SmartDSP Operating System API Reference Manual

Document Number: CWSCSDOSAPIRM
Nov 2015

freescale™

Contents

Chapter 1
Introduction

Chapter 2
Kernel

2.1 Overview . . . 3

2.2 SmartDSP OS Initialization . . . 3
2.2.1 Overview . . . 3
2.2.2 Data Structure Documentation . . . 4
2.2.2.1 struct os_mem_heap_t . . . 4
2.2.3 Function Documentation . . . 4
2.2.3.1 osInitializeKernel . . . 4
2.2.3.2 osActivate . . . 5
2.2.3.3 osArchInitialize . . . 5
2.2.3.4 osArchDevicesInitialize . . . 6
2.2.3.5 osMemInitialize . . . 6
2.2.3.6 osMemLocalInitialize . . . 6
2.2.3.7 osMemSharedInitialize . . . 7
2.2.3.8 osHwiInitialize . . . 7
2.2.3.9 osTickInitialize . . . 8
2.2.3.10 osTickStart . . . 8
2.2.3.11 osSwiInitialize . . . 8
2.2.3.12 osTimerInitialize . . . 9
2.2.3.13 osHwTimerInitialize . . . 9
2.2.3.14 osMessageInitialize . . . 9
2.2.3.15 osMultiCoreSyncInitialize . . . 10
2.2.3.16 sysInit . . . 10
2.2.3.17 osQueueInitialize . . . 10
2.2.3.18 osFrameInitialize . . . 11
2.2.3.19 osCioInitialize . . . 11
2.2.3.20 osBioInitialize . . . 11
2.2.3.21 osSioInitialize . . . 12
2.2.3.22 osCopInitialize . . . 12

Section number	Title	Page
2.2.3.23	osDmaInitialize	12
2.2.3.24	osHwWatchdogsInit	13
2.2.3.25	osEventsInitialize	13
2.2.3.26	osTasksInitialize	13
2.2.3.27	osLogInitialize	14
2.2.4	B4860 Initialization API	14
2.2.4.1	Overview	14
2.2.4.2	Data Structure Documentation	15
2.2.4.2.1	union platform_init_params_t	15
2.2.4.2.1.1	Field Documentation	15
2.2.4.2.1.1.1	__pad0__	15
2.2.4.2.1.1.2	ge	15
2.2.4.2.1.1.3	ice	15
2.2.4.2.1.1.4	dce	15
2.2.4.2.1.1.5	__pad1__	16
2.2.4.2.1.1.6	soee	16
2.2.4.2.1.1.7	vccc	16
2.2.4.2.1.1.8	vcee	16
2.2.4.2.1.1.9	__pad2__	16
2.2.4.2.1.1.10	mpe	16
2.2.4.2.1.1.11	ate	16
2.2.4.2.1.1.12	eccee	16
2.2.4.2.1.1.13	__pad3__	16
2.2.4.2.1.1.14	m_cr	17
2.2.4.3	Function Documentation	17
2.2.4.3.1	b4860GetCoreInformation	17
2.2.4.3.2	b4860Initialize	17
2.2.4.3.3	b4860HetBspStubInitialize	18
2.2.4.3.4	b4860HetBootInitialize	18
2.2.4.3.5	b4860HetInitialize	18
2.2.4.3.6	b4420HssiLanesPowerDown	19
2.3	**HAL-Hardware Abstraction Layers**	**19**
2.3.1	Overview	19
2.3.2	BIO-Buffered I/O Module API	19
2.3.2.1	Overview	19
2.3.2.2	Data Structure Documentation	20
2.3.2.2.1	struct bio_dev_open_params_t	20
2.3.2.2.1.1	Field Documentation	21
2.3.2.2.1.1.1	common_pool	21
2.3.2.2.1.1.2	lld_params	21
2.3.2.2.2	struct bio_ch_open_params_t	21
2.3.2.2.2.1	Field Documentation	21
2.3.2.2.2.1.1	channel_num	21
2.3.2.2.2.1.2	callback	21

Section number	Title	Page
2.3.2.2.2.1.3	cb_parameter	22
2.3.2.2.2.1.4	frames_pool	22
2.3.2.2.2.1.5	buffers_pool	22
2.3.2.2.2.1.6	lld_params	22
2.3.2.2.3	struct bio_channel_t	22
2.3.2.2.3.1	Field Documentation	22
2.3.2.2.3.1.1	status	22
2.3.2.2.3.1.2	callback	22
2.3.2.2.3.1.3	cb_parameter	23
2.3.2.2.3.1.4	lld_ch	23
2.3.2.2.3.1.5	bio_interf	23
2.3.2.2.3.1.6	channel_type	23
2.3.2.2.3.1.7	frames_queue	23
2.3.2.2.3.1.8	frames_pool	23
2.3.2.2.3.1.9	buffers_pool	23
2.3.2.2.3.1.10	buffers_queue	23
2.3.2.2.3.1.11	f_RxFrameBuild	23
2.3.2.2.4	struct bio_interface_stat_t	23
2.3.2.2.4.1	Field Documentation	24
2.3.2.2.4.1.1	rx_frames	24
2.3.2.2.4.1.2	rx_bytes	24
2.3.2.2.4.1.3	tx_frames	24
2.3.2.2.4.1.4	tx_bytes	24
2.3.2.2.4.1.5	tx_errors	24
2.3.2.2.4.1.6	rx_errors	24
2.3.2.2.4.1.7	interface_speed	24
2.3.2.2.4.1.8	is_promisc	24
2.3.2.3	Macro Definition Documentation	24
2.3.2.3.1	BIO_READ	24
2.3.2.3.2	BIO_WRITE	24
2.3.2.3.3	BIO_ACTIVE	25
2.3.2.3.4	BIO_LLD_COMMAND	25
2.3.2.3.5	BIO_COMMAND_WRITE	25
2.3.2.3.6	BIO_COMMAND_READ	25
2.3.2.3.7	BIO_CHANNEL_TX_ENABLE	25
2.3.2.3.8	BIO_CHANNEL_RX_ENABLE	25
2.3.2.3.9	BIO_CHANNEL_TX_DISABLE	25
2.3.2.3.10	BIO_CHANNEL_RX_DISABLE	25
2.3.2.3.11	BIO_CHANNEL_STAT_GET	25
2.3.2.3.12	BIO_CHANNEL_RX_ADD_ID	26
2.3.2.3.13	BIO_CHANNEL_RX_REMOVE_ID	26
2.3.2.3.14	BIO_CHANNEL_RX_PHYS_LLD_ADDR_ADD	26
2.3.2.3.15	BIO_CHANNEL_RX_PHYS_LLD_ADDR_REMOVE	26
2.3.2.3.16	BIO_DEVICE_TX_ENABLE	26
2.3.2.3.17	BIO_DEVICE_RX_ENABLE	26

Section number	Title	Page
2.3.2.3.18	BIO_DEVICE_TX_DISABLE	26
2.3.2.3.19	BIO_DEVICE_RX_DISABLE	26
2.3.2.3.20	BIO_DEVICE_RESET	26
2.3.2.3.21	BIO_DEVICE_SUPPORT_MULTIBUFFER	27
2.3.2.3.22	BIO_DEVICE_STAT_GET	27
2.3.2.3.23	BIO_REMOVE_BUFFER	27
2.3.2.4	Typedef Documentation	27
2.3.2.4.1	bio_dev_handle	27
2.3.2.5	BIO Initialization API	27
2.3.2.5.1	Overview	27
2.3.2.5.2	Function Documentation	27
2.3.2.5.2.1	osBioDeviceOpen	27
2.3.2.5.2.2	osBioChannelOpen	28
2.3.2.5.2.3	osBioChannelClose	28
2.3.2.6	BIO Runtime API	29
2.3.2.6.1	Overview	29
2.3.2.6.2	Function Documentation	29
2.3.2.6.2.1	osBioChannelTx	29
2.3.2.6.2.2	osBioChannelRx	30
2.3.2.6.2.3	osBioDeviceCtrl	31
2.3.2.6.2.4	osBioChannelCtrl	31
2.3.3	COP-Co-Processor Module API	32
2.3.3.1	Overview	32
2.3.3.2	Data Structure Documentation	33
2.3.3.2.1	struct cop_job_handle	33
2.3.3.2.1.1	Field Documentation	33
2.3.3.2.1.1.1	job_id	33
2.3.3.2.1.1.2	device_specific	34
2.3.3.2.1.1.3	next	34
2.3.3.2.2	struct cop_dev_open_params_t	34
2.3.3.2.2.1	Field Documentation	34
2.3.3.2.2.1.1	dispatch_callback	34
2.3.3.2.2.1.2	error_callback	34
2.3.3.2.2.1.3	lld_params	34
2.3.3.2.2.1.4	max_num_of_channels	34
2.3.3.2.3	struct cop_ch_open_params_t	34
2.3.3.2.3.1	Field Documentation	35
2.3.3.2.3.1.1	channel_num	35
2.3.3.2.3.1.2	num_jobs	35
2.3.3.2.3.1.3	callback_parameter	35
2.3.3.2.3.1.4	error_callback_parameter	35
2.3.3.2.3.1.5	lld_params	35
2.3.3.2.3.1.6	heap	35
2.3.3.2.4	struct cop_interface_stat_t	35
2.3.3.2.4.1	Field Documentation	35

Section number	Title	Page
2.3.3.2.4.1.1	dispatched_jobs	35
2.3.3.2.4.1.2	reaped_jobs	36
2.3.3.2.4.1.3	failed_jobs	36
2.3.3.3	Macro Definition Documentation	36
2.3.3.3.1	MAX_NUM_OF_CHANNELS	36
2.3.3.3.2	COP_CH_REAP	36
2.3.3.3.3	COP_CH_DISPATCH	36
2.3.3.3.4	COP_CH_SHARED_DELETED	36
2.3.3.3.5	COP_LLD_COMMAND	36
2.3.3.3.6	COP_NON_BLOCKING_COMMAND	36
2.3.3.3.7	COP_CHANNEL_DISABLE	36
2.3.3.3.8	COP_CHANNEL_ENABLE	36
2.3.3.3.9	COP_CHANNEL_STAT_GET	37
2.3.3.3.10	COP_CHANNEL_POLL	37
2.3.3.3.11	COP_DEVICE_RESET	37
2.3.3.3.12	COP_DEVICE_STAT_GET	37
2.3.3.3.13	COP_DEVICE_DISPACHER_SET	37
2.3.3.3.14	COP_DEVICE_ISR_SET	37
2.3.3.3.15	COP_CHANNEL_CALLBACK_SET	37
2.3.3.3.16	COP_CHANNEL_CALLBACK_REMOVE	37
2.3.3.4	Typedef Documentation	38
2.3.3.4.1	cop_dev_handle	38
2.3.3.4.2	cop_job_id	38
2.3.3.4.3	os_cop_callback	38
2.3.3.4.4	os_cop_error	38
2.3.3.4.5	cop_channel_t	38
2.3.3.5	COP Initialization API	38
2.3.3.5.1	Overview	38
2.3.3.5.2	Function Documentation	39
2.3.3.5.2.1	osCopDeviceOpen	39
2.3.3.5.2.2	osCopDeviceClose	39
2.3.3.5.2.3	osCopChannelOpen	39
2.3.3.5.2.4	osCopSharedChannelOpen	40
2.3.3.5.2.5	osCopChannelClose	41
2.3.3.5.2.6	osCopSharedChannelClose	42
2.3.3.6	COP Runtime API	42
2.3.3.6.1	Overview	42
2.3.3.6.2	Function Documentation	42
2.3.3.6.2.1	osCopDeviceCtrl	42
2.3.3.6.2.2	osCopChannelCtrl	44
2.3.3.6.2.3	osCopChannelDispatch	44
2.3.4	CIO-Character I/O Module API	45
2.3.4.1	Overview	45
2.3.4.2	Data Structure Documentation	46
2.3.4.2.1	struct cio_ch_open_params_t	46

Section number	Title	Page
2.3.4.2.1.1	Field Documentation	46
2.3.4.2.1.1.1	channel_num	46
2.3.4.2.1.1.2	data_base	47
2.3.4.2.1.1.3	data_size	47
2.3.4.2.1.1.4	lld_params	47
2.3.4.2.1.1.5	cb_tx_rx	47
2.3.4.2.1.1.6	cb_parameter	47
2.3.4.2.2	struct cio_channel_t	47
2.3.4.2.2.1	Field Documentation	48
2.3.4.2.2.1.1	status	48
2.3.4.2.2.1.2	data_base	48
2.3.4.2.2.1.3	data_end	48
2.3.4.2.2.1.4	data_current	48
2.3.4.2.2.1.5	data_current_free	48
2.3.4.2.2.1.6	data_next	48
2.3.4.2.2.1.7	cb_parameter	48
2.3.4.2.2.1.8	borrow_right	48
2.3.4.2.2.1.9	available_size	48
2.3.4.2.2.1.10	max_size	48
2.3.4.2.2.1.11	cb_tx_rx	48
2.3.4.2.2.1.12	lld_ch	48
2.3.4.2.2.1.13	cio_interf	49
2.3.4.2.2.1.14	channel_type	49
2.3.4.3	Macro Definition Documentation	49
2.3.4.3.1	CIO_READ	49
2.3.4.3.2	CIO_WRITE	49
2.3.4.3.3	CIO_ACTIVE	49
2.3.4.3.4	CIO_LLD_COMMAND	49
2.3.4.3.5	CIO_COMMAND_WRITE	49
2.3.4.3.6	CIO_COMMAND_READ	49
2.3.4.3.7	CIO_CHANNEL_TX_ENABLE	49
2.3.4.3.8	CIO_CHANNEL_RX_ENABLE	50
2.3.4.3.9	CIO_CHANNEL_TX_DISABLE	50
2.3.4.3.10	CIO_CHANNEL_RX_DISABLE	50
2.3.4.3.11	CIO_DEVICE_TX_ENABLE	50
2.3.4.3.12	CIO_DEVICE_RX_ENABLE	50
2.3.4.3.13	CIO_DEVICE_TX_DISABLE	50
2.3.4.3.14	CIO_DEVICE_RX_DISABLE	50
2.3.4.4	Typedef Documentation	50
2.3.4.4.1	cio_dev_handle	50
2.3.4.5	CIO Initialization API	51
2.3.4.5.1	Overview	51
2.3.4.5.2	Function Documentation	51
2.3.4.5.2.1	osCioDeviceOpen	51
2.3.4.5.2.2	osCioChannelOpen	51

Section number	Title	Page
2.3.4.5.2.3	osCioChannelClose	52
2.3.4.6	CIO Runtime API	52
2.3.4.6.1	Overview	52
2.3.4.6.2	Function Documentation	52
2.3.4.6.2.1	osCioChannelBufferGet	52
2.3.4.6.2.2	osCioChannelTxBufferPut	54
2.3.4.6.2.3	osCioChannelRxBufferGet	54
2.3.4.6.2.4	osCioChannelRxBufferFree	55
2.3.4.6.2.5	osCioDeviceCtrl	55
2.3.4.6.2.6	osCioChannelCtrl	55
2.3.5	SIO-Synchronized I/O Module API	56
2.3.5.1	Overview	56
2.3.5.2	Data Structure Documentation	57
2.3.5.2.1	struct sio_dev_open_params_t	57
2.3.5.2.1.1	Field Documentation	58
2.3.5.2.1.1.1	rx_callback	58
2.3.5.2.1.1.2	rx_callback_parameter	58
2.3.5.2.1.1.3	tx_callback	58
2.3.5.2.1.1.4	tx_callback_parameter	58
2.3.5.2.1.1.5	error_callback	58
2.3.5.2.1.1.6	lld_params	58
2.3.5.2.2	struct sio_ch_open_params_t	58
2.3.5.2.2.1	Field Documentation	58
2.3.5.2.2.1.1	channel_num	58
2.3.5.2.2.1.2	callback_parameter	59
2.3.5.2.2.1.3	num_of_buffers	59
2.3.5.2.2.1.4	buffer_size	59
2.3.5.2.2.1.5	buffer_data_size	59
2.3.5.2.2.1.6	num_of_channel_buffers	59
2.3.5.2.2.1.7	channel_buffers_base	59
2.3.5.2.2.1.8	lld_params	59
2.3.5.2.3	struct sio_channel_t	59
2.3.5.2.3.1	Field Documentation	60
2.3.5.2.3.1.1	sio_interf	60
2.3.5.2.3.1.2	callback_parameter	60
2.3.5.2.3.1.3	status	60
2.3.5.2.3.1.4	mode	60
2.3.5.2.3.1.5	num_of_buffers	60
2.3.5.2.3.1.6	buffer_size	60
2.3.5.2.3.1.7	buffer_data_size	60
2.3.5.2.3.1.8	channel_buffers_base	60
2.3.5.2.3.1.9	first_driver_buffer	60
2.3.5.2.3.1.10	last_driver_buffer	60
2.3.5.2.3.1.11	lld_channel	61
2.3.5.2.3.1.12	next	61

Section number	Title	Page
2.3.5.2.3.1.13	application_buffer	61
2.3.5.3	Macro Definition Documentation	61
2.3.5.3.1	SIO_READ	61
2.3.5.3.2	SIO_WRITE	61
2.3.5.3.3	SIO_ACTIVE	61
2.3.5.3.4	SIO_LLD_COMMAND	61
2.3.5.3.5	SIO_COMMAND_WRITE	61
2.3.5.3.6	SIO_COMMAND_READ	61
2.3.5.3.7	SIO_CHANNEL_TX_ENABLE	61
2.3.5.3.8	SIO_CHANNEL_RX_ENABLE	62
2.3.5.3.9	SIO_CHANNEL_TX_DISABLE	62
2.3.5.3.10	SIO_CHANNEL_RX_DISABLE	62
2.3.5.3.11	SIO_DEVICE_TX_ENABLE	62
2.3.5.3.12	SIO_DEVICE_RX_ENABLE	62
2.3.5.3.13	SIO_DEVICE_RX_TX_ENABLE	62
2.3.5.3.14	SIO_DEVICE_TX_DISABLE	62
2.3.5.3.15	SIO_DEVICE_RX_DISABLE	62
2.3.5.3.16	SIO_DEVICE_RX_INTERRUPT_STATUS	63
2.3.5.4	Typedef Documentation	63
2.3.5.4.1	sio_dev_handle	63
2.3.5.5	SIO Initialization API	63
2.3.5.5.1	Overview	63
2.3.5.5.2	Function Documentation	63
2.3.5.5.2.1	osSioDeviceOpen	63
2.3.5.5.2.2	osSioChannelOpen	64
2.3.5.5.2.3	osSioChannelClose	64
2.3.5.6	SIO Runtime API	64
2.3.5.6.1	Overview	64
2.3.5.6.2	Function Documentation	65
2.3.5.6.2.1	osSioDeviceCtrl	65
2.3.5.6.2.2	osSioChannelCtrl	65
2.3.5.6.2.3	osSioBufferGet	66
2.3.5.6.2.4	osSioBufferPut	67
2.4	**System Memory Management**	**67**
2.4.1	Overview	67
2.4.2	Data Structure Documentation	70
2.4.2.1	struct os_mem_part_t	70
2.4.2.1.1	Field Documentation	71
2.4.2.1.1.1	guard	71
2.4.2.1.1.2	array_addr	71
2.4.2.1.1.3	curr_index	71
2.4.2.1.1.4	block_size	71
2.4.2.1.1.5	shared	71
2.4.2.1.1.6	buffer_offset	71

