffer, 259
 - layout, 259
 - parcel_reader, 258
 - pop, 259
 - pop_char, 259
 - pop_int8, 260
 - pop_trivial_aligned, 260
 - pop_trivial_unaligned, 260
 - pop_uint8, 260
- upcxx::parcel_writer, 261
 - alignment, 262
 - buf_, 264
 - buffer, 262
 - lay_, 264
 - layout, 262
 - parcel_writer, 261
 - put, 262
 - put_char, 262
 - put_int8, 263
 - put_trivial_aligned, 263
 - put_trivial_unaligned, 263
 - put_uint8, 264
 - size, 264
- upcxx::print_proxy
 - operator<<, 265
 - subject, 265
- upcxx::print_proxy< T >, 265
- upcxx::print_reflector, 266
 - comma, 266
 - o, 267
 - opaque, 266
 - operator(), 266
- upcxx::promise< T >, 267
- upcxx::promise_cx
 - pro_, 268
- upcxx::promise_cx< T >, 267
- upcxx::reflection
 - operator(), 270
- upcxx::reflection< std::array< T, n > >, 270
 - operator(), 270
- upcxx::reflection< std::pair< A, B > >, 271
 - operator(), 271
- upcxx::reflection< std::tuple< Ts... > >, 271
 - operator(), 272
- upcxx::reflection< Subject >, 269
- upcxx::reflection_tuple
 - operator(), 272
- upcxx::reflection_tuple< Tup, i, n >, 272
- upcxx::reflection_tuple< Tup, n, n >, 273
 - operator(), 273
- upcxx::rpc_cx
 - fn_, 302
 - rpc_cx, 302
- upcxx::rpc_cx< Fn >, 302
- upcxx::trait_all< Tr >, 322
- upcxx::trait_all< Tr0, Trs... >, 322

- upcxx::trait_all< Tr0, Trs... >::type value, [337](#)
- upcxx::trait_all< Tr0, Trs... >::type< T >, [337](#)
- upcxx::trait_all<>, [323](#)
 - type, [323](#)
- upcxx::trait_any< Tr >, [324](#)
- upcxx::trait_any< Tr0, Trs... >, [324](#)
- upcxx::trait_any< Tr0, Trs... >::type value, [338](#)
- upcxx::trait_any< Tr0, Trs... >::type< T >, [337](#)
- upcxx::trait_any<>, [324](#)
 - type, [325](#)
- upcxx::trait_forall< Test >, [325](#)
 - value, [326](#)
- upcxx::trait_forall< Test, T >, [325](#)
- upcxx::trait_forall< Test, T, Ts... >, [326](#)
 - value, [326](#)
- upcxx::trait_forall_tupled< Test, std::tuple< T... > >, [327](#)
 - value, [328](#)
- upcxx::trait_forall_tupled< Test, Tuple >, [327](#)
- upcxx::tuple_types_into< std::tuple< T... >, Into >, [336](#)
 - type, [336](#)
- upcxx::tuple_types_into< Tuple, Into >, [336](#)
- use_lock
 - dl_malloc.c, [392](#)
- use_mmap
 - dl_malloc.c, [393](#)
- use_noncontiguous
 - dl_malloc.c, [393](#)
- user
 - upcxx, [24](#)
- user_actions_head_
 - upcxx::backend::gasnet1_seq, [45](#)
- user_actions_tailp_
 - upcxx::backend::gasnet1_seq, [45](#)
- usmblocks
 - mallinfo, [204](#)
- utility.hpp, [421](#)
- value
 - upcxx::binding_is_trivial, [75](#)
 - upcxx::binding_is_trivial< T, std::integral_↵ constant< bool, binding< T >::is_trivial > >, [76](#)
 - upcxx::future_is_trivially_ready, [170](#)
 - upcxx::future_is_trivially_ready< future1< detail_↵ ::future_kind_mapped< Arg, Fn >, T... > >, [170](#)
 - upcxx::future_is_trivially_ready< future1< detail_↵ ::future_kind_result, T... > >, [171](#)
 - upcxx::future_is_trivially_ready< future1< detail_↵ ::future_kind_shref< HeaderOps >, T... > >, [172](#)
 - upcxx::future_is_trivially_ready< future1< detail_↵ ::future_kind_when_all< Arg... >, T... > >, [172](#)
 - upcxx::integer_golden_ratio, [194](#)
 - upcxx::integer_golden_ratio_bits< 32 >, [195](#)
 - upcxx::integer_golden_ratio_bits< 64 >, [196](#)
 - upcxx::packing_is_trivial, [236](#)
 - upcxx::packing_is_trivial< T, std::integral_↵ constant< bool, false &packing< T >::is_↵ _trivial > >, [236](#)
 - upcxx::trait_all< Tr0, Trs... >::type, [337](#)
 - upcxx::trait_any< Tr0, Trs... >::type, [338](#)
 - upcxx::trait_forall< Test >, [326](#)
 - upcxx::trait_forall< Test, T, Ts... >, [326](#)
 - upcxx::trait_forall_tupled< Test, std::tuple< T... > >, [328](#)
 - value_
 - upcxx::constant_function, [86](#)
- w
 - upcxx::packing_pack_reflector, [240](#)
- w0
 - upcxx::digest, [91](#)
- w1
 - upcxx::digest, [91](#)
- wait
 - upcxx, [38](#)
- wait.hpp, [423](#)
- when_all
 - upcxx, [39](#)
- when_all_return_t
 - upcxx::detail, [49](#)
- when_here
 - upcxx::dist_id, [96](#)
- where
 - upcxx::global_ptr, [186](#)
- with_types
 - upcxx::detail::future_kind_mapped, [173](#)
 - upcxx::detail::future_kind_result, [174](#)
 - upcxx::detail::future_kind_shref, [175](#)
 - upcxx::detail::future_kind_when_all, [175](#)
- zero
 - upcxx::digest, [89](#)