Section number	Title	Page
2.4.3	Macro Definition Documentation	71
2.4.3.1	OFFSET_0_BYTES	71
2.4.3.2	OFFSET_2_BYTES	71
2.4.3.3	OFFSET_4_BYTES	71
2.4.3.4	OFFSET_8_BYTES	71
2.4.3.5	OFFSET_16_BYTES	72
2.4.3.6	OFFSET_32_BYTES	72
2.4.3.7	OFFSET_64_BYTES	72
2.4.3.8	OFFSET_128_BYTES	72
2.4.3.9	OFFSET_256_BYTES	72
2.4.3.10	OFFSET_512_BYTES	72
2.4.3.11	OFFSET_1024_BYTES	72
2.4.3.12	OFFSET_2048_BYTES	72
2.4.3.13	OFFSET_4096_BYTES	72
2.4.3.14	ALIGNED_1_BYTES	72
2.4.3.15	ALIGNED_2_BYTES	73
2.4.3.16	ALIGNED_4_BYTES	73
2.4.3.17	ALIGNED_8_BYTES	73
2.4.3.18	ALIGNED_16_BYTES	73
2.4.3.19	ALIGNED_32_BYTES	73
2.4.3.20	ALIGNED_64_BYTES	73
2.4.3.21	ALIGNED_128_BYTES	73
2.4.3.22	ALIGNED_256_BYTES	73
2.4.3.23	ALIGNED_512_BYTES	73
2.4.3.24	ALIGNED_1024_BYTES	73
2.4.3.25	ALIGNED_2048_BYTES	74
2.4.3.26	ALIGNED_4096_BYTES	74
2.4.3.27	ALIGN_SIZE	74
2.4.3.28	ALIGN_DOWN_SIZE	74
2.4.3.29	ALIGNABLE_SIZE	74
2.4.3.30	ALIGN_ADDRESS	74
2.4.3.31	ALIGN_DOWN_ADDRESS	74
2.4.3.32	ALIGN_DOWN_PHYS_ADDRESS	74
2.4.3.33	IS_ALIGNED	75
2.4.3.34	IS_COMPILER_ALIGNED	75
2.4.3.35	MEM_PART_SIZE	75
2.4.3.36	MEM_PART_DATA_SIZE	75
2.4.3.37	OS_VALID_NUM_MASK	75
2.4.3.38	OS_MEM_FLAGS_MASK	75
2.4.3.39	OS_MEM_TYPE_MASK	75
2.4.3.40	OS_VALID_NUM	76
2.4.3.41	OS_SMARTDSP_HEAP	76
2.4.3.42	OS_MEM_CACHEABLE_TYPE	76
2.4.3.43	OS_MEM_SHARED_TYPE	76
2.4.3.44	OS_MEM_HET_TYPE	76

Section number	Title	Page
2.4.3.45	OS_MEM_M1_TYPE	76
2.4.3.46	OS_MEM_M2_TYPE	76
2.4.3.47	OS_MEM_M3_TYPE	76
2.4.3.48	OS_MEM_DDR0_TYPE	76
2.4.3.49	OS_MEM_DDR1_TYPE	76
2.4.3.50	OS_MEM_QE_PRAM_TYPE	77
2.4.3.51	OS_MEM_MAPLE_PRAM_TYPE	77
2.4.3.52	OS_MEM_MAPLE0_BD_RING_TYPE	77
2.4.3.53	OS_MEM_MAPLE1_BD_RING_TYPE	77
2.4.3.54	OS_MEM_MAPLE2_BD_RING_TYPE	77
2.4.3.55	OS_MEM_PRAM	77
2.4.3.56	OS_MEM_DDR_LOCAL	77
2.4.3.57	OS_MEM_DDR_SHARED	77
2.4.3.58	OS_MEM_DDR_SHARED_CACHEABLE	77
2.4.3.59	OS_MEM_DDR_LOCAL_NONCACHEABLE	78
2.4.4	Enumeration Type Documentation	78
2.4.4.1	os_mem_type	78
2.4.5	MMU - Memory Management Unit API	79
2.4.5.1	Overview	79
2.4.5.2	Data Structure Documentation	80
2.4.5.2.1	struct os_mmu_error	80
2.4.5.2.1.1	Field Documentation	81
2.4.5.2.1.1.1	error_address	81
2.4.5.2.1.1.2	error_pc	81
2.4.5.2.1.1.3	rw_access	81
2.4.5.2.1.1.4	priviledge_level	81
2.4.5.2.1.1.5	access_width	81
2.4.5.3	Macro Definition Documentation	81
2.4.5.3.1	OS_MMU_SYSTEM_CONTEXT	81
2.4.5.4	Typedef Documentation	81
2.4.5.4.1	os_mmu_segment_handle	81
2.4.5.5	Enumeration Type Documentation	81
2.4.5.5.1	mmu_memory_t	81
2.4.5.6	Function Documentation	82
2.4.5.6.1	osMmuDataSegmentFind	82
2.4.5.6.2	osMmuDataSegmentCreate	82
2.4.5.6.3	osMmuDataSegmentEnable	83
2.4.5.6.4	osMmuDataSegmentSizeGet	83
2.4.5.6.5	osMmuDataSegmentVirtBaseGet	84
2.4.5.6.6	osMmuProgSegmentSizeGet	84
2.4.5.6.7	osMmuProgSegmentVirtBaseGet	84
2.4.5.6.8	osMmuDataVirtToPhys	86
2.4.5.6.9	osMmuDataVirtProbe	86
2.4.5.6.10	osMmuDataSegmentProbe	86
2.4.5.6.11	osMmuDataNumOfSegmentsGet	87

Section number	Title	Page
2.4.5.6.12	osMmuDataNumOfUsedSegmentsGet	87
2.4.5.6.13	osMmuDataMaxIdGet	87
2.4.5.6.14	osMmuDataErrorDetect	87
2.4.5.6.15	osMmuDataErrorClear	88
2.4.5.7	Data MMU API	88
2.4.5.8	Program MMU API	88
2.4.5.8.1	Overview	88
2.4.5.8.2	Function Documentation	88
2.4.5.8.2.1	osMmuProgSegmentFind	88
2.4.5.8.2.2	osMmuProgSegmentCreate	90
2.4.5.8.2.3	osMmuProgSegmentEnable	91
2.4.5.8.2.4	osMmuProgCurrentIDSet	91
2.4.5.8.2.5	osMmuProgCurrentIDGet	91
2.4.5.8.2.6	osMmuProgVirtToPhys	91
2.4.5.8.2.7	osMmuProgVirtProbe	92
2.4.5.8.2.8	osMmuProgSegmentProbe	92
2.4.5.8.2.9	osMmuDataGetAttr	92
2.4.5.8.2.10	osMmuProgGetAttr	93
2.4.5.8.2.11	osMmuProgNumOfSegmentsGet	93
2.4.5.8.2.12	osMmuProgNumOfUsedSegmentsGet	93
2.4.5.8.2.13	osMmuProgMaxIdGet	94
2.4.5.8.2.14	osMmuProgErrorDetect	94
2.4.5.8.2.15	osMmuProgErrorClear	94
2.4.5.8.2.16	osMmuDataPhysToVirtManual	94
2.4.5.9	SC3900 MMU Initialization API	95
2.4.5.9.1	Overview	95
2.4.5.9.2	Data Structure Documentation	95
2.4.5.9.2.1	union sc39xx_mmu_init_params_t	95
2.4.5.9.2.1.1	Field Documentation	95
2.4.5.9.2.1.2	gr	95
2.4.5.9.2.1.3	gp	95
2.4.5.9.2.1.4	ge	95
2.4.5.9.2.1.5	ice	96
2.4.5.9.2.1.6	dce	96
2.4.5.9.2.1.7	cvae	96
2.4.5.9.2.1.8	soee	96
2.4.5.9.2.1.9	vccc	96
2.4.5.9.2.1.10	ncee	96
2.4.5.9.2.1.11	mpe	96
2.4.5.9.2.1.12	ate	97
2.4.5.9.2.1.13	eccee	97
2.4.5.9.2.1.14	m_cr	97
2.4.5.10	SC3900 MMU API	97
2.4.5.10.1	Overview	97
2.4.5.10.2	Macro Definition Documentation	100

Section number	Title	Page
2.4.5.10.2.1	MMU_FIELD_SET	100
2.4.5.10.2.2	MMU_ATTR_FIELD_SET	100
2.4.5.10.2.3	MMU_ATTR_FIELD_GET	100
2.4.5.10.2.4	MMU_BIT	100
2.4.5.10.2.5	MMU_BIT_SET	100
2.4.5.10.2.6	MMU_ATTR_BIT_SET	100
2.4.5.10.2.7	MMU_ATTR_BIT_GET	100
2.4.5.10.2.8	MMU_REGA_SHIFT	100
2.4.5.10.2.9	MMU_REGC_SHIFT	100
2.4.5.10.2.10	MMU_MAX_TID	100
2.4.5.10.2.11	MMU_PROG_MATT_ENABLE	101
2.4.5.10.2.12	MMU_PROG_DEF_XPERM_SUPER	101
2.4.5.10.2.13	MMU_PROG_DEF_XPERM_USER	101
2.4.5.10.2.14	MMU_PROG_CACHEABLE_REGION	101
2.4.5.10.2.15	MMU_PROG_NO_PREFETCH	101
2.4.5.10.2.16	MMU_PROG_PREFETCH_ON_MISS	101
2.4.5.10.2.17	MMU_PROG_PREFETCH_ANY	101
2.4.5.10.2.18	MMU_PROG_FLEX_SEGMENT_MODEL	101
2.4.5.10.2.19	MMU_PROG_PID	101
2.4.5.10.2.20	MMU_PROG_L2_PARTITION_ID	101
2.4.5.10.2.21	MMU_PROG_COHERENT	102
2.4.5.10.2.22	MMU_DATA_MATT_ENABLE	102
2.4.5.10.2.23	MMU_DATA_DEF_WPERM_SUPER	102
2.4.5.10.2.24	MMU_DATA_DEF_RPERM_SUPER	102
2.4.5.10.2.25	MMU_DATA_DEF_WPERM_USER	102
2.4.5.10.2.26	MMU_DATA_DEF_RPERM_USER	102
2.4.5.10.2.27	MMU_DATA_CACHEABLE_REGION	102
2.4.5.10.2.28	MMU_DATA_WRITETHROUGH_REGION	102
2.4.5.10.2.29	MMU_DATA_NO_PREFETCH	102
2.4.5.10.2.30	MMU_DATA_PREFETCH_ON_MISS	102
2.4.5.10.2.31	MMU_DATA_PREFETCH_ANY	103
2.4.5.10.2.32	MMU_DATA_FLEX_SEGMENT_MODEL	103
2.4.5.10.2.33	MMU_DATA_DID	103
2.4.5.10.2.34	MMU_DATA_COHERENT	103
2.4.5.10.2.35	MMU_DATA_PERIPHERAL	103
2.4.5.10.2.36	MMU_DATA_GUARDED	103
2.4.5.10.2.37	MMU_DATA_STACK_ENABLE	103
2.4.5.10.2.38	MMU_DATA_L2_PARTITION_ID	103
2.4.5.10.2.39	M_CR_GE	103
2.4.5.10.2.40	M_CR_ICE	103
2.4.5.10.2.41	M_CR_DCE	104
2.4.5.10.2.42	M_CR_SOEE	104
2.4.5.10.2.43	M_CR_VCCC	104
2.4.5.10.2.44	M_CR_VCEE	104
2.4.5.10.2.45	M_CR_MPE	104

Section number	Title	Page
2.4.5.10.2.46	M_CR_ATE	104
2.4.5.10.2.47	M_CR_ECCEE	104
2.4.5.10.2.48	M_CR_CMIR	104
2.4.5.10.2.49	M_DSR_DCCV	104
2.4.5.10.2.50	M_DSR_DAVW	104
2.4.5.10.2.51	M_DSR_DAVD	104
2.4.5.10.2.52	M_DSR_DMSD	104
2.4.5.10.2.53	M_DSR_DPV	104
2.4.5.10.2.54	M_DSR_DNAE	105
2.4.5.10.2.55	M_DSR_DSM	105
2.4.5.10.2.56	M_DSR_DECC	105
2.4.5.10.2.57	M_DSR_DRE	105
2.4.5.10.2.58	M_DSR_DNC	105
2.4.5.10.2.59	M_DSR_DNCC	105
2.4.5.10.2.60	M_DSR_DSOV	105
2.4.5.10.2.61	M_DSR_DSAS	105
2.4.5.10.2.62	M_DSR_DPAS	105
2.4.5.10.2.63	M_PSR_PMSD	105
2.4.5.10.2.64	M_PSR_PNAE	105
2.4.5.10.2.65	M_PSR_PSM	105
2.4.5.10.2.66	M_PSR_PECC	105
2.4.5.10.2.67	M_PSR_PPV	106
2.4.5.10.2.68	M_PSR_PFE	106
2.4.5.10.2.69	M_PSR_PNC	106
2.4.5.10.2.70	MMU_ATTR_PSDA_MASK	106
2.4.5.10.2.71	MMU_ATTR_PSDC_MASK	106
2.4.5.10.2.72	MMU_ATTR_DSDA_MASK	106
2.4.5.10.2.73	MMU_ATTR_DSDC_MASK	106
2.4.5.10.2.74	MMU_ATTR_GET_PSDA	107
2.4.5.10.2.75	MMU_ATTR_GET_PSDC	107
2.4.5.10.2.76	MMU_ATTR_GET_DSDA	107
2.4.5.10.2.77	MMU_ATTR_GET_DSDC	107
2.4.5.10.2.78	MMU_ATTR_GET_MATC	107
2.4.5.10.2.79	MMU_ATTR_GET_MATC	107
2.4.5.10.2.80	MMU_PHYS_ADDR_LOW	107
2.4.5.10.2.81	MMU_PHYS_ADDR_HIGH	107
2.4.5.10.2.82	MMU_SC39XX_MIN_SIZE	107
2.4.5.10.2.83	MMU_SC39XX_MAX_FLEX_SIZE	107
2.4.5.10.2.84	MMU_MATT_ADDRESS_MASK	108
2.4.5.10.2.85	MMU_MATT_ADDRESS_EXT_MASK	108
2.4.5.10.2.86	MMU_DSP_CLUSTER_MATT	108
2.4.5.10.2.87	MMU_SYS_STACK_MATT	108
2.4.5.10.2.88	MMU_TASK_STACK_MATT	108
2.4.6	Memory Allocation API	108
2.4.6.1	Overview	108

Section number	Title	Page
2.4.6.2	Function Documentation	108
2.4.6.2.1	osMalloc	108
2.4.6.2.2	osFastMalloc	109
2.4.6.2.3	osAlignedMalloc	109
2.4.6.2.4	osFastAlignedMalloc	109
2.4.6.2.5	osFree	110
2.4.6.2.6	osMemTypeIsValid	110
2.4.6.2.7	osDumpMemory	110
2.4.6.2.8	osGetFreeMemSize	111
2.4.6.2.9	osGetMaxBlockSize	111
2.4.7	Buffers Pool Management API	111
2.4.7.1	Overview	111
2.4.7.2	Macro Definition Documentation	112
2.4.7.2.1	osMemBlockSize	112
2.4.7.2.2	osMemBlockOffset	112
2.4.7.2.3	osMemBlockUnsafeGet	113
2.4.7.2.4	osMemBlockUnsafeFree	113
2.4.7.2.5	osMemBlockCountGet	113
2.4.7.3	Function Documentation	114
2.4.7.3.1	osMemPartCreate	114
2.4.7.3.2	osMemBlockGet	114
2.4.7.3.3	osMemBlockFree	115
2.4.7.3.4	osMemLocalHeapSet	116
2.4.7.3.5	osMemBlockSyncGet	116
2.4.7.3.6	osMemBlockSyncFree	116
2.4.8	Caches Control API	117
2.4.8.1	Overview	117
2.4.8.2	Data SC3900 Cache API	117
2.4.8.2.1	Overview	117
2.4.8.2.2	Data Structure Documentation	120
2.4.8.2.2.1	struct l2_cache_pf_params_t	120
2.4.8.2.2.1.1	Field Documentation	120
2.4.8.2.2.1.2	start_addr	120
2.4.8.2.2.1.3	buffer_size	120
2.4.8.2.2.2	struct cpc_init_params_t	120
2.4.8.2.2.2.1	Field Documentation	120
2.4.8.2.2.2.2	os_l3_enable	120
2.4.8.2.2.2.3	os_m3_enable	120
2.4.8.2.2.2.4	__pad0__	120
2.4.8.2.2.3	struct l2_error_info_t	120
2.4.8.2.2.3.1	Field Documentation	121
2.4.8.2.2.3.2	l2_error_type	121
2.4.8.2.2.3.3	multi_l2_errors	121
2.4.8.2.2.3.4	err_capture_addr	121
2.4.8.2.2.3.5	err_capture_data	121

Section number	Title	Page
2.4.8.2.2.3.6	ecc_syndrome	121
2.4.8.2.2.3.7	core_id	121
2.4.8.2.2.3.8	trans_source	121
2.4.8.2.2.3.9	trans_type	121
2.4.8.2.2.3.10	double_word_num	122
2.4.8.2.2.3.11	tag_ecc_error_count	122
2.4.8.2.2.3.12	data_ecc_error_count	122
2.4.8.2.2.4	struct l2_init_params_t	122
2.4.8.2.2.4.1	Field Documentation	122
2.4.8.2.2.4.2	os_l2_enable	122
2.4.8.2.2.4.3	os_l2_partition_enable	122
2.4.8.2.2.4.4	os_l2_lock_enable	122
2.4.8.2.2.4.5	os_l2_err_report_enable	122
2.4.8.2.2.4.6	os_l2_err_threshold	123
2.4.8.2.2.4.7	tag_multi_way_hit_int_en	123
2.4.8.2.2.4.8	tag_multiple_bit_ecc_err_int_en	123
2.4.8.2.2.4.9	tag_ecc_error_int_en	123
2.4.8.2.2.4.10	data_multiple_bit_ecc_err_int_en	123
2.4.8.2.2.4.11	data_ecc_err_int_en	123
2.4.8.2.2.4.12	l2_configuration_err_int_en	123
2.4.8.2.2.4.13	__pad0__	123
2.4.8.2.2.4.14	l2_err_isr	123
2.4.8.2.2.5	struct os_cme_maintenance_instruction_params_t	123
2.4.8.2.2.5.1	Field Documentation	124
2.4.8.2.2.5.2	task_id	124
2.4.8.2.2.5.3	virt_addr	124
2.4.8.2.2.5.4	block_size	124
2.4.8.2.2.5.5	maintenance_instruction_type	124
2.4.8.2.2.6	struct l2_query_params_t	124
2.4.8.2.2.6.1	Field Documentation	124
2.4.8.2.2.6.2	l2_cache_bank_id	124
2.4.8.2.2.6.3	l2_cache_way_id	124
2.4.8.2.2.6.4	l2_cache_lock_status	124
2.4.8.2.2.6.5	l2_cache_coherency_status	124
2.4.8.2.2.7	struct l1_query_params_t	124
2.4.8.2.2.7.1	Field Documentation	125
2.4.8.2.2.7.2	l1_cache_way_id	125
2.4.8.2.2.7.3	l1_cache_hit_indication	125
2.4.8.2.2.7.4	sgb_hit_indication	125
2.4.8.2.2.7.5	l1_cache_edc_error_status	125
2.4.8.2.2.8	struct mem_query_params_t	125
2.4.8.2.2.8.1	Field Documentation	125
2.4.8.2.2.8.2	L1_ext_err	125
2.4.8.2.2.8.3	cacheable_descriptor_hit	125
2.4.8.2.2.8.4	bank_0_indication	125

Section number	Title	Page
2.4.8.2.2.8.5	mmue_status	126
2.4.8.2.2.8.6	Segment_descriptor_index	126
2.4.8.2.2.9	struct cme_external_query_result_t	126
2.4.8.2.2.9.1	Field Documentation	126
2.4.8.2.2.9.2	query_physical_address	126
2.4.8.2.2.9.3	l2_query_params	126
2.4.8.2.2.9.4	l1_query_params	126
2.4.8.2.2.9.5	mem_query_params	126
2.4.8.2.3	Macro Definition Documentation	126
2.4.8.2.3.1	OS_LOCK_L2	126
2.4.8.2.3.2	OS_UNLOCK_L2	126
2.4.8.2.3.3	NOT_CACHED	127
2.4.8.2.3.4	L1_CACHED	127
2.4.8.2.3.5	L2_CACHED	127
2.4.8.2.3.6	L1_L2_CACHED	127
2.4.8.2.3.7	OS_L2PAR_STALLOC	127
2.4.8.2.3.8	OS_L2PAR_DRDALLOC	127
2.4.8.2.3.9	OS_L2PAR_IRDALLOC	127
2.4.8.2.3.10	OS_L2PAR_DSTALLOC	127
2.4.8.2.3.11	OS_L2PAR_DATA_DEFAULT	127
2.4.8.2.3.12	OS_L2PAR_PROG_DEFAULT	127
2.4.8.2.4	Typedef Documentation	127
2.4.8.2.4.1	os_l2_cache_err_isr	127
2.4.8.2.5	Enumeration Type Documentation	128
2.4.8.2.5.1	l2_err_trans_t	128
2.4.8.2.5.2	l2_err_trans_source_t	128
2.4.8.2.5.3	l2_error_type_t	128
2.4.8.2.5.4	os_cme_external_query_type	128
2.4.8.2.5.5	os_cme_maintenance_instruction_type	129
2.4.8.2.5.6	os_l2_cache_bank_id	129
2.4.8.2.5.7	os_l2_cache_way_id	130
2.4.8.2.5.8	os_l2_cache_lock_status	130
2.4.8.2.5.9	os_l2_cache_coherency_status	130
2.4.8.2.5.10	os_l1_cache_way_id	131
2.4.8.2.5.11	os_l1_cache_hit_indication	131
2.4.8.2.5.12	os_sgb_hit_indication	131
2.4.8.2.5.13	os_l1_cache_edc_error_status	131
2.4.8.2.5.14	os_mem_L1_ext_err	132
2.4.8.2.5.15	os_mem_cacheable_descriptor_hit	132
2.4.8.2.5.16	os_mem_bank_0_indication	132
2.4.8.2.5.17	os_mem_mmu_error_status	132
2.4.8.2.6	Function Documentation	132
2.4.8.2.6.1	osCmeExtQuery	132
2.4.8.2.6.2	osCmeMaintenanceInstruction	134
2.4.8.2.6.3	osCmeIsDataMaintenanceInstructionInProgress	134

Section number	Title	Page
2.4.8.2.6.4	osCmeIsProgMaintenanceInstructionInProgress	135
2.4.8.2.6.5	osCacheStoreBarrier	135
2.4.8.2.6.6	osCacheLoadBarrier	135
2.4.8.2.6.7	osCacheL2DataLock	136
2.4.8.2.6.8	osCacheL2ProgLock	136
2.4.8.2.6.9	osCacheL2SetPartitioning	136
2.4.8.2.6.10	osCacheL2GetDefaultPartitionWayMask	137
2.4.8.2.6.11	osCacheL2Query	137
2.4.8.2.6.12	osCacheL2GlobalLockClear	137
2.4.8.2.6.13	osCacheL2ClearPartition	137
2.4.8.2.6.14	osCacheL2FastGlobalFlush	138
2.4.8.2.6.15	osSpecificCacheL2GlobalFlush	138
2.4.8.2.6.16	osCacheL2GlobalFlush	138
2.5	**Interrupts**	**138**
2.5.1	Overview	138
2.5.2	Software Interrupts API	139
2.5.2.1	Overview	139
2.5.2.2	Software Interrupts Runtime	139
2.5.2.2.1	Overview	139
2.5.2.2.2	Macro Definition Documentation	140
2.5.2.2.2.1	OS_SWI0	140
2.5.2.2.2.2	OS_SWI1	140
2.5.2.2.2.3	OS_SWI2	140
2.5.2.2.2.4	OS_SWI3	140
2.5.2.2.2.5	OS_SWI4	140
2.5.2.2.2.6	OS_SWI5	141
2.5.2.2.2.7	OS_SWI6	141
2.5.2.2.2.8	OS_SWI7	141
2.5.2.2.2.9	OS_SWI8	141
2.5.2.2.2.10	OS_SWI9	141
2.5.2.2.2.11	OS_SWI10	141
2.5.2.2.2.12	OS_TIMER_SWI	141
2.5.2.2.2.13	OS_SWI_PRIORITY0	141
2.5.2.2.2.14	OS_SWI_PRIORITY1	141
2.5.2.2.2.15	OS_SWI_PRIORITY2	141
2.5.2.2.2.16	OS_SWI_PRIORITY3	141
2.5.2.2.2.17	OS_SWI_PRIORITY4	141
2.5.2.2.2.18	OS_SWI_PRIORITY5	141
2.5.2.2.2.19	OS_SWI_PRIORITY6	142
2.5.2.2.2.20	OS_SWI_PRIORITY7	142
2.5.2.2.2.21	OS_SWI_PRIORITY8	142
2.5.2.2.2.22	OS_SWI_PRIORITY9	142
2.5.2.2.2.23	OS_SWI_PRIORITY10	142
2.5.2.2.2.24	OS_SWI_PRIORITY11	142

Section number	Title	Page
2.5.2.2.2.25	OS_SWI_PRIORITY12	142
2.5.2.2.2.26	OS_SWI_PRIORITY13	142
2.5.2.2.2.27	OS_SWI_PRIORITY14	142
2.5.2.2.2.28	OS_SWI_PRIORITY15	142
2.5.2.2.2.29	OS_SWI_PRIORITY_ALL	142
2.5.2.2.2.30	OS_SWI_PRIORITY_NONE	142
2.5.2.2.2.31	OS_SWI_UNUSED	142
2.5.2.2.2.32	OS_SWI_DELETED	143
2.5.2.2.2.33	OS_SWI_ACQUIRED	143
2.5.2.2.2.34	OS_SWI_CREATED	143
2.5.2.2.2.35	OS_SWI_READY	143
2.5.2.2.3	Function Documentation	143
2.5.2.2.3.1	osSwiCreate	143
2.5.2.2.3.2	osSwiDelete	143
2.5.2.2.3.3	osSwiActivate	144
2.5.2.2.3.4	osSwiEnable	144
2.5.2.2.3.5	osSwiDisable	144
2.5.2.2.3.6	osSwiCountInc	145
2.5.2.2.3.7	osSwiCountDec	146
2.5.2.2.3.8	osSwiCountGet	146
2.5.2.2.3.9	osSwiCountSet	146
2.5.2.2.3.10	osSwiPriorityGet	148
2.5.2.2.3.11	osSwiPrioritySet	148
2.5.2.2.3.12	osSwiStatus	149
2.5.2.2.3.13	osSwiFind	149
2.5.2.2.3.14	osSwiSelf	149
2.5.3	Hardware Interrupts API	150
2.5.3.1	Overview	150
2.5.3.2	Macro Definition Documentation	151
2.5.3.2.1	osHwiRemove	151
2.5.3.2.2	osHwiGetGPIO	151
2.5.3.3	Function Documentation	151
2.5.3.3.1	EI	151
2.5.3.3.2	DI	151
2.5.3.3.3	osHwiSwiftDisableNoLog	151
2.5.3.3.4	osHwiSwiftEnableNoLog	152
2.5.3.3.5	osHwiSwiftNestReduce	152
2.5.3.3.6	osHwiCreate	152
2.5.3.3.7	osHwiDispatcherCreate	153
2.5.3.3.8	osHwiIsCreated	153
2.5.3.3.9	osHwiDelete	154
2.5.3.3.10	osHwiMultiplexedCreate	154
2.5.3.3.11	osHwiMultiplexedDispatcherCreate	155
2.5.3.3.12	osHwiMultiplexedDelete	155
2.5.3.3.13	osHwiPrioritySet	156

Section number	Title	Page
2.5.3.3.14	osHwiPendingClear	156
2.5.3.3.15	osHwiVectorInstall	156
2.5.3.3.16	osHwiDisable	157
2.5.3.3.17	osHwiEnable	157
2.5.3.3.18	osHwiSwiftDisable	158
2.5.3.3.19	osHwiSwiftEnable	158
2.5.3.3.20	osHwiGpioMap	158
2.5.3.3.21	osHwiGpioIrqConfigure	158
2.5.3.3.22	invoke_interrupt	159
2.5.4	Virtual Interrupts API	159
2.5.4.1	Overview	159
2.5.4.2	Function Documentation	159
2.5.4.2.1	osVirtualInterruptsInitialize	159
2.5.4.2.2	osVirtualInterruptFind	160
2.5.4.2.3	osVirtualInterruptRelease	161
2.5.4.2.4	osVirtualInterruptReserve	161
2.5.4.2.5	osVirtualInterruptActivate	161
2.6	**Timers**	**162**
2.6.1	Overview	162
2.6.2	Software Timers API	162
2.6.2.1	Overview	162
2.6.2.2	Software Timers	163
2.6.2.2.1	Overview	163
2.6.2.2.2	Function Documentation	163
2.6.2.2.2.1	osTimerCreate	163
2.6.2.2.2.2	osTimerDelete	164
2.6.2.2.2.3	osTimerStart	164
2.6.2.2.2.4	osTimerStartDelayed	164
2.6.2.2.2.5	osTimerStop	165
2.6.2.2.2.6	osTimerSetInterval	165
2.6.2.2.2.7	osTimerGetDelay	166
2.6.2.2.2.8	osTimerFind	166
2.6.2.2.2.9	osTimerSelf	166
2.6.3	Hardware Timers APIF	167
2.6.3.1	Overview	167
2.6.3.2	Data Structure Documentation	173
2.6.3.2.1	struct timer32_open_params_t	173
2.6.3.2.1.1	Field Documentation	174
2.6.3.2.1.1.1	interval	174
2.6.3.2.1.1.2	preload	174
2.6.3.2.1.1.3	operation_mode	174
2.6.3.2.1.1.4	counting_mode	174
2.6.3.2.1.1.5	direction	174
2.6.3.2.1.1.6	handler	174

Section number	Title	Page
2.6.3.2.1.1.7	priority	174
2.6.3.2.1.1.8	primary_source	174
2.6.3.2.1.1.9	secondary_source	175
2.6.3.2.1.1.10	output_enable	175
2.6.3.2.1.1.11	preload_ctrl1	175
2.6.3.3	Macro Definition Documentation	175
2.6.3.3.1	TIN0SOURCE	175
2.6.3.3.2	TIN1SOURCE	175
2.6.3.3.3	TIN2SOURCE	175
2.6.3.3.4	TIN3SOURCE	175
2.6.3.3.5	OFLAG0SOURCE	175
2.6.3.3.6	OFLAG1SOURCE	175
2.6.3.3.7	OFLAG2SOURCE	175
2.6.3.3.8	OFLAG3SOURCE	176
2.6.3.3.9	PRESCALER1SOURCE	176
2.6.3.3.10	PRESCALER2SOURCE	176
2.6.3.3.11	PRESCALER4SOURCE	176
2.6.3.3.12	PRESCALER8SOURCE	176
2.6.3.3.13	PRESCALER16SOURCE	176
2.6.3.3.14	PRESCALER32SOURCE	176
2.6.3.3.15	PRESCALER64SOURCE	176
2.6.3.3.16	PRESCALER128SOURCE	176
2.6.3.3.17	DEFAULT_TIMER_SOURCE	176
2.6.3.3.18	NUM_OF_HW_TIMERS_SOURCES	177
2.6.3.3.19	HWT_SOC_16BIT	177
2.6.3.3.20	HWT_SOC_32BIT	177
2.6.3.3.21	HWT_SOC_64BIT	177
2.6.3.3.22	COUNT_RISING_EDGE	177
2.6.3.3.23	COUNT_RISING_AND_FALL_EDGE	177
2.6.3.3.24	COUNT_RISING_PRIMARY	177
2.6.3.3.25	COUNT_PRIMARY_DIR_SECONDARY	177
2.6.3.3.26	COUNT_SECONDARY_TRIGGER	177
2.6.3.3.27	COUNT_CASCADED	177
2.6.3.3.28	MAX_HW_TIMERS_SECONDARY_SOURCES	178
2.6.3.3.29	TIMER32_NUM_OF_TRIGGERS_PER_GROUP	178
2.6.3.3.30	TIMER32_MAX_SOC_INTERRUPTS	178
2.6.3.4	Enumeration Type Documentation	178
2.6.3.4.1	anonymous enum	178
2.6.3.4.2	anonymous enum	178
2.6.3.4.3	timer32_operation_mode_t	178
2.6.3.4.4	timer32_counting_mode_t	179
2.6.3.4.5	timer32_primary_source_t	179
2.6.3.4.6	timer32_secondary_source_t	180
2.6.3.4.7	timer32_count_direction_t	180
2.6.3.4.8	timer32_number_in_module_t	180

Section number	Title	Page
2.6.3.4.9	timer32_preload_ctrl_t	180
2.6.3.4.10	anonymous enum	180
2.6.3.4.11	hw_timer_trigger_input_t	181
2.6.3.4.12	timer_clock_input_t	183
2.6.3.4.13	soc_timer32_num_t	184
2.6.3.4.14	soc_timer32_module_t	187
2.6.3.4.15	soc_timer32_group_t	188
2.6.3.4.16	soc_timer32_maple_t	188
2.6.3.5	Function Documentation	188
2.6.3.5.1	osHwTimerGlobalGet	188
2.6.3.5.2	osHwTimerGlobalConfig	189
2.6.3.5.3	osHwTimerLock	189
2.6.3.5.4	osHwTimerUnlock	190
2.6.3.5.5	hwTimer32Open	190
2.6.3.5.6	hwTimer32SetInterval	191
2.6.3.5.7	hwTimer32Delete	191
2.6.3.5.8	hwTimer32Start	192
2.6.3.5.9	hwTimer32Stop	192
2.6.3.5.10	hwTimer32ClearEvent	193
2.6.3.5.11	hwTimer32ValueGet	193
2.6.3.5.12	hwTimer32HoldGet	193
2.6.3.5.13	hwTimer32CounterSet	194
2.6.3.5.14	hwTimer32PreloadSet	194
2.6.3.5.15	hwTimer32CompareSet	195
2.6.3.5.16	hwTimer32SetAndForget	195
2.6.3.5.17	hwTimer32LockClear	195
2.6.3.5.18	hwTimer32GlobalConfig	196
2.6.3.5.19	hwTimer32GlobalGet	196
2.6.3.5.20	hwTimer32SetInitiatorMode	197
2.6.3.5.21	osHwTimerCreate	197
2.6.3.5.22	osHwTimerSetInterval	198
2.6.3.5.23	osHwTimerDelete	199
2.6.3.5.24	osHwTimerStart	199
2.6.3.5.25	osHwTimerStop	199
2.6.3.5.26	osHwTimerClearEvent	200
2.6.3.5.27	osHwTimerFind	200
2.6.3.5.28	osHwTimerTypeFind	200
2.6.3.5.29	osHwTimerFindForce	201
2.6.3.5.30	osHwTimerValueGet	201
2.6.3.5.31	osHwTimerSyncStart	202
2.6.3.5.32	osHwTimerSyncStop	202
2.6.3.5.33	osHwTimerFreeze	202
2.6.3.5.34	osHwTimerUnfreeze	203
2.6.3.5.35	hwTimer32MapleInterruptAssign	204
2.6.3.6	Hardware Timers	204

Section number	Title	Page
2.6.4	CORE Watchdogs API	204
2.6.4.1	Overview	204
2.6.4.2	Macro Definition Documentation	205
2.6.4.2.1	OS_WDT_AUTO_HANDLING	205
2.6.4.2.2	OS_WDT_USER_HANDLING	205
2.6.4.2.3	OS_WDT_DISABLED	205
2.6.4.2.4	OS_CORE_WDT_AUTO_HANDLING	205
2.6.4.2.5	OS_CORE_WDT_USER_HANDLING	206
2.6.4.3	Enumeration Type Documentation	206
2.6.4.3.1	watchdog_wrs_mode	206
2.6.4.4	Function Documentation	206
2.6.4.4.1	osCoreWatchdogSupport	206
2.6.4.4.2	osCoreWatchdogCreate	207
2.6.4.4.3	osCoreWatchdogEnable	207
2.6.4.4.4	osCoreWatchdogDisable	207
2.6.4.4.5	osCoreWatchdogService	208
2.6.4.4.6	osCoreWatchdogCounterGet	208
2.6.5	OS Tick API	208
2.6.5.1	Overview	208
2.6.5.2	OS Tick runtime API	208
2.6.5.2.1	Overview	208
2.6.5.2.2	Function Documentation	208
2.6.5.2.2.1	osTickCount	208
2.6.5.2.2.2	osTickTime	209
2.7	**Utilities**	**209**
2.7.1	Overview	209
2.7.2	General System Utilities	209
2.7.2.1	Overview	209
2.7.2.2	Data Types	209
2.7.2.2.1	Overview	209
2.7.2.2.2	Macro Definition Documentation	210
2.7.2.2.2.1	HAVE_C99_DATATYPES	210
2.7.2.2.2.2	INLINE	210
2.7.2.2.2.3	OS_BYTE_TYPE	210
2.7.2.2.2.4	OS_SHORT_TYPE	210
2.7.2.2.2.5	OS_LONG_TYPE	210
2.7.2.2.2.6	OS_LONG_LONG_TYPE	210
2.7.2.2.2.7	OS_FLOAT_TYPE	210
2.7.2.2.2.8	OS_DOUBLE_TYPE	211
2.7.2.2.2.9	FALSE	211
2.7.2.2.2.10	TRUE	211
2.7.2.2.2.11	NULL	211
2.7.2.3	OS Error Coding	211
2.7.2.3.1	Overview	211

Section number	Title	Page
2.7.2.3.2	Macro Definition Documentation	224
2.7.2.3.2.1	OS_ERR_ARCH_SPECIFIC	224
2.7.2.3.2.2	OS_ERROR_BITS	224
2.7.2.3.2.3	OS_EVENT_BITS	224
2.7.2.3.2.4	OS_INFORMATION_BITS	224
2.7.2.3.2.5	OS_ERROR	224
2.7.2.3.2.6	OS_EVENT	224
2.7.2.3.2.7	OS_INFO	224
2.7.2.3.2.8	OS_ARCH_ERROR	224
2.7.2.3.2.9	OS_ERRARCH_MSC815X	225
2.7.2.3.2.10	OS_ERRARCH_PSC9X3X	225
2.7.2.3.2.11	OS_ERRMODULE_GENERAL	225
2.7.2.3.2.12	OS_ERRMODULE_HWI	225
2.7.2.3.2.13	OS_ERRMODULE_SWI	225
2.7.2.3.2.14	OS_ERRMODULE_TIMERS	225
2.7.2.3.2.15	OS_ERRMODULE_HW_TIMERS	225
2.7.2.3.2.16	OS_ERRMODULE_QUEUES	225
2.7.2.3.2.17	OS_ERRMODULE_MESSAGES	225
2.7.2.3.2.18	OS_ERRMODULE_DMA	225
2.7.2.3.2.19	OS_ERRMODULE_NET	225
2.7.2.3.2.20	OS_ERRMODULE_BIO	225
2.7.2.3.2.21	OS_ERRMODULE_CIO	225
2.7.2.3.2.22	OS_ERRMODULE_SIO	226
2.7.2.3.2.23	OS_ERRMODULE_WDT	226
2.7.2.3.2.24	OS_ERRMODULE_TSK	226
2.7.2.3.2.25	OS_ERRMODULE_EVENT	226
2.7.2.3.2.26	OS_ERRMODULE_EVENT_QUEUE	226
2.7.2.3.2.27	OS_ERRMODULE_EVENT_SEMAPHORE	226
2.7.2.3.2.28	OS_ERRMODULE_LLD	226
2.7.2.3.2.29	OS_ERRMODULE_TICK	226
2.7.2.3.2.30	OS_ERRMODULE_COP	226
2.7.2.3.2.31	OS_ERRMODULE_TDM	226
2.7.2.3.2.32	OS_ERRMODULE_ETHERNET	226
2.7.2.3.2.33	OS_ERRMODULE_UART	226
2.7.2.3.2.34	OS_ERRMODULE_HDI	226
2.7.2.3.2.35	OS_ERRMODULE_DSI	227
2.7.2.3.2.36	OS_ERRMODULE_MMU	227
2.7.2.3.2.37	OS_ERRMODULE_SRIO	227
2.7.2.3.2.38	OS_ERRMODULE_I2C	227
2.7.2.3.2.39	OS_ERRMODULE_SPI	227
2.7.2.3.2.40	OS_ERRMODULE_CLOCK	227
2.7.2.3.2.41	OS_ERRMODULE_CLASS	227
2.7.2.3.2.42	OS_ERRMODULE_PERF_MON	227
2.7.2.3.2.43	OS_ERRMODULE_L2_CACHE	227
2.7.2.3.2.44	OS_ERRMODULE_MAPLE	227

Section number	Title	Page
2.7.2.3.2.45	OS_ERRMODULE_VIRQ	227
2.7.2.3.2.46	OS_ERRMODULE_DEBUG_PRINT	227
2.7.2.3.2.47	OS_ERRMODULE_EMSG	227
2.7.2.3.2.48	OS_ERRMODULE_QML	228
2.7.2.3.2.49	OS_ERRMODULE_BML	228
2.7.2.3.2.50	OS_ERRMODULE_HSSI	228
2.7.2.3.2.51	OS_ERRMODULE_HDLC	228
2.7.2.3.2.52	OS_ERRMODULE_AIC	228
2.7.2.3.2.53	OS_ERRMODULE_CPRI	228
2.7.2.3.2.54	OS_ERRMODULE_BMAN	228
2.7.2.3.2.55	OS_ERRMODULE_QMAN	228
2.7.2.3.2.56	OS_ERRMODULE_DTU	228
2.7.2.3.2.57	OS_ERRMODULE_QDMA	228
2.7.2.3.2.58	OS_ERRMODULE_BOOT	228
2.7.2.3.2.59	OS_FAIL	228
2.7.2.3.2.60	OS_SUCCESS	228
2.7.2.3.2.61	OS_ERR_UNKNOWN	229
2.7.2.3.2.62	OS_ERR_NO_MEMORY	229
2.7.2.3.2.63	OS_ERR_ADDRESS_ERROR	229
2.7.2.3.2.64	OS_ERR_TRANSFER_ERROR	229
2.7.2.3.2.65	OS_ERR_INVALID_CONFIGURATION	229
2.7.2.3.2.66	OS_ERR_HANDLE_INVALID	229
2.7.2.3.2.67	OS_ERR_PARAM_INVALID	229
2.7.2.3.2.68	OS_ERR_MODE_INVALID	229
2.7.2.3.2.69	OS_ERR_FUNCTION_INVALID	229
2.7.2.3.2.70	OS_ERR_COMMAND_INVALID	229
2.7.2.3.2.71	OS_ERR_COMMAND_UNSUPPORTED	229
2.7.2.3.2.72	OS_ERR_SPINLOCK_TAKEN	229
2.7.2.3.2.73	OS_ERR_UNAVAILABLE	229
2.7.2.3.2.74	OS_ERR_ALREADY_CREATED	230
2.7.2.3.2.75	OS_ERR_NOT_CREATED	230
2.7.2.3.2.76	OS_ERR_NOT_ACTIVE	230
2.7.2.3.2.77	OS_ERR_ALREADY_FREE	230
2.7.2.3.2.78	OS_ERR_PRIORITY_INVALID	230
2.7.2.3.2.79	OS_ERR_SOURCE_INVALID	230
2.7.2.3.2.80	OS_ERR_DEST_INVALID	230
2.7.2.3.2.81	OS_ERR_INTERVAL_UNSUPPORTED	230
2.7.2.3.2.82	OS_ERR_EMPTY	230
2.7.2.3.2.83	OS_ERR_FULL	230
2.7.2.3.2.84	OS_ERR_TOO_MUCH	230
2.7.2.3.2.85	OS_ERR_BUSY	230
2.7.2.3.2.86	OS_ERR_NO_ROOM	230
2.7.2.3.2.87	OS_ERR_TYPE_INVALID	231
2.7.2.3.2.88	OS_ERR_LENGTH_INVALID	231
2.7.2.3.2.89	OS_ERR_TOO_BIG	231

Section number	Title	Page
2.7.2.3.2.90	OS_ERR_TOO_SMALL	231
2.7.2.3.2.91	OS_ERR_VERSION_ERROR	231
2.7.2.3.2.92	OS_ERR_TIMEOUT	231
2.7.2.3.2.93	OS_ERR_PEND_IN_INTERRUPT	231
2.7.2.3.2.94	OS_ERR_OVERFLOW	231
2.7.2.3.2.95	OS_ERR_DELAY_IN_INTERRUPT	231
2.7.2.3.2.96	OS_ERR_YIELD_IN_INTERRUPT	231
2.7.2.3.2.97	OS_ERR_YIELD_TO_ILLEGAL_TASK	231
2.7.2.3.2.98	OS_ERR_YIELD_TO_SINGLE_TASK	231
2.7.2.3.2.99	OS_ERR_PEND_IN_LOCK	231
2.7.2.3.2.100	OS_ERR_ALREADY_ACTIVE	232
2.7.2.3.2.101	OS_ERR_DELETE_LOCKED	232
2.7.2.3.2.102	OS_ERR_DELAY_IN_LOCK	232
2.7.2.3.2.103	OS_ERR_UNDERRUN	232
2.7.2.3.2.104	OS_RETURN_NO_SUCCESS	232
2.7.2.3.2.105	OS_ERR_TSK_BLOCKING	232
2.7.2.3.2.106	OS_ERR_TSK_ALREADY_SUSPENDED	232
2.7.2.3.2.107	OS_ERR_TSK_NOT_CREATED	232
2.7.2.3.2.108	OS_ERR_TSK_ALREADY_CREATED	232
2.7.2.3.2.109	OS_ERR_TSK_UNAVAILABLE	232
2.7.2.3.2.110	OS_ERR_TSK_DELAY_IN_INTERR	232
2.7.2.3.2.111	OS_ERR_TSK_YIELD_TO_ILLEGAL_TASK	233
2.7.2.3.2.112	OS_ERR_TSK_YIELD_TO_SINGLE_TASK	233
2.7.2.3.2.113	OS_ERR_TSK_OS_ERR_YIELD_IN_INTERR	233
2.7.2.3.2.114	OS_ERR_TSK_OS_ERR_ALREADY_ACTIVE	233
2.7.2.3.2.115	OS_ERR_TSK_NOT_ACTIVE	233
2.7.2.3.2.116	OS_ERR_TSK_DELAY_IN_LOCK	233
2.7.2.3.2.117	OS_ERR_TSK_DELETE_LOCKED	233
2.7.2.3.2.118	OS_ERR_TSK_MMU_SET_IN_INTERR	233
2.7.2.3.2.119	OS_ERR_TSK_STACK_TOO_BIG	233
2.7.2.3.2.120	OS_ERR_TSK_STACK_UNALIGNED	233
2.7.2.3.2.121	OS_ERR_EVENT_UNAVAILABLE	234
2.7.2.3.2.122	OS_ERR_EVENT_PEND_IN_LOCK	234
2.7.2.3.2.123	OS_ERR_EVENT_QUEUE_TIMEOUT	234
2.7.2.3.2.124	OS_ERR_EVENT_QUEUE_UNAVAILABLE	234
2.7.2.3.2.125	OS_ERR_EVENT_QUEUE_PEND_INTERR	234
2.7.2.3.2.126	OS_ERR_EVENT_QUEUE_OVERFLOW	234
2.7.2.3.2.127	OS_ERR_EVENT_QUEUE_ALREADY_CREATED	234
2.7.2.3.2.128	OS_ERR_EVENT_QUEUE_INVALID	234
2.7.2.3.2.129	OS_ERR_EVENT_SEMA_TIMEOUT	234
2.7.2.3.2.130	OS_ERR_EVENT_SEMA_UNAVAILABLE	234
2.7.2.3.2.131	OS_ERR_EVENT_SEMA_PEND_INTERR	235
2.7.2.3.2.132	OS_ERR_EVENT_SEMA_OVERFLOW	235
2.7.2.3.2.133	OS_ERR_EVENT_SEMA_ALREADY_CREATED	235
2.7.2.3.2.134	OS_ERR_EVENT_SEMA_INVALID	235

Section number	Title	Page
2.7.2.3.2.135	OS_ERR_HWI_INVALID	235
2.7.2.3.2.136	OS_ERR_HWI_MODE_INVALID	235
2.7.2.3.2.137	OS_ERR_HWI_FUNCTION_INVALID	235
2.7.2.3.2.138	OS_ERR_HWI_PRIORITY_INVALID	235
2.7.2.3.2.139	OS_ERR_HWI_COMMAND_INVALID	235
2.7.2.3.2.140	OS_ERR_HWI_DISPATCHER_INVALID	235
2.7.2.3.2.141	OS_ERR_HWI_NO_ROOM	236
2.7.2.3.2.142	OS_ERR_SWI_INVALID	236
2.7.2.3.2.143	OS_ERR_SWI_UNAVAILABLE	236
2.7.2.3.2.144	OS_ERR_SWI_ALREADY_CREATED	236
2.7.2.3.2.145	OS_ERR_SWI_NOT_CREATED	236
2.7.2.3.2.146	OS_ERR_SWI_NOT_ACTIVE	236
2.7.2.3.2.147	OS_ERR_SWI_FUNCTION_INVALID	236
2.7.2.3.2.148	OS_ERR_SWI_PRIORITY_INVALID	236
2.7.2.3.2.149	OS_ERR_TM_INVALID	236
2.7.2.3.2.150	OS_ERR_TM_MODE_INVALID	236
2.7.2.3.2.151	OS_ERR_TM_FUNCTION_INVALID	237
2.7.2.3.2.152	OS_ERR_TM_UNAVAILABLE	237
2.7.2.3.2.153	OS_ERR_TM_ALREADY_CREATED	237
2.7.2.3.2.154	OS_ERR_TM_NOT_CREATED	237
2.7.2.3.2.155	OS_ERR_TM_NOT_ACTIVE	237
2.7.2.3.2.156	OS_ERR_HW_TM_INVALID	237
2.7.2.3.2.157	OS_ERR_HW_TM_UNAVAILABLE	237
2.7.2.3.2.158	OS_ERR_HW_TM_ALREADY_CREATED	237
2.7.2.3.2.159	OS_ERR_HW_TM_NOT_CREATED	237
2.7.2.3.2.160	OS_ERR_HW_TM_MODE_INVALID	237
2.7.2.3.2.161	OS_ERR_HW_TM_PRIORITY_INVALID	238
2.7.2.3.2.162	OS_ERR_HW_TM_SOURCE_INVALID	238
2.7.2.3.2.163	OS_ERR_HW_TM_INTERVAL_UNSUPPORTED	238
2.7.2.3.2.164	OS_ERR_HW_TM_ALREADY_ACTIVE	238
2.7.2.3.2.165	OS_ERR_HW_TM_NOT_ACTIVE	238
2.7.2.3.2.166	OS_ERR_HW_TM_OS_ERR_BUSY	238
2.7.2.3.2.167	OS_ERR_Q_UNAVAILABLE	238
2.7.2.3.2.168	OS_ERR_Q_ALREADY_CREATED	238
2.7.2.3.2.169	OS_ERR_Q_NOT_CREATED	238
2.7.2.3.2.170	OS_ERR_Q_EMPTY	238
2.7.2.3.2.171	OS_ERR_Q_FULL	239
2.7.2.3.2.172	OS_ERR_MSG_INVALID	239
2.7.2.3.2.173	OS_ERR_MSG_UNAVAILABLE	239
2.7.2.3.2.174	OS_ERR_MSG_ALREADY_CREATED	239
2.7.2.3.2.175	OS_ERR_MSG_NOT_CREATED	239
2.7.2.3.2.176	OS_ERR_MSG_FUNCTION_INVALID	239
2.7.2.3.2.177	OS_ERR_MSG_NUM_LIMIT	239
2.7.2.3.2.178	OS_ERR_MSG_DEST_INVALID	239
2.7.2.3.2.179	OS_ERR_MSG_SRC_INVALID	239

Section number	Title	Page
2.7.2.3.2.180	OS_ERR_MSG_BUSY	239
2.7.2.3.2.181	OS_ERR_MSG_DEST_LOCKED	240
2.7.2.3.2.182	OS_ERR_MSG_NO_MSG_TO_GET	240
2.7.2.3.2.183	OS_ERR_DMA_RESOURCE_UNAVAILABLE	240
2.7.2.3.2.184	OS_ERR_DMA_RESOURCE_ALREADY_FREE	240
2.7.2.3.2.185	OS_ERR_DMA_NO_ROOM_FOR_CHAIN	240
2.7.2.3.2.186	OS_ERR_DMA_CHAIN_FULL	240
2.7.2.3.2.187	OS_ERR_DMA_CHAIN_TYPE_INVALID	240
2.7.2.3.2.188	OS_ERR_DMA_CHAIN_LENGTH_INVALID	240
2.7.2.3.2.189	OS_ERR_DMA_CHAIN_BOUND	240
2.7.2.3.2.190	OS_ERR_DMA_CHANNEL_BOUND	240
2.7.2.3.2.191	OS_ERR_DMA_CHANNEL_NOT_BOUND	241
2.7.2.3.2.192	OS_ERR_DMA_CHANNEL_INDEX_INVALID	241
2.7.2.3.2.193	OS_ERR_DMA_PRIORITY_TAKEN	241
2.7.2.3.2.194	OS_ERR_DMA_DEST_NOT_ALIGNED	241
2.7.2.3.2.195	OS_ERR_DMA_SOURCE_NOT_ALIGNED	241
2.7.2.3.2.196	OS_ERR_DMA_SIZE_NOT_ALIGNED	241
2.7.2.3.2.197	OS_ERR_NET_CONSUMED	241
2.7.2.3.2.198	OS_ERR_NET_TOO_LARGE	241
2.7.2.3.2.199	OS_ERR_NET_TOO_SMALL	241
2.7.2.3.2.200	OS_ERR_SOCKET_UNAVAILABLE	241
2.7.2.3.2.201	OS_ERR_RTP_LENGTH_ERROR	242
2.7.2.3.2.202	OS_ERR_RTP_VERSION_ERROR	242
2.7.2.3.2.203	OS_ERR_RTCP_VERSION_ERROR	242
2.7.2.3.2.204	OS_ERR_RTCP_TYPE_INVALID	242
2.7.2.3.2.205	OS_ERR_RTCP_LENGTH_ERROR	242
2.7.2.3.2.206	OS_ERR_NET_UNREACHABLE	242
2.7.2.3.2.207	OS_ERR_NET_SP_MISMATCH	242
2.7.2.3.2.208	OS_ERR_NO_BUFFERS	242
2.7.2.3.2.209	OS_ERR_NO_FRAMES	242
2.7.2.3.2.210	OS_ERR_ARP_TABLE_FULL	242
2.7.2.3.2.211	OS_ERR_NET_UNKNOWN_PROTOCOL	243
2.7.2.3.2.212	OS_ERR_NET_ND_TABLE_FULL	243
2.7.2.3.2.213	OS_ERR_MMU_VIRT_OVERLAPPED	243
2.7.2.3.2.214	OS_ERR_MMU_PHYS_OVERLAPPED	243
2.7.2.3.2.215	OS_ERR_MMU_WRONG_ALIGNMENT	243
2.7.2.3.2.216	OS_ERR_MMU_WRONG_SIZE	243
2.7.2.3.2.217	OS_ERR_MMU_WRONG_DESC	243
2.7.2.3.2.218	OS_ERR_MMU_WRONG_BOUNDARY	243
2.7.2.3.2.219	OS_ERR_MMU_MULTIPLE_HIT	243
2.7.2.3.2.220	OS_ERR_MMU_NON_MAPPED	243
2.7.2.3.2.221	OS_ERR_MMU_PRIVILEGE	244
2.7.2.3.2.222	OS_ERR_MMU_NOT_ALIGNED	244
2.7.2.3.2.223	OS_ERR_MMU_THESAME_ADDR	244
2.7.2.3.2.224	OS_ERR_MMU_SEG_MISS	244

Section number	Title	Page
2.7.2.3.2.225	OS_ERR_MMU_EDC	244
2.7.2.3.2.226	OS_ERR_MMU_DSPE_ERR	244
2.7.2.3.2.227	OS_ERR_MMU_PSPE_ERR	244
2.7.2.3.2.228	OS_ERR_MMU_DPV_ERR	244
2.7.2.3.2.229	OS_ERR_MMU_DCCV_ERR	244
2.7.2.3.2.230	OS_ERR_MMU_DAVW_ERR	244
2.7.2.3.2.231	OS_ERR_MMU_DAVD_ERR	245
2.7.2.3.2.232	OS_ERR_MMU_DECC_ERR	245
2.7.2.3.2.233	OS_ERR_MMU_DRE_ERR	245
2.7.2.3.2.234	OS_ERR_MMU_DNC_ERR	245
2.7.2.3.2.235	OS_ERR_MMU_DNCC_ERR	245
2.7.2.3.2.236	OS_ERR_MMU_DSOV_ERR	245
2.7.2.3.2.237	OS_ERR_MMU_DSAS_ERR	245
2.7.2.3.2.238	OS_ERR_MMU_DPAS_ERR	245
2.7.2.3.2.239	OS_ERR_MMU_PECC_ERR	245
2.7.2.3.2.240	OS_ERR_MMU_PFE_ERR	245
2.7.2.3.2.241	OS_ERR_MMU_PNC_ERR	246
2.7.2.3.2.242	OS_ERROR_MEM_TYPE_ATTR_NOT_METCHING	246
2.7.2.3.2.243	OS_EVENT_BIO_DEQUEUE	246
2.7.2.3.2.244	OS_ERR_COP_QUEUE_SIZE_MISMATCH	246
2.7.2.3.2.245	OS_ERR_L2_CACHE_INVALID_FUNCTION	246
2.7.2.3.2.246	OS_ERR_L2_CACHE_INVALID	246
2.7.2.3.2.247	OS_ERR_L2_CACHE_UNAVAILABLE	246
2.7.2.3.2.248	OS_ERR_L2_CACHE_ALREADY_CREATED	246
2.7.2.3.2.249	OS_ERR_L2_CACHE_NOT_CREATED	246
2.7.2.3.2.250	OS_ERR_L2_CACHE_WRONG_ALIGNMENT	246
2.7.2.3.2.251	OS_ERR_L2_CACHE_WRONG_SIZE	247
2.7.2.3.2.252	OS_ERR_L2_CACHE_NOT_CACHEABLE	247
2.7.2.3.2.253	OS_ERR_L2_CACHE_PARTITION_FULL	247
2.7.2.3.2.254	OS_ERR_L2_CACHE_CME_LACK_OF_SPACE	247
2.7.2.3.2.255	OS_ERR_L2_CACHE_CME_DISABLE_IN_CONTROL_REGS	247
2.7.2.3.2.256	OS_ERR_L2_CACHE_CME_INVALID_COMMAND	247
2.7.2.3.2.257	OS_ERR_L2_CACHE_CME_IS_BUSY	247
2.7.2.3.2.258	OS_ERR_L2_CACHE_CME_INVALID_PARAMS	247
2.7.2.3.2.259	OS_ERR_DTU_TRIAD_COUNTERS_ARE_ALREADY_ALLOCATED	247
2.7.2.3.2.260	OS_ERR_I2C_INVALID_PARAM	247
2.7.2.3.2.261	OS_ERR_I2C_LOST_ARBITRATION	248
2.7.2.3.2.262	OS_ERR_I2C_9TH_CLOCK_TIMEOUT	248
2.7.2.3.2.263	OS_ERR_I2C_BUS_STUCK	248
2.7.2.3.2.264	OS_ERR_I2C_UNAVAILABLE	248
2.7.2.3.2.265	OS_ERR_I2C_BUS_BUSY	248
2.7.2.3.2.266	OS_ERR_QMAN_INVALID_PARAM	248
2.7.2.3.2.267	OS_ERR_QMAN_INVALID_STATE	248
2.7.2.3.2.268	OS_ERR_QMAN_QUEUE_IS_EMPTY	248

Section number	Title	Page
2.7.2.3.2.269	OS_ERR_QMAN_PORTAL_IS_BUSY	248
2.7.2.3.2.270	OS_ERR_QMAN_FQ_IS_BUSY	248
2.7.2.3.2.271	OS_ERR_WDT_ALREADY_CREATED	249
2.7.2.3.2.272	OS_ERR_WDT_NOT_CREATED	249
2.7.2.3.2.273	OS_ERR_WDT_INVALID_TIMEOUT	249
2.7.2.3.2.274	OS_ERR_WDT_INVALID_MODE	249
2.7.2.3.2.275	OS_ERR_WDT_DISABLED	249
2.7.2.3.2.276	OS_ERR_WDT_TIMEOUT_TOO_SMALL	249
2.7.2.3.2.277	OS_ERR_SPI_INVALID_MODE	249
2.7.2.3.2.278	OS_ERR_SPI_NO_MEMORY	249
2.7.2.3.2.279	OS_ERR_SPI_UNAVAILABLE	249
2.7.2.3.2.280	OS_ERR_SPI_INVALID_PARAM	249
2.7.2.3.2.281	OS_ERR_SPI_TX_BD_BUSY	250
2.7.2.3.2.282	OS_ERR_SPI_TX_ERROR	250
2.7.2.3.2.283	OS_ERR_SPI_RX_BUSY	250
2.7.2.3.2.284	OS_ERR_SPI_MULTI_MASTER_ERROR	250
2.7.2.3.2.285	OS_ERR_ETHERNET_CHANNEL_UNAVAILABLE	250
2.7.2.3.2.286	OS_ERR_HDLC_CHANNEL_UNAVAILABLE	250
2.7.2.3.2.287	OS_ERR_CLASS_UNAVAILABLE	250
2.7.2.3.2.288	OS_ERR_CLASS_INVALID_CLASS	250
2.7.2.3.2.289	OS_ERR_CLASS_BAD_CONFIG	250
2.7.2.3.2.290	OS_ERR_PERF_MON_BAD_CONFIG	250
2.7.2.3.2.291	OS_ERR_MAPLE_CHANNEL_UNAVAILABLE	251
2.7.2.3.2.292	OS_ERR_MAPLE_NO_ROOM	251
2.7.2.3.2.293	OS_ERR_MAPLE_NO_MEM	251
2.7.2.3.2.294	OS_ERR_MAPLE_TOO_MANY_BD	251
2.7.2.3.2.295	OS_SUCESS_LLD_TX	251
2.7.2.3.2.296	OS_EVENT_LLD_EVENT	251
2.7.2.3.2.297	OS_ERR_VIRQ_UNAVAILABLE	251
2.7.2.3.2.298	OS_ERR_VIRQ_ALREADY_FREE	251
2.7.2.3.2.299	OS_ERR_DEBUG_PRINT_UNAVAILABLE	251
2.7.2.3.2.300	OS_ERR_DEBUG_PRINT_NOT_CREATED	251
2.7.2.3.2.301	OS_ERR_DEBUG_PRINT_NOT_ACTIVE	252
2.7.2.3.2.302	OS_ERR_DEBUG_PRINT_LENGTH_INVALID	252
2.7.2.3.2.303	OS_ERR_DEBUG_PRINT_MDP_ACTIVE	252
2.7.2.3.2.304	OS_ERR_DEBUG_PRINT_OVERFLOW	252
2.7.2.3.2.305	OS_ERR_RIO_NO_SYNC	252
2.7.2.3.2.306	OS_ERR_RIO_PORT_STOPPED	252
2.7.2.3.2.307	OS_ERR_RIO_LINK_RESPONSE_TIME_OUT	252
2.7.2.3.2.308	OS_ERR_RIO_LINK_UNRECOVERABLE_ERROR	252
2.7.2.3.2.309	OS_ERR_EMSG_CHANNEL_UNAVAILABLE	252
2.7.2.3.2.310	OS_ERR_QML_QUEUE_FULL	252
2.7.2.3.2.311	OS_ERR_QML_QUEUE_EMPTY	253
2.7.2.3.2.312	OS_ERR_QML_CHANNEL_UNAVAILABLE	253
2.7.2.3.2.313	OS_ERR_BML_CHANNEL_UNAVAILABLE	253

Section number	Title	Page
2.7.2.3.2.314	OS_ERR_BML_RING_FULL	253
2.7.2.3.2.315	OS_ERR_BML_RING_EMPTY	253
2.7.2.3.2.316	OS_ERR_HSSI_NO_SYNC	253
2.7.2.3.2.317	OS_EVENT_BUSY	253
2.7.2.3.3	Function Documentation	253
2.7.2.3.3.1	osReportEvent	253
2.7.2.3.3.2	REPORT_EVENT	254
2.7.2.3.3.3	REPORT_EVENT_MESSAGE	254
2.7.2.3.3.4	REPORT_ERROR	254
2.7.2.3.3.5	REPORT_ERROR_MESSAGE	255
2.7.2.3.3.6	RETURN_ERROR	256
2.7.2.3.3.7	RETURN_ERROR_MESSAGE	256
2.7.2.4	Accessor API	256
2.7.2.4.1	Overview	256
2.7.2.4.2	Function Documentation	258
2.7.2.4.2.1	READ_DATA8	258
2.7.2.4.2.2	READ_DATA16	258
2.7.2.4.2.3	READ_DATA32	258
2.7.2.4.2.4	READ_DATA64	259
2.7.2.4.2.5	GET_DATA8	259
2.7.2.4.2.6	GET_DATA16	259
2.7.2.4.2.7	GET_DATA32	260
2.7.2.4.2.8	GET_DATA64	260
2.7.2.4.2.9	WRITE_UINT8	261
2.7.2.4.2.10	WRITE_UINT16	261
2.7.2.4.2.11	WRITE_UINT32	261
2.7.2.4.2.12	WRITE_UINT64	262
2.7.2.4.2.13	READ_MEM_UINT8	262
2.7.2.4.2.14	READ_MEM_UINT16	262
2.7.2.4.2.15	READ_MEM_UINT32	263
2.7.2.4.2.16	GET_MEM_UINT8	263
2.7.2.4.2.17	GET_MEM_UINT16	263
2.7.2.4.2.18	GET_MEM_UINT32	264
2.7.2.4.2.19	WRITE_MEM_UINT8	264
2.7.2.4.2.20	WRITE_MEM_UINT16	265
2.7.2.4.2.21	WRITE_MEM_UINT32	265
2.7.2.4.2.22	CLEAR_UINT8	265
2.7.2.4.2.23	SET_UINT8	266
2.7.2.4.2.24	CLEAR_UINT16	267
2.7.2.4.2.25	SET_UINT8	267
2.7.2.4.2.26	CLEAR_UINT32	267
2.7.2.4.2.27	SET_UINT8	268
2.7.2.4.3	SC39XX Accessor API	268
2.7.2.5	General System Functions	268
2.7.2.5.1	Overview	268

Section number	Title	Page
2.7.2.5.2	Data Structure Documentation	269
2.7.2.5.2.1	struct os_version_t	269
2.7.2.5.2.1.1	Field Documentation	270
2.7.2.5.2.1.2	major	270
2.7.2.5.2.1.3	minor	270
2.7.2.5.2.1.4	custom	270
2.7.2.5.2.1.5	patch	270
2.7.2.5.2.2	struct os_core_info_t	270
2.7.2.5.2.2.1	Field Documentation	270
2.7.2.5.2.2.2	soc_core_num	270
2.7.2.5.2.2.3	cluster_core_num	270
2.7.2.5.2.2.4	cluster_num	270
2.7.2.5.2.2.5	soc_master_core	270
2.7.2.5.2.2.6	cluster_master_core	270
2.7.2.5.3	Macro Definition Documentation	271
2.7.2.5.3.1	STACK_MINIMAL_ALIGNMENT	271
2.7.2.5.3.2	HAVE_OS_ERROR_DEBUG	271
2.7.2.5.3.3	OS_COMPILER_ASSERT	271
2.7.2.5.3.4	OS_ASSERT	271
2.7.2.5.3.5	OS_ASSERT_COND	271
2.7.2.5.3.6	OS_ASSERT_COMPILER_COND	271
2.7.2.5.3.7	osGetDeviceID	271
2.7.2.5.3.8	osGetSocMasterCore	272
2.7.2.5.3.9	osGetSocCoreID	272
2.7.2.5.3.10	osGetMpicCoreID	272
2.7.2.5.3.11	osGetMpicMasterCore	272
2.7.2.5.3.12	osGetClusterMasterCore	272
2.7.2.5.3.13	osGetClusterCoreID	273
2.7.2.5.3.14	osGetClusterNum	273
2.7.2.5.3.15	osGetMasterCore	273
2.7.2.5.3.16	osGetCoreID	273
2.7.2.5.4	Function Documentation	274
2.7.2.5.4.1	SYS_NOP	274
2.7.2.5.4.2	COPY_ALIGNED_2	274
2.7.2.5.4.3	osTestDebug	274
2.7.2.5.4.4	osInitialize	274
2.7.2.5.4.5	osStart	275
2.7.2.5.4.6	osGetVersionInfo	275
2.7.2.5.4.7	osErrorDebug	275
2.7.2.5.4.8	osGetSystemClock	276
2.7.2.5.4.9	osGetNumberOfCores	276
2.7.2.5.4.10	osWait	276
2.7.2.5.4.11	osMulticoreSupport	276
2.7.2.5.5	Variable Documentation	276
2.7.2.5.5.1	g_core_info	276

Section number	Title	Page
2.7.2.6	System Information	277
2.7.2.6.1	Overview	277
2.7.2.6.2	Data Structure Documentation	277
2.7.2.6.2.1	struct os_task_info_t	277
2.7.2.6.2.1.1	Field Documentation	277
2.7.2.6.2.1.2	stack_pointer	277
2.7.2.6.2.1.3	top_of_stack	277
2.7.2.6.2.1.4	task_function	278
2.7.2.6.2.1.5	priority	278
2.7.2.6.2.1.6	task_handle	278
2.7.2.6.2.2	struct os_hwi_info_t	278
2.7.2.6.2.2.1	Field Documentation	278
2.7.2.6.2.2.2	vector	278
2.7.2.6.2.3	struct os_swi_info_t	278
2.7.2.6.2.3.1	Field Documentation	278
2.7.2.6.2.3.2	swi_num	278
2.7.2.6.2.4	struct os_sys_info_t	278
2.7.2.6.2.4.1	Field Documentation	279
2.7.2.6.2.4.2	active_task_info	279
2.7.2.6.2.4.3	active_swi_info	279
2.7.2.6.2.4.4	active_hwi_info	279
2.7.2.6.2.4.5	os_context_info	279
2.7.2.6.3	Enumeration Type Documentation	279
2.7.2.6.3.1	os_context_info_t	279
2.7.2.6.4	Function Documentation	279
2.7.2.6.4.1	osSysInfoGet	279
2.7.3	Debug Utilities	280
2.7.3.1	Overview	280
2.7.3.2	CLASS Profiling API	280
2.7.3.2.1	Overview	280
2.7.3.2.2	SC39XX DTU API	280
2.7.3.2.2.1	Overview	280
2.7.3.2.2.2	Macro Definition Documentation	284
2.7.3.2.2.2.1	NUMBER_OF_CORE_COMMAND_REGISTERS	284
2.7.3.2.2.2.2	NUMBER_OF_CORE_COMMAND_DATA_REGISTERS	284
2.7.3.2.2.2.3	NUMBER_OF_PROFILING_COUNTER_VALUE_REGISTERS_A	284
2.7.3.2.2.2.4	NUMBER_OF_PROFILING_SNAPSHOT_REGISTERS_A	285
2.7.3.2.2.2.5	NUMBER_OF_PROFILING_COUNTER_VALUE_REGISTERS_B	285
2.7.3.2.2.2.6	NUMBER_OF_PROFILING_SNAPSHOT_REGISTERS_B	285
2.7.3.2.2.2.7	NUMBER_OF_INDIRECT_EVENT_COND_TRASITION_CONF_↩ REGISTERS	285
2.7.3.2.2.2.8	OS_DTU_MODE	285
2.7.3.2.2.3	Enumeration Type Documentation	285
2.7.3.2.2.3.1	b486x_dtu_triad	285
2.7.3.2.2.3.2	b486x_dtu_triad_counter	285

Section number	Title	Page
2.7.3.2.2.3.3	b486x_dtu_cev_triad_a	285
2.7.3.2.2.3.4	b486x_dtu_cev_triad_b	286
2.7.3.2.2.3.5	b486x_dtu_profiling_mode	287
2.7.3.2.2.4	Function Documentation	289
2.7.3.2.2.4.1	osDtuReadCount	289
2.7.3.2.2.4.2	osDtuInitProfiler	289
2.7.3.2.2.4.3	osDtuDisableProfiler	290
2.7.3.2.2.4.4	osDtuStartProfiling	290
2.7.3.2.2.4.5	osDtuStopProfiling	290
2.7.3.2.3	B4860 Debug Print API	290
2.7.3.2.3.1	Overview	290
2.7.3.2.3.2	B4860 Debug Print Initialization API	290
2.7.3.2.3.2.1	Overview	290
2.7.3.2.3.2.2	Data Structure Documentation	291
2.7.3.2.3.2.3	struct b486x_debug_print_init_params_t	291
2.7.3.2.3.2.4	Field Documentation	291
2.7.3.2.3.2.5	dcsr_phys_addr	291
2.7.3.2.3.2.6	dcsr_virt_addr	291
2.7.3.2.3.2.7	vtb_start_addr	291
2.7.3.2.3.2.8	vtb_mem_location	291
2.7.3.2.3.2.9	vtb_size	291
2.7.3.2.3.2.10	num_of_segments	291
2.7.3.2.3.2.11	flags	291
2.7.3.2.3.2.12	soc_het_ctrl	292
2.7.3.2.3.2.13	mngmnt_cores	292
2.7.3.2.3.2.14	Enumeration Type Documentation	292
2.7.3.2.3.2.15	vtb_size_t	292
2.7.3.2.3.2.16	vtb_segments_t	292
2.7.3.2.3.2.17	Function Documentation	292
2.7.3.2.3.2.18	b486xDebugPrintInitialize	292
2.7.3.2.3.3	B4860 Debug Print runtime API	292
2.7.3.2.3.3.1	Overview	292
2.7.3.2.3.3.2	Enumeration Type Documentation	293
2.7.3.2.3.3.3	b486x_debug_print_trigger_level_t	293
2.7.3.2.3.3.4	Function Documentation	293
2.7.3.2.3.3.5	b486xDebugPrintFast	293
2.7.3.2.3.3.6	b486xDebugPrintMass	293
2.7.3.3	Logging API	294
2.7.3.3.1	Overview	294
2.7.3.3.2	Macro Definition Documentation	295
2.7.3.3.2.1	osDebugHookDelete	295
2.7.3.3.3	Enumeration Type Documentation	295
2.7.3.3.3.1	os_debug_hooks_enum	295
2.7.3.3.4	Function Documentation	296
2.7.3.3.4.1	osDebugHookCreate	296

Section number	Title	Page
2.7.3.3.5	Logging runtime API	296
2.7.3.3.5.1	Overview	296
2.7.3.3.5.2	Macro Definition Documentation	297
2.7.3.3.5.2.1	LOG_ENTER_EVENT	297
2.7.3.3.5.2.2	LOG_EXIT_EVENT	298
2.7.3.3.5.2.3	OS_LOG_HWI	298
2.7.3.3.5.2.4	OS_LOG_SWI	298
2.7.3.3.5.2.5	OS_LOG_SPINLOCK	298
2.7.3.3.5.2.6	OS_LOG_TASK	298
2.7.3.3.5.2.7	OS_LOG_CLASS_UTILIZATION	298
2.7.3.3.5.2.8	OS_LOG_USER_DEFINED_EVENT	298
2.7.3.3.5.2.9	LOG_OPCODE_TO_STACK	298
2.7.3.3.5.2.10	LOG_OPCODE_FROM_STACK	298
2.7.3.3.5.2.11	LOG_COMMAND_TYPE_MASK	298
2.7.3.3.5.2.12	LOG_HWI_ENTER	298
2.7.3.3.5.2.13	LOG_HWI_EXIT	298
2.7.3.3.5.2.14	LOG_HWI_CREATE	298
2.7.3.3.5.2.15	LOG_HWI_SET_PRIORITY	299
2.7.3.3.5.2.16	LOG_SWI_CREATE	299
2.7.3.3.5.2.17	LOG_SWI_ENTER	299
2.7.3.3.5.2.18	LOG_SWI_EXIT	299
2.7.3.3.5.2.19	LOG_SWI_ACTIVATE	299
2.7.3.3.5.2.20	LOG_SWI_SET_PRIORITY	299
2.7.3.3.5.2.21	LOG_SPINLOCK_GET	299
2.7.3.3.5.2.22	LOG_SPINLOCK_RELEASE	299
2.7.3.3.5.2.23	LOG_SPINLOCK_IRQ_GET	299
2.7.3.3.5.2.24	LOG_SPINLOCK_IRQ_RELEASE	299
2.7.3.3.5.2.25	LOG_CLASS_M2_UTILIZATION	299
2.7.3.3.5.2.26	LOG_CLASS_M3_UTILIZATION	299
2.7.3.3.5.2.27	LOG_CLASS_DDR_UTILIZATION	299
2.7.3.3.5.2.28	LOG_CLASS_SKYBLUE_UTILIZATION	300
2.7.3.3.5.2.29	LOG_CLASS_SYSTEM_TO_M2_UTILIZATION	300
2.7.3.3.5.2.30	LOG_CLASS_CORES_TO_M3_DDR_UTILIZATION	300
2.7.3.3.5.2.31	LOG_CLASS_QE_UTILIZATION	300
2.7.3.3.5.2.32	LOG_CLASS_SRIO_UTILIZATION	300
2.7.3.3.5.2.33	LOG_CLASS_TDM_UTILIZATION	300
2.7.3.3.5.2.34	LOG_CLASS_DMA_PORT0_UTILIZATION	300
2.7.3.3.5.2.35	LOG_CLASS_DMA_PORT1_UTILIZATION	300
2.7.3.3.5.2.36	LOG_HWI_ENABLE	300
2.7.3.3.5.2.37	LOG_HWI_DISABLE	300
2.7.3.3.5.2.38	LOG_SWIFT_HWI_DISABLE	300
2.7.3.3.5.2.39	LOG_SWIFT_HWI_ENABLE	301
2.7.3.3.5.2.40	LOG_TASK_CREATE	301
2.7.3.3.5.2.41	LOG_TASK_SWITCH	301
2.7.3.3.5.2.42	LOG_TASK_ACTIVATE	301

Section number	Title	Page
2.7.3.3.5.2.43	LOG_TASK_SET_PRIORITY	301
2.7.3.3.5.2.44	LOG_TASK_PEND_QUEUE	301
2.7.3.3.5.2.45	LOG_TASK_PEND_SEM	301
2.7.3.3.5.2.46	LOG_TASK_DELETE	301
2.7.3.3.5.2.47	LOG_TASK_POST_QUEUE	301
2.7.3.3.5.2.48	LOG_TASK_POST_SEM	301
2.7.3.3.5.2.49	LOG_TASK_SUSPEND	301
2.7.3.3.5.2.50	LOG_TASK_DELAY	301
2.7.3.3.5.2.51	LOG_TASK_YIELD	301
2.7.3.3.5.2.52	GET_FUNCTION_CODE	302
2.7.3.3.5.3	Function Documentation	302
2.7.3.3.5.3.1	osLog	302
2.7.3.3.5.3.2	osLogRegisterHandler	302
2.7.3.3.5.3.3	osLogImmediate	303
2.8	**Queues API**	**303**
2.8.1	Overview	303
2.8.2	Typedef Documentation	304
2.8.2.1	os_queue_handle	304
2.8.3	Enumeration Type Documentation	304
2.8.3.1	os_queue_status	304
2.8.3.2	os_queue_shared	305
2.8.4	Function Documentation	305
2.8.4.1	osQueueCreate	305
2.8.4.2	osQueueCreateMultiple	305
2.8.4.3	osQueueDelete	306
2.8.4.4	osQueueDeleteUnsafe	306
2.8.4.5	osQueueReset	307
2.8.4.6	osQueueResetUnsafe	307
2.8.4.7	osQueueEnqueue	308
2.8.4.8	osQueueUnsafeEnqueue	309
2.8.4.9	osQueueHeadEnqueue	309
2.8.4.10	osQueueUnsafeHeadEnqueue	310
2.8.4.11	osQueueDequeue	310
2.8.4.12	osQueueUnsafeDequeue	311
2.8.4.13	osQueuePeek	311
2.8.4.14	osQueueTailDequeue	312
2.8.4.15	osQueueUnsafeTailDequeue	312
2.8.4.16	osQueueStatus	313
2.8.4.17	osQueueUnsafeStatus	313
2.8.4.18	osQueueFind	314
2.8.4.19	osQueueEnqueueMultiple	314
2.8.4.20	osQueueUnsafeEnqueueMultiple	314
2.8.4.21	osQueueDequeueMultiple	315
2.8.4.22	osQueueUnsafeDequeueMultiple	315

Section number	Title	Page
2.8.4.23	osQueueShared	316
2.8.4.24	osQueueSize	316
2.8.4.25	osQueueMultipleSize	317
2.8.4.26	osQueueMultipleNum	317
2.9	**Frames API**	**317**
2.9.1	Overview	317
2.9.2	Data Structure Documentation	319
2.9.2.1	struct os_frames_pool_t	319
2.9.2.1.1	Field Documentation	319
2.9.2.1.1.1	frames_part	319
2.9.2.1.1.2	buffers_per_frame	319
2.9.2.1.1.3	frame_expansion	320
2.9.2.1.1.4	init_frame_offset	320
2.9.2.1.1.5	f_MemBlockGet	320
2.9.2.1.1.6	f_MemBlockFree	320
2.9.2.2	struct os_frame_t	320
2.9.2.2.1	Field Documentation	321
2.9.2.2.1.1	data_part	321
2.9.2.2.1.2	lld_channel_handle	321
2.9.2.2.1.3	frames_pool	321
2.9.2.2.1.4	info	321
2.9.2.2.1.5	total	322
2.9.2.2.1.6	expansion	322
2.9.2.2.1.7	ref_count	322
2.9.2.2.1.8	total_length	322
2.9.2.2.1.9	total_offset	322
2.9.2.2.1.10	used	322
2.9.2.2.1.11	first	322
2.9.2.2.1.12	last	322
2.9.2.2.1.13	mbf	322
2.9.2.2.1.14	data_length	322
2.9.2.2.1.15	offset	322
2.9.2.2.1.16	data	322
2.9.2.2.1.17	buffers	323
2.9.3	Macro Definition Documentation	323
2.9.3.1	FRAME_NONE	323
2.9.3.2	GET_MAX_BUFFER_SIZE	323
2.9.3.3	osFrameBufferNew	323
2.9.3.4	osFrameBufferFree	323
2.9.3.5	osFrameInfoGet	324
2.9.3.6	osFrameInfoSet	324
2.9.3.7	osFrameRefCntInc	324
2.9.3.8	osFrameRefCntDec	324
2.9.3.9	osFrameRefCntGet	325

Section number	Title	Page
2.9.3.10	osFrameRefCntSet	325
2.9.3.11	osFrameDataPartGet	325
2.9.3.12	osFrameLengthGet	326
2.9.3.13	osFrameLengthSet	326
2.9.3.14	osFrameOffsetGet	326
2.9.3.15	osFrameOffsetBuffer	327
2.9.3.16	osFrameTotalBuffers	327
2.9.3.17	osFrameExpansionGet	328
2.9.3.18	osFrameExpansionSize	328
2.9.3.19	osFrameSingleBufferGet	328
2.9.3.20	osFrameSingleBufferSet	329
2.9.3.21	osFramePoolNumBuffersGet	329
2.9.4	Function Documentation	330
2.9.4.1	osFramePoolCreate	330
2.9.4.2	osFrameGet	330
2.9.4.3	osFrameRelease	331
2.9.4.4	osFrameBufferOffsetGet	331
2.9.4.5	osFrameBufferOffsetSet	331
2.9.4.6	osFrameUsedBuffers	332
2.9.4.7	osFrameBufferGet	333
2.9.4.8	osFrameDataStart	333
2.9.4.9	osFrameBufferSet	334
2.9.4.10	osFrameBufferAppend	334
2.9.4.11	osFrameBufferPrepend	335
2.9.4.12	osFrameHeadBufferRemove	335
2.9.4.13	osFrameTailBufferRemove	336
2.9.4.14	osFrameHeaderPush	337
2.9.4.15	osFrameHeaderPop	337
2.9.4.16	osFrameBufferSetWithOffset	338
2.9.4.17	osFrameTailRemove	338
2.9.4.18	osFrameTailAppend	338
2.9.4.19	osFrameDuplicate	339
2.9.4.20	osFrameCopyToMem	339
2.9.4.21	osFrameLldHeaderVerify	339
2.10	**Clock API**	**340**
2.10.1	Overview	340
2.10.2	B486x Clock API	340
2.10.2.1	Overview	340
2.10.2.2	Macro Definition Documentation	340
2.10.2.2.1	osCoreClockGet	340
2.10.2.2.2	osPlatformClockGet	341
2.10.2.2.3	osSlowPeriphClockGet	341
2.10.2.3	Function Documentation	341
2.10.2.3.1	b486xInitializeClocks	341

Section number	Title	Page
2.11	**Multi Core and Multi Device Management**	**342**
2.11.1	Overview	342
2.11.2	Spinlocks API	342
2.11.2.1	Overview	342
2.11.2.2	Spinlocks initialization functions	342
2.11.2.2.1	Overview	342
2.11.2.2.2	Function Documentation	342
2.11.2.2.2.1	osSpinLockDataStore	342
2.11.2.2.2.2	osSpinLockDataRelease	343
2.11.2.2.2.3	osSpinLockInitialize	343
2.11.2.2.2.4	osSpinLockDisable	343
2.11.2.3	Spinlocks runtime functions	343
2.11.2.3.1	Overview	343
2.11.2.3.2	Macro Definition Documentation	344
2.11.2.3.2.1	osSpinLockSwiftRelease	344
2.11.2.3.2.2	osSpinLockSwiftIrqRelease	344
2.11.2.3.2.3	osSharedResourceLock	344
2.11.2.3.2.4	osSharedResourceUnlock	344
2.11.2.3.3	Function Documentation	346
2.11.2.3.3.1	osSpinLockGet	346
2.11.2.3.3.2	osSpinLockSwiftGet	346
2.11.2.3.3.3	osSpinLockTryGet	346
2.11.2.3.3.4	osSpinLockSwiftTryGet	347
2.11.2.3.3.5	osSpinLockRelease	347
2.11.2.3.3.6	osSpinLockIrqGet	347
2.11.2.3.3.7	osSpinLockSwiftIrqGet	347
2.11.2.3.3.8	osSpinLockIrqRelease	348
2.11.2.3.4	SC3900 Spinlocks with barriers API	348
2.11.2.3.4.1	Overview	348
2.11.2.3.4.2	Function Documentation	348
2.11.2.3.4.2.1	osSpinLockBarrierGet	348
2.11.2.3.4.2.2	osSpinLockBarrierSwiftGet	349
2.11.2.3.4.2.3	osSpinLockBarrierTryGet	349
2.11.2.3.4.2.4	osSpinLockBarrierSwiftTryGet	349
2.11.2.3.4.2.5	osSpinLockBarrierRelease	350
2.11.2.3.4.2.6	osSpinLockBarrierIrqGet	350
2.11.2.3.4.2.7	osSpinLockBarrierSwiftIrqGet	351
2.11.2.3.4.2.8	osSpinLockBarrierIrqRelease	351
2.11.2.3.4.2.9	osSpinLockNoBarrierGet	351
2.11.2.3.4.2.10	osSpinLockNoBarrierSwiftGet	351
2.11.2.3.4.2.11	osSpinLockNoBarrierTryGet	352
2.11.2.3.4.2.12	osSpinLockNoBarrierSwiftTryGet	352
2.11.2.3.4.2.13	osSpinLockNoBarrierRelease	352
2.11.2.3.4.2.14	osSpinLockNoBarrierIrqGet	352
2.11.2.3.4.2.15	osSpinLockNoBarrierSwiftIrqGet	353

Section number	Title	Page
2.11.2.3.4.2.16	osSpinLockNoBarrierIrqRelease	353
2.11.3	Barriers API	353
2.11.3.1	Overview	353
2.11.3.2	Data Structure Documentation	353
2.11.3.2.1	struct os_barrier_t	353
2.11.3.2.1.1	Field Documentation	354
2.11.3.2.1.1.1	guard	354
2.11.3.2.1.1.2	counter	354
2.11.3.2.1.1.3	height	354
2.11.3.2.1.1.4	comm_sensor	354
2.11.3.2.1.1.5	sensors	354
2.11.3.2.1.1.6	init_flag	354
2.11.3.3	Barriers runtime functions	354
2.11.3.3.1	Overview	354
2.11.3.3.2	Function Documentation	355
2.11.3.3.2.1	osCentralBarrierWait	355
2.11.3.3.2.2	osCentralBarrierVerify	355
2.11.3.3.2.3	osWaitForAllCores	355
2.11.3.4	Barriers initialization functions	355
2.11.3.4.1	Overview	355
2.11.3.4.2	Function Documentation	356
2.11.3.4.2.1	osCentralBarrierInit	356
2.11.4	Intercore Messages	356
2.11.4.1	Overview	356
2.11.4.2	Messages API	356
2.11.4.3	Intercore Messages	356
2.11.4.3.1	Overview	356
2.11.4.3.2	Function Documentation	357
2.11.4.3.2.1	osMessageQueueInitialize	357
2.11.4.3.2.2	osMessageQueueCreate	357
2.11.4.3.2.3	osMessageQueueHwiGet	358
2.11.4.3.2.4	osMessageQueuePost	358
2.11.4.3.2.5	osMessageQueueGet	358
2.11.4.3.2.6	osMessageQueueDispatcher	360
2.12	**Multitasking**	**360**
2.12.1	Overview	360
2.12.2	Asymmetric Multiprocessing Scheduling	360
2.12.2.1	Overview	360
2.12.2.2	Event Semaphores API	361
2.12.2.2.1	Overview	361
2.12.2.2.2	Event Semaphores RunTime functions	361
2.12.2.2.2.1	Overview	361
2.12.2.2.2.2	Function Documentation	361
2.12.2.2.2.2.1	osEventSemaphoreFind	361

Section number	Title	Page
2.12.2.2.2.2	osEventSemaphoreCreate	361
2.12.2.2.2.3	osEventSemaphoreDelete	363
2.12.2.2.2.4	osEventSemaphoreReset	363
2.12.2.2.2.5	osEventSemaphoreCountGet	363
2.12.2.2.2.6	osEventSemaphorePend	365
2.12.2.2.2.7	osEventSemaphoreAccept	365
2.12.2.2.2.8	osEventSemaphorePost	366
2.12.2.3	Event Queues API	366
2.12.2.3.1	Overview	366
2.12.2.3.1.0.9	Event Queues RunTime functions	366
2.12.2.3.1.0.10	Overview	366
2.12.2.3.1.0.11	Function Documentation	367
2.12.2.3.1.0.12	osEventQueueFind	367
2.12.2.3.1.0.13	osEventQueueCreate	367
2.12.2.3.1.0.14	osEventQueuePend	367
2.12.2.3.1.0.15	osEventQueueAccept	368
2.12.2.3.1.0.16	osEventQueuePost	369
2.12.2.3.1.0.17	osEventQueueReset	369
2.12.2.4	Tasks API	369
2.12.2.4.1	Overview	369
2.13	**Architectures**	**370**
2.13.1	Overview	370
2.13.2	CCSR Memory Maps	370
2.13.2.1	Overview	370
2.13.2.2	B4860 CCSR Memory Maps	370
2.13.2.2.1	Overview	370
2.13.2.2.2	Data Structure Documentation	370
2.13.2.2.2.1	struct dpaa_memmap_t	370
2.13.2.2.2.2	struct pa_cluster_l2_map_t	371
2.13.2.2.2.2.1	Field Documentation	371
2.13.2.2.2.2.2	l2csr0	371
2.13.2.2.2.2.3	l2csr1	371
2.13.2.2.2.2.4	l2cfg0	371
2.13.2.2.2.2.5	l2ipbrr1	371
2.13.2.2.2.2.6	l2ipbrr2	371
2.13.2.2.2.2.7	l2errinjhi	371
2.13.2.2.2.2.8	l2errinjlo	372
2.13.2.2.2.2.9	l2errinjctl	372
2.13.2.2.2.2.10	l2captdatahi	372
2.13.2.2.2.2.11	l2captdatalo	372
2.13.2.2.2.2.12	l2captecc	372
2.13.2.2.2.2.13	l2errdet	372
2.13.2.2.2.2.14	l2errdis	372
2.13.2.2.2.2.15	l2errinten	372

Section number	Title	Page
2.13.2.2.2.16	l2errattr	372
2.13.2.2.2.17	l2erreaddr	372
2.13.2.2.2.18	l2erraddr	372
2.13.2.2.2.19	l2errctl	372
2.13.2.2.2.20	l2hdbcr0	372
2.13.2.2.2.21	l2hdbcr1	373
2.13.2.2.2.22	l2hdbcr2	373
2.13.3	Peripheral Memory Maps	373
2.13.3.1	Overview	373
2.13.3.2	Macro Definition Documentation	373
2.13.3.2.1	CPRI_MAX_NUM_OF_ANTENNA_CARRIERS	373
2.13.3.3	OCN DMA Memory Map	373
2.13.3.3.1	Overview	373
2.13.3.3.2	Data Structure Documentation	374
2.13.3.3.2.1	struct ocn_dma_ch_map_t	374
2.13.3.3.2.1.1	Field Documentation	374
2.13.3.3.2.1.2	mr	374
2.13.3.3.2.1.3	sr	374
2.13.3.3.2.1.4	eclndar	374
2.13.3.3.2.1.5	clndar	374
2.13.3.3.2.1.6	satr	374
2.13.3.3.2.1.7	sar	374
2.13.3.3.2.1.8	datr	375
2.13.3.3.2.1.9	dar	375
2.13.3.3.2.1.10	bcr	375
2.13.3.3.2.1.11	enlndar	375
2.13.3.3.2.1.12	nlndar	375
2.13.3.3.2.1.13	eclsdar	375
2.13.3.3.2.1.14	clsdar	375
2.13.3.3.2.1.15	enlsdar	375
2.13.3.3.2.1.16	nlsdar	375
2.13.3.3.2.1.17	ssr	375
2.13.3.3.2.1.18	dsr	375
2.13.3.3.2.2	struct ocn_dma_memmap_t	375
2.13.3.3.2.2.1	Field Documentation	376
2.13.3.3.2.2.2	dma_ch_map1	376
2.13.3.3.2.2.3	dgsr0	376
2.13.3.3.2.2.4	dma_ch_map2	376
2.13.3.3.2.2.5	dgsr1	376
2.13.3.4	CPRI Memory Map	376
2.13.3.4.1	Overview	376
2.13.3.4.2	Data Structure Documentation	376
2.13.3.4.2.1	struct cpri_framer_registers_t	376
2.13.3.4.2.1.1	Field Documentation	377
2.13.3.4.2.1.2	cpri_status	377

Section number	Title	Page
2.13.3.4.2.1.3	cpri_config	377
2.13.3.4.2.1.4	cpri_lcv	378
2.13.3.4.2.1.5	cpri_bfn	378
2.13.3.4.2.1.6	cpri_hfn	378
2.13.3.4.2.1.7	cpri_hw_reset	378
2.13.3.4.2.1.8	cpri_cm_config	378
2.13.3.4.2.1.9	cpri_cm_status	378
2.13.3.4.2.1.10	cpri_rx_delay	378
2.13.3.4.2.1.11	cpri_round_delay	378
2.13.3.4.2.1.12	cpri_ex_delay_config	378
2.13.3.4.2.1.13	cpri_ex_delay_status	378
2.13.3.4.2.1.14	cpri_tx_prot_ver	378
2.13.3.4.2.1.15	cpri_tx_scr_seed	378
2.13.3.4.2.1.16	cpri_rx_scr_seed	378
2.13.3.4.2.1.17	cpri_serdes_config	379
2.13.3.4.2.1.18	cpri_map_config	379
2.13.3.4.2.1.19	cpri_map_cnt_config	379
2.13.3.4.2.1.20	cpri_map_tbl_config	379
2.13.3.4.2.1.21	cpri_map_offset_rx	379
2.13.3.4.2.1.22	cpri_map_offset_tx	379
2.13.3.4.2.1.23	cpri_start_offset_tx	379
2.13.3.4.2.1.24	cpri_iq_rx_buf_status1	379
2.13.3.4.2.1.25	cpri_iq_rx_buf_status2	379
2.13.3.4.2.1.26	cpri_iq_tx_buf_status1	379
2.13.3.4.2.1.27	cpri_iq_tx_buf_status2	379
2.13.3.4.2.1.28	eth_rx_status	379
2.13.3.4.2.1.29	eth_config_1	379
2.13.3.4.2.1.30	eth_config_2	380
2.13.3.4.2.1.31	eth_rx_control	380
2.13.3.4.2.1.32	eth_rx_ex_status	380
2.13.3.4.2.1.33	eth_addr_msb	380
2.13.3.4.2.1.34	eth_addr_lsb	380
2.13.3.4.2.1.35	eth_hash_table	380
2.13.3.4.2.1.36	eth_config_3	380
2.13.3.4.2.1.37	eth_cnt_rx_frame	380
2.13.3.4.2.1.38	eth_cnt_tx_frame	380
2.13.3.4.2.1.39	hdlc_rx_status	380
2.13.3.4.2.1.40	hdlc_config_1	380
2.13.3.4.2.1.41	hdlc_config_2	380
2.13.3.4.2.1.42	hdlc_rx_control	380
2.13.3.4.2.1.43	hdlc_rx_ex_status	381
2.13.3.4.2.1.44	hdlc_config_3	381
2.13.3.4.2.1.45	hdlc_cnt_rx_frame	381
2.13.3.4.2.1.46	hdlc_cnt_tx_frame	381
2.13.3.4.2.2	struct cpri_complex_registers_t	381

Section number	Title	Page
2.13.3.4.2.2.1	Field Documentation	382
2.13.3.4.2.2.2	riqmts	382
2.13.3.4.2.2.3	riqsdmts	382
2.13.3.4.2.2.4	tiqmts	382
2.13.3.4.2.2.5	rvssmts	382
2.13.3.4.2.2.6	tvssmts	382
2.13.3.4.2.2.7	riqsdba	382
2.13.3.4.2.2.8	riqbs	382
2.13.3.4.2.2.9	riqsdbs	382
2.13.3.4.2.2.10	tiqbs	382
2.13.3.4.2.2.11	rvssbs	382
2.13.3.4.2.2.12	tvssbs	382
2.13.3.4.2.2.13	rethbs	383
2.13.3.4.2.2.14	rhdlcbs	383
2.13.3.4.2.2.15	rvssba	383
2.13.3.4.2.2.16	tvssba	383
2.13.3.4.2.2.17	rebdrba	383
2.13.3.4.2.2.18	tebdrba	383
2.13.3.4.2.2.19	rhbdrba	383
2.13.3.4.2.2.20	thbdrba	383
2.13.3.4.2.2.21	rebdrs	383
2.13.3.4.2.2.22	tebdrs	383
2.13.3.4.2.2.23	rhbdrs	383
2.13.3.4.2.2.24	thbdrs	383
2.13.3.4.2.2.25	rgcm	383
2.13.3.4.2.2.26	tgcm	384
2.13.3.4.2.2.27	tscr	384
2.13.3.4.2.2.28	tcfbs	384
2.13.3.4.2.2.29	tctie1	384
2.13.3.4.2.2.30	tctie2	384
2.13.3.4.2.2.31	tmrc	384
2.13.3.4.2.2.32	rfpw	384
2.13.3.4.2.2.33	tfpw	384
2.13.3.4.2.3	struct cpri_control_registers_t	384
2.13.3.4.2.3.1	Field Documentation	385
2.13.3.4.2.3.2	rcr	385
2.13.3.4.2.3.3	tcr	385
2.13.3.4.2.3.4	raccr	385
2.13.3.4.2.3.5	taccr	385
2.13.3.4.2.3.6	rca	385
2.13.3.4.2.3.7	rcd	385
2.13.3.4.2.3.8	tca	385
2.13.3.4.2.3.9	tcd	385
2.13.3.4.2.3.10	riqft	386
2.13.3.4.2.3.11	riqst	386

Section number	Title	Page
2.13.3.4.2.3.12	riqt	386
2.13.3.4.2.3.13	tiqft	386
2.13.3.4.2.3.14	tiqst	386
2.13.3.4.2.3.15	tiqt	386
2.13.3.4.2.3.16	rvsst	386
2.13.3.4.2.3.17	tvsst	386
2.13.3.4.2.3.18	rethct	386
2.13.3.4.2.3.19	tethct	386
2.13.3.4.2.3.20	rcier	386
2.13.3.4.2.3.21	tcier	386
2.13.3.4.2.3.22	riqtsd	386
2.13.3.4.2.3.23	eier	387
2.13.3.4.2.3.24	tmre	387
2.13.3.4.2.3.25	rewpr	387
2.13.3.4.2.3.26	tewpr	387
2.13.3.4.2.3.27	rhwpr	387
2.13.3.4.2.3.28	thwpr	387
2.13.3.4.2.3.29	racpr	387
2.13.3.4.2.3.30	tacpr	387
2.13.3.4.2.3.31	maskr	387
2.13.3.4.2.3.32	auxcr	387
2.13.3.4.2.4	struct cpri_status_registers_t	387
2.13.3.4.2.4.1	Field Documentation	388
2.13.3.4.2.4.2	riqbdr	388
2.13.3.4.2.4.3	riqsdbdr	388
2.13.3.4.2.4.4	tiqbdr	388
2.13.3.4.2.4.5	rccr	388
2.13.3.4.2.4.6	rvssbdr	388
2.13.3.4.2.4.7	tvssbdr	388
2.13.3.4.2.4.8	rethbd1	389
2.13.3.4.2.4.9	rethbd2	389
2.13.3.4.2.4.10	tethbd1	389
2.13.3.4.2.4.11	tethbd2	389
2.13.3.4.2.4.12	rerpr	389
2.13.3.4.2.4.13	terpr	389
2.13.3.4.2.4.14	rhdlcbd1	389
2.13.3.4.2.4.15	rhdlcbd2	389
2.13.3.4.2.4.16	thdlcbd1	389
2.13.3.4.2.4.17	thdlcbd2	389
2.13.3.4.2.4.18	rhrpr	389
2.13.3.4.2.4.19	thrpr	389
2.13.3.4.2.4.20	rer	389
2.13.3.4.2.4.21	ter	390
2.13.3.4.2.4.22	eer	390
2.13.3.4.2.4.23	rethcs	390

Section number	Title	Page
2.13.3.4.2.4.24	tethcs	390
2.13.3.4.2.4.25	tmrsr	390
2.13.3.4.2.4.26	rsr	390
2.13.3.4.2.4.27	tsr	390
2.13.3.4.2.5	struct cpri_configuration_memories_t	390
2.13.3.4.2.5.1	Field Documentation	390
2.13.3.4.2.5.2	rcm	390
2.13.3.4.2.5.3	tcm	390
2.13.3.4.2.6	struct cpri_general_registers_t	391
2.13.3.4.2.6.1	Field Documentation	391
2.13.3.4.2.6.2	ccr	391
2.13.3.4.2.6.3	ICR	391
2.13.3.4.2.6.4	rccier	391
2.13.3.4.2.6.5	tccier	391
2.13.3.4.2.6.6	grsr	391
2.13.3.4.2.6.7	gtsr	391
2.13.3.4.2.6.8	iesr	391
2.13.3.4.2.6.9	cpridmastop	391
2.13.3.4.2.6.10	cpridmastopst	392
2.13.3.4.2.6.11	cprirrocr	392
2.13.3.4.2.7	struct cpri_unit_map_t	392
2.13.3.4.2.7.1	Field Documentation	392
2.13.3.4.2.7.2	cpri_framer	392
2.13.3.4.2.7.3	cpri_complex	392
2.13.3.4.2.7.4	cpri_control	392
2.13.3.4.2.7.5	cpri_status	392
2.13.3.4.2.7.6	cpri_configuration_memories	392
2.13.3.4.2.7.7	cpri_general_registers	392
2.13.3.5	I2C Memory Map	393
2.13.3.5.1	Overview	393
2.13.3.5.2	Data Structure Documentation	393
2.13.3.5.2.1	struct i2c_map_t	393
2.13.3.5.2.1.1	Field Documentation	393
2.13.3.5.2.1.2	i2cadr	393
2.13.3.5.2.1.3	i2cfdr	393
2.13.3.5.2.1.4	i2ccr	393
2.13.3.5.2.1.5	i2csr	393
2.13.3.5.2.1.6	i2cdr	393
2.13.3.5.2.1.7	i2cdfsrr	393
2.13.3.6	Maple-B3 Memory Map	394
2.13.3.6.1	Overview	394
2.13.3.6.2	Data Structure Documentation	395
2.13.3.6.2.1	struct maple_pe_bd_params_t	395
2.13.3.6.2.1.1	Field Documentation	395
2.13.3.6.2.1.2	brpap	395

Section number	Title	Page
2.13.3.6.2.1.3	brpbp	395
2.13.3.6.2.2	struct maple_pe_bd_ring_params_t	395
2.13.3.6.2.2.1	Field Documentation	395
2.13.3.6.2.2.2	high_bd	395
2.13.3.6.2.2.3	low_bd	395
2.13.3.6.2.3	struct maple_pe_bd_steering_t	395
2.13.3.6.2.3.1	Field Documentation	396
2.13.3.6.2.3.2	tidstp	396
2.13.3.6.2.4	struct maple_pe_bd_ring_steering_t	396
2.13.3.6.2.4.1	Field Documentation	396
2.13.3.6.2.4.2	high_bd	396
2.13.3.6.2.4.3	low_bd	396
2.13.3.6.2.5	struct maple_pe_mpisr_tid_t	396
2.13.3.6.2.6	union ftpedss_t	396
2.13.3.6.2.6.1	Field Documentation	396
2.13.3.6.2.6.2	ftpedsspy	396
2.13.3.6.2.7	struct maple_mpisr_p_t	396
2.13.3.6.2.7.1	Field Documentation	397
2.13.3.6.2.7.2	mpisr_ack_address	397
2.13.3.6.2.7.3	mpisr_ack_value	397
2.13.3.6.2.7.4	all_tsk_link	397
2.13.3.6.2.8	struct maple_lte_pram_map_t	397
2.13.3.6.2.8.1	Field Documentation	399
2.13.3.6.2.8.2	mbdrcp0	399
2.13.3.6.2.8.3	mbdrcp1	399
2.13.3.6.2.8.4	mbdrcp2	400
2.13.3.6.2.8.5	mucvp	400
2.13.3.6.2.8.6	mp_tpp	400
2.13.3.6.2.8.7	mcgcp	400
2.13.3.6.2.8.8	mmc0p	400
2.13.3.6.2.8.9	mmc1p	400
2.13.3.6.2.8.10	mtvcp	400
2.13.3.6.2.8.11	cdomcp	400
2.13.3.6.2.8.12	cubmsp	400
2.13.3.6.2.8.13	mpdschcp	400
2.13.3.6.2.8.14	mpuschcp	400
2.13.3.6.2.8.15	mcrrcip	400
2.13.3.6.2.8.16	tvpe_bd	401
2.13.3.6.2.8.17	ftpe_bd	401
2.13.3.6.2.8.18	depe_bd	401
2.13.3.6.2.8.19	crcpe_bd	401
2.13.3.6.2.8.20	eqpe_bd	401
2.13.3.6.2.8.21	convpe_bd	401
2.13.3.6.2.8.22	pusch_bd	401
2.13.3.6.2.8.23	pdsch_bd	401

Section number	Title	Page
2.13.3.6.2.8.24	pufft_bd	401
2.13.3.6.2.8.25	tvpe_steering	401
2.13.3.6.2.8.26	ftpe_steering	401
2.13.3.6.2.8.27	depe_steering	401
2.13.3.6.2.8.28	crcpe_steering	402
2.13.3.6.2.8.29	eqpe_steering	402
2.13.3.6.2.8.30	convpe_steering	402
2.13.3.6.2.8.31	pusch_steering	402
2.13.3.6.2.8.32	pdsch_high_bd_steering	402
2.13.3.6.2.8.33	mpisr_tid	402
2.13.3.6.2.8.34	pdsch_low_bd_steering	402
2.13.3.6.2.8.35	pufft_steering	402
2.13.3.6.2.8.36	mtvpvhcp	402
2.13.3.6.2.8.37	mtvpvlcp	402
2.13.3.6.2.8.38	tvpe_puncture_vector	402
2.13.3.6.2.8.39	pper	402
2.13.3.6.2.8.40	mtvppcp	403
2.13.3.6.2.8.41	tvpe_puncture_period	403
2.13.3.6.2.8.42	mtvpvsc0p	403
2.13.3.6.2.8.43	mtvpvsc1p	403
2.13.3.6.2.8.44	mtvpvscp	403
2.13.3.6.2.8.45	mpdsch_bf_config	403
2.13.3.6.2.8.46	ftpedss	403
2.13.3.6.2.8.47	uprmbpp	403
2.13.3.6.2.8.48	upsmbpp	403
2.13.3.6.2.8.49	ubsp	403
2.13.3.6.2.8.50	multiply_ftpe	403
2.13.3.6.2.8.51	ftpecubrp	403
2.13.3.6.2.8.52	mpdsch_ant_config_2nd_set	404
2.13.3.6.2.8.53	mtrcecp	404
2.13.3.6.2.8.54	mtrcwpp	404
2.13.3.6.2.8.55	mpufftcp	404
2.13.3.6.2.8.56	mpufftpmtap	404
2.13.3.6.2.8.57	mpufftfcbr	404
2.13.3.6.2.8.58	mpufftfcbi	404
2.13.3.6.2.8.59	mpufftfcsr	404
2.13.3.6.2.8.60	mpufftfcsi	404
2.13.3.6.2.8.61	mpufftpstmtap	404
2.13.3.6.2.8.62	mpdsch_pad_data	404
2.13.3.6.2.8.63	mpdsch_pad_data1	404
2.13.3.6.2.8.64	mpdsch_pad_data2	405
2.13.3.6.2.8.65	mpdsch_sec_config	405
2.13.3.6.2.8.66	mpdsch_ant_config	405
2.13.3.6.2.8.67	mpdsch_cell_config	405
2.13.3.6.2.8.68	mpdsch_sec_config_2nd_set	405

Section number	Title	Page
2.13.3.6.2.8.69	mpdsch_cell_config_2nd_set	405
2.13.3.6.2.8.70	mpdsch_start_qse0p	405
2.13.3.6.2.8.71	mpdsch_pdpe_qse0p	405
2.13.3.6.2.8.72	mpdsch_done_qse0p	405
2.13.3.6.2.8.73	mpdsch_start_qse1p	405
2.13.3.6.2.8.74	mpdsch_pdpe_qse1p	405
2.13.3.6.2.8.75	mpdsch_done_qse1p	406
2.13.3.6.2.8.76	mpdsch_start_qse2p	406
2.13.3.6.2.8.77	mpdsch_pdpe_qse2p	406
2.13.3.6.2.8.78	mpdsch_done_qse2p	406
2.13.3.6.2.8.79	mpdsch_start_qse3p	406
2.13.3.6.2.8.80	mpdsch_pdpe_qse3p	406
2.13.3.6.2.8.81	mpdsch_done_qse3p	406
2.13.3.6.2.8.82	mpdsch_start_qs0wp	406
2.13.3.6.2.8.83	mpdsch_pdpe_qs0wp	406
2.13.3.6.2.8.84	mpdsch_done_qs0wp	406
2.13.3.6.2.8.85	mpdsch_start_qs1wp	406
2.13.3.6.2.8.86	mpdsch_pdpe_qs1wp	406
2.13.3.6.2.8.87	mpdsch_done_qs1wp	406
2.13.3.6.2.8.88	mpdsch_start_qs2wp	407
2.13.3.6.2.8.89	mpdsch_pdpe_qs2wp	407
2.13.3.6.2.8.90	mpdsch_done_qs2wp	407
2.13.3.6.2.8.91	mpdsch_start_qs3wp	407
2.13.3.6.2.8.92	mpdsch_pdpe_qs3wp	407
2.13.3.6.2.8.93	mpdsch_done_qs3wp	407
2.13.3.6.2.8.94	mpdsch_start_qs0rp	407
2.13.3.6.2.8.95	mpdsch_pdpe_qs0rp	407
2.13.3.6.2.8.96	mpdsch_done_qs0rp	407
2.13.3.6.2.8.97	mpdsch_start_qs1rp	407
2.13.3.6.2.8.98	mpdsch_pdpe_qs1rp	407
2.13.3.6.2.8.99	mpdsch_done_qs1rp	407
2.13.3.6.2.8.100	mpdsch_start_qs2rp	407
2.13.3.6.2.8.101	mpdsch_pdpe_qs2rp	408
2.13.3.6.2.8.102	mpdsch_done_qs2rp	408
2.13.3.6.2.8.103	mpdsch_start_qs3rp	408
2.13.3.6.2.8.104	mpdsch_pdpe_qs3rp	408
2.13.3.6.2.8.105	mpdsch_done_qs3rp	408
2.13.3.6.2.8.106	mpdsch_s0np	408
2.13.3.6.2.8.107	mpdsch_s1np	408
2.13.3.6.2.8.108	mpdsch_s2np	408
2.13.3.6.2.8.109	mpdsch_s3np	408
2.13.3.6.2.8.110	maple_xxpe_acc	408
2.13.3.6.2.8.111	mpdsch_bf_config_2nd_set	408
2.13.3.6.2.8.112	mpdschoba_2nd_set	409
2.13.3.6.2.8.113	mpdschobs_2nd_set	409

Section number	Title	Page
2.13.3.6.2.8.114	mpdsch_ack	409
2.13.3.6.2.8.115	mpdschoba	409
2.13.3.6.2.8.116	mpdschobs	409
2.13.3.6.2.8.117	mpufftiba	409
2.13.3.6.2.8.118	mpufftibs	409
2.13.3.6.2.9	struct maple_wcdma_pram_map_t	409
2.13.3.6.2.9.1	Field Documentation	410
2.13.3.6.2.9.2	mbdrcp0	410
2.13.3.6.2.9.3	mbdrcp1	410
2.13.3.6.2.9.4	mucvp	410
2.13.3.6.2.9.5	mp_tpp	410
2.13.3.6.2.9.6	mcgcp	411
2.13.3.6.2.9.7	mmc0p	411
2.13.3.6.2.9.8	mmc1p	411
2.13.3.6.2.9.9	mtvmp	411
2.13.3.6.2.9.10	cdomcp	411
2.13.3.6.2.9.11	cubmsp	411
2.13.3.6.2.9.12	mcrrcip	411
2.13.3.6.2.9.13	ftpe_bd	411
2.13.3.6.2.9.14	tcpe_bd	411
2.13.3.6.2.9.15	crcpe_bd	411
2.13.3.6.2.9.16	cgpe_bd	411
2.13.3.6.2.9.17	pufft_bd	411
2.13.3.6.2.9.18	ftpe_steering	412
2.13.3.6.2.9.19	tcpe_steering	412
2.13.3.6.2.9.20	crcpe_steering	412
2.13.3.6.2.9.21	cgpe_steering	412
2.13.3.6.2.9.22	pufft_steering	412
2.13.3.6.2.9.23	ftpedss	412
2.13.3.6.2.9.24	uprmbpp	412
2.13.3.6.2.9.25	upsmbpp	412
2.13.3.6.2.9.26	ubsp	412
2.13.3.6.2.9.27	multiply_ftpe	412
2.13.3.6.2.9.28	ftpecubrp	412
2.13.3.6.2.9.29	mpisr_tid	412
2.13.3.6.2.9.30	mpufticp	413
2.13.3.6.2.9.31	mpufftpmtap	413
2.13.3.6.2.9.32	mpufftfcbr	413
2.13.3.6.2.9.33	mpufftfcbi	413
2.13.3.6.2.9.34	mpufftfcsr	413
2.13.3.6.2.9.35	mpufftfcsi	413
2.13.3.6.2.9.36	mpufftpstmtap	413
2.13.3.6.2.9.37	mpisr_param	413
2.13.3.6.2.9.38	mpisr_cnt_int_to_task	413
2.13.3.6.2.9.39	mpufftiba	413

Section number	Title	Page
2.13.3.6.2.9.40	mpufftibs	414
2.13.3.6.2.10	struct maple_lte_psif_dram_map_t	414
2.13.3.6.2.10.1	Member Function Documentation	414
2.13.3.6.2.10.2	OS_MEM_RESERVED	414
2.13.3.6.2.10.3	Field Documentation	414
2.13.3.6.2.10.4	pram	414
2.13.3.6.2.10.5	bd_rings	414
2.13.3.6.2.10.6	mpdschdcp	414
2.13.3.6.2.10.7	mpisr_param	414
2.13.3.6.2.10.8	mpisr_cnt_int_to_task	414
2.13.3.6.2.10.9	smartdsp_os_reserved	415
2.13.3.6.2.11	struct maple_wcdma_psif_dram_map_t	415
2.13.3.6.2.11.1	Field Documentation	415
2.13.3.6.2.11.2	pram	415
2.13.3.6.2.11.3	bd_rings	415
2.13.3.6.2.11.4	smartdsp_os_reserved	415
2.13.3.6.2.12	struct maple_tvpe_map_t	415
2.13.3.6.2.12.1	Field Documentation	415
2.13.3.6.2.12.2	tvaqcr	415
2.13.3.6.2.13	struct maple_ftpe_map_t	415
2.13.3.6.2.13.1	Field Documentation	416
2.13.3.6.2.13.2	ftpedsr0	416
2.13.3.6.2.13.3	ftpedsr1	416
2.13.3.6.2.13.4	ftpedsr2	416
2.13.3.6.2.13.5	ftpecr	416
2.13.3.6.2.13.6	ftpestr	416
2.13.3.6.2.14	struct maple_depe_map_t	416
2.13.3.6.2.15	struct maple_eqpe_map_t	416
2.13.3.6.2.15.1	Field Documentation	416
2.13.3.6.2.15.2	eq_eccevent	416
2.13.3.6.2.16	struct maple_pdpe_map_t	416
2.13.3.6.2.16.1	Field Documentation	417
2.13.3.6.2.16.2	pdpe_pesr	417
2.13.3.6.2.16.3	pdpe_sec_conf	417
2.13.3.6.2.17	struct maple_lte_mbus_memmap_t	417
2.13.3.6.2.17.1	Field Documentation	417
2.13.3.6.2.17.2	tvpe	417
2.13.3.6.2.17.3	ftpe	417
2.13.3.6.2.17.4	depe	417
2.13.3.6.2.17.5	eqpe	417
2.13.3.6.2.17.6	pdpe	417
2.13.3.6.2.18	struct maple_wcdma_mbus_memmap_t	417
2.13.3.6.2.18.1	Field Documentation	418
2.13.3.6.2.18.2	ftpe	418
2.13.3.6.2.19	struct maple_lte_sbus_memmap_t	418

Section number	Title	Page
2.13.3.6.2.19.1	Field Documentation	418
2.13.3.6.2.19.2	m_dalr	418
2.13.3.6.2.19.3	m_damr	418
2.13.3.6.2.19.4	m_ecr	418
2.13.3.6.2.19.5	m_ear	419
2.13.3.6.2.19.6	m_cr	419
2.13.3.6.2.19.7	m_dsdval	419
2.13.3.6.2.19.8	m_dsda_pl	419
2.13.3.6.2.19.9	m_dsdb_pl	419
2.13.3.6.2.19.10	m_dsdc_pl	419
2.13.3.6.2.19.11	m_dsdm_pl	419
2.13.3.6.2.19.12	m_dmper	419
2.13.3.6.2.19.13	m_dsda	419
2.13.3.6.2.19.14	m_dsdb	419
2.13.3.6.2.19.15	m_dsdc	419
2.13.3.6.2.19.16	m_dsdd	419
2.13.3.6.2.19.17	pcr	420
2.13.3.6.2.19.18	pspicer0	420
2.13.3.6.2.19.19	pspicer1	420
2.13.3.6.2.19.20	pspicer2	420
2.13.3.6.2.19.21	pspicelr	420
2.13.3.6.2.19.22	pspicmr0	420
2.13.3.6.2.19.23	pspicmr1	420
2.13.3.6.2.19.24	pspicmr2	420
2.13.3.6.2.19.25	pspiciacr	420
2.13.3.6.2.19.26	mpp_dst	420
2.13.3.6.2.19.27	clk_counter	420
2.13.3.6.3	Macro Definition Documentation	420
2.13.3.6.3.1	PRAM_BD_RING_SIZE	420
2.13.3.6.3.2	MAPLE_TRACE_BUFFER_SIZE	421
2.13.3.6.3.3	FTPE_DSS_NUM	421
2.13.3.6.3.4	TVPE_VITERBI_POLY_SETS_NUM	421
2.13.3.6.3.5	PUFFT_SOC_TIMER_SETS	421
2.13.3.6.3.6	PDSCH_SOC_TIMER_SETS	421
2.13.3.6.3.7	MAPLE3LW_FTPE_NUM	421
2.13.3.6.3.8	MAPLE3W_FTPE_NUM	421
2.13.3.6.3.9	MAPLE_STP_NUM	421
2.13.3.6.3.10	MAPLE_NUM_SLOTS	421
2.13.3.6.3.11	MAPLE_NUM_SECTORS	421
2.13.3.6.3.12	MAPLE_NUM_ANT	421
2.13.3.6.3.13	MAPLE_MAX_NUM_PE_BD_RINGS_LOW_PRIORITY	421
2.13.3.6.3.14	MAPLE_MAX_NUM_PE_BD_RINGS_HIGH_PRIORITY	421
2.13.3.6.3.15	MAPLE_PE_MAX_NUM_BD_RINGS	422
2.13.3.6.4	Typedef Documentation	422
2.13.3.6.4.1	maple_wcdma_sbus_memmap_t	422

Section number	Title	Page
2.13.3.7	sRIO Memory Map	422
2.13.3.7.1	Overview	422
2.13.3.7.2	Data Structure Documentation	422
2.13.3.7.2.1	struct rapidio_arch_map_t	422
2.13.3.7.2.1.1	Field Documentation	423
2.13.3.7.2.1.2	didcar	423
2.13.3.7.2.1.3	dicar	423
2.13.3.7.2.1.4	aidcar	423
2.13.3.7.2.1.5	aicar	423
2.13.3.7.2.1.6	pefcar	423
2.13.3.7.2.1.7	socar	423
2.13.3.7.2.1.8	docar	423
2.13.3.7.2.1.9	dsicar	423
2.13.3.7.2.1.10	dsllcsr	423
2.13.3.7.2.1.11	pellccsr	423
2.13.3.7.2.1.12	lcsba1csr	423
2.13.3.7.2.1.13	bdidcsr	424
2.13.3.7.2.1.14	hbdidlcsr	424
2.13.3.7.2.1.15	ctcsr	424
2.13.3.7.2.2	struct rapidio_port_status_map_t	424
2.13.3.7.2.2.1	Field Documentation	424
2.13.3.7.2.2.2	lmreqcsr	424
2.13.3.7.2.2.3	lmrespcsr	424
2.13.3.7.2.2.4	lascsr	424
2.13.3.7.2.2.5	escsr	424
2.13.3.7.2.2.6	ccsr	424
2.13.3.7.2.3	struct rapidio_1xlp_serial_map_t	424
2.13.3.7.2.3.1	Field Documentation	425
2.13.3.7.2.3.2	pmbh0	425
2.13.3.7.2.3.3	pltoccsr	425
2.13.3.7.2.3.4	prtoccsr	425
2.13.3.7.2.3.5	pgccsr	425
2.13.3.7.2.3.6	port_status	425
2.13.3.7.2.4	struct rapidio_err_log_map_t	425
2.13.3.7.2.4.1	Field Documentation	425
2.13.3.7.2.4.2	erbh	425
2.13.3.7.2.4.3	ltledcsr	426
2.13.3.7.2.4.4	ltleecsr	426
2.13.3.7.2.4.5	ltlaccsr	426
2.13.3.7.2.4.6	ltldidccsr	426
2.13.3.7.2.4.7	ltlcccsr	426
2.13.3.7.2.5	struct rapidio_err_phy_map_t	426
2.13.3.7.2.5.1	Field Documentation	426
2.13.3.7.2.5.2	edcsr	426
2.13.3.7.2.5.3	erecsr	426

Section number	Title	Page
2.13.3.7.2.5.4	ecacsr	426
2.13.3.7.2.5.5	pcseccsr	426
2.13.3.7.2.5.6	ercsr	427
2.13.3.7.2.5.7	ertcsr	427
2.13.3.7.2.6	struct rapidio_port_general_map_t	427
2.13.3.7.2.6.1	Field Documentation	427
2.13.3.7.2.6.2	pnadidcsr	427
2.13.3.7.2.6.3	pnaacr	427
2.13.3.7.2.6.4	pnlopttlcr	427
2.13.3.7.2.6.5	pniecsr	427
2.13.3.7.2.6.6	pnpcr	427
2.13.3.7.2.6.7	pnslcsr	427
2.13.3.7.2.6.8	pnsleicr	427
2.13.3.7.2.7	struct rapidio_general_map_t	428
2.13.3.7.2.7.1	Field Documentation	428
2.13.3.7.2.7.2	llcr	428
2.13.3.7.2.7.3	epwisr	428
2.13.3.7.2.7.4	lretcr	428
2.13.3.7.2.7.5	pretcr	428
2.13.3.7.2.8	struct rapidio_rev_map_t	428
2.13.3.7.2.8.1	Field Documentation	428
2.13.3.7.2.8.2	ipbrr	428
2.13.3.7.2.9	struct rapidio_atmu_out_t	428
2.13.3.7.2.9.1	Field Documentation	429
2.13.3.7.2.9.2	rowtar	429
2.13.3.7.2.9.3	rowtear	429
2.13.3.7.2.9.4	rowbar	429
2.13.3.7.2.9.5	rowar	429
2.13.3.7.2.9.6	rowsr	429
2.13.3.7.2.10	struct rapidio_atmu_in_t	429
2.13.3.7.2.10.1	Field Documentation	429
2.13.3.7.2.10.2	riwtar	429
2.13.3.7.2.10.3	riwbar	429
2.13.3.7.2.10.4	riwar	429
2.13.3.7.2.11	struct rapidio_atmu_map_t	429
2.13.3.7.2.12	struct srio_memmap_t	430
2.13.3.7.2.12.1	Field Documentation	430
2.13.3.7.2.12.2	rapidio_arch	430
2.13.3.7.2.12.3	rapidio_1xlp_serial	430
2.13.3.7.2.12.4	rapidio_err_log	430
2.13.3.7.2.12.5	rapidio_err_phy	430
2.13.3.7.2.12.6	rapidio_general	430
2.13.3.7.2.12.7	rapidio_rev	430
2.13.3.7.2.12.8	rapidio_atmu	430
2.13.3.8	Timers 16 bit Memory Map	431

Section number	Title	Page
2.13.3.8.1	Overview	431
2.13.3.8.2	Data Structure Documentation	431
2.13.3.8.2.1	struct timer_map_t	431
2.13.3.8.2.1.1	Field Documentation	431
2.13.3.8.2.1.2	tmr_cmp1	431
2.13.3.8.2.1.3	tmr_cmp2	431
2.13.3.8.2.1.4	tmr_cap	431
2.13.3.8.2.1.5	tmr_load	431
2.13.3.8.2.1.6	tmr_hold	431
2.13.3.8.2.1.7	tmr_cntr	432
2.13.3.8.2.1.8	tmr_ctrl	432
2.13.3.8.2.1.9	tmr_sctl	432
2.13.3.8.2.1.10	tmr_cmpld1	432
2.13.3.8.2.1.11	tmr_cmpld2	432
2.13.3.8.2.1.12	tmr_comscr	432
2.13.3.8.2.2	struct tmr_map_t	432
2.13.3.8.2.2.1	Field Documentation	432
2.13.3.8.2.2.2	tmr	432
2.13.3.9	Timers 32 bit Memory Map	432
2.13.3.9.1	Overview	432
2.13.3.9.2	Data Structure Documentation	433
2.13.3.9.2.1	struct timer_32b_map_t	433
2.13.3.9.2.1.1	Field Documentation	433
2.13.3.9.2.1.2	tmr_cmp1	433
2.13.3.9.2.1.3	tmr_cmp2	433
2.13.3.9.2.1.4	tmr_cap	433
2.13.3.9.2.1.5	tmr_load	433
2.13.3.9.2.1.6	tmr_hold	433
2.13.3.9.2.1.7	tmr_cntr	433
2.13.3.9.2.1.8	tmr_ctrl	433
2.13.3.9.2.1.9	tmr_sctl	434
2.13.3.9.2.1.10	tmr_cmpld1	434
2.13.3.9.2.1.11	tmr_cmpld2	434
2.13.3.9.2.1.12	tmr_comscr	434
2.13.3.9.2.2	struct tmr32b_map_t	434
2.13.3.9.2.2.1	Field Documentation	434
2.13.3.9.2.2.2	tmr	434
2.13.3.9.2.2.3	tmr_glb	434
2.13.3.9.2.2.4	tmr_glbctl	434
2.13.3.9.2.2.5	tmr_saf	434
2.13.3.9.2.2.6	tmr_clrl	434
2.13.3.10	Watchdog timer Memory Map	435
2.13.3.10.1	Overview	435
2.13.3.10.2	Data Structure Documentation	435
2.13.3.10.2.1	struct wdt_map_t	435

Section number	Title	Page
2.13.3.10.2.1.1	Field Documentation	435
2.13.3.10.2.1.2	swcrr	435
2.13.3.10.2.1.3	swcnr	435
2.13.3.10.2.1.4	swsrr	435
2.13.4	SC39XX Memory Maps	435
2.13.4.1	Overview	435
2.13.4.2	SC39XX Platform Memory Maps	436
2.13.4.2.1	Overview	436
2.13.4.2.2	Data Structure Documentation	436
2.13.4.2.2.1	struct dsp_plat_mmu_map_t	436
2.13.4.2.2.1.1	Field Documentation	437
2.13.4.2.2.1.2	m_cr	437
2.13.4.2.2.1.3	m_dvpc	437
2.13.4.2.2.1.4	m_dva	437
2.13.4.2.2.1.5	m_dsr	437
2.13.4.2.2.1.6	m_ndvr	437
2.13.4.2.2.1.7	m_ndsr	438
2.13.4.2.2.1.8	m_pir	438
2.13.4.2.2.1.9	m_dbl	438
2.13.4.2.2.1.10	m_dbe	438
2.13.4.2.2.1.11	m_desra0	438
2.13.4.2.2.1.12	m_desra1	438
2.13.4.2.2.1.13	m_desrs	438
2.13.4.2.2.1.14	msg_pir	438
2.13.4.2.2.1.15	msg_gpir	438
2.13.4.2.2.1.16	msg_lpidr	438
2.13.4.2.2.1.17	ccsr_base	438
2.13.4.2.2.1.18	m_dsdval	438
2.13.4.2.2.1.19	m_dsdmask	438
2.13.4.2.2.1.20	m_dsda_pl	439
2.13.4.2.2.1.21	m_dsdb_pl	439
2.13.4.2.2.1.22	m_dsdc_pl	439
2.13.4.2.2.1.23	m_dsdm_pl	439
2.13.4.2.2.1.24	m_dsda_pl_sb	439
2.13.4.2.2.1.25	m_dsdb_pl_sb	439
2.13.4.2.2.1.26	m_dsdc_pl_sb	439
2.13.4.2.2.1.27	m_dsdm_pl_sb	439
2.13.4.2.2.1.28	m_dmper	439
2.13.4.2.2.1.29	m_wmcfg	439
2.13.4.2.2.1.30	m_dsda	439
2.13.4.2.2.1.31	m_dsdb	439
2.13.4.2.2.1.32	m_dsdc	439
2.13.4.2.2.1.33	m_pva	440
2.13.4.2.2.1.34	m_psr	440
2.13.4.2.2.1.35	m_npvr	440

Section number	Title	Page
2.13.4.2.2.1.36	m_npsr	440
2.13.4.2.2.1.37	m_pesra0	440
2.13.4.2.2.1.38	m_pesra1	440
2.13.4.2.2.1.39	m_pesrs	440
2.13.4.2.2.1.40	m_psdval	440
2.13.4.2.2.1.41	m_psdmask	440
2.13.4.2.2.1.42	m_psda_pl	440
2.13.4.2.2.1.43	m_psdb_pl	440
2.13.4.2.2.1.44	m_psdc_pl	440
2.13.4.2.2.1.45	m_psdm_pl	440
2.13.4.2.2.1.46	m_psda_pl_sb	441
2.13.4.2.2.1.47	m_psdb_pl_sb	441
2.13.4.2.2.1.48	m_psdc_pl_sb	441
2.13.4.2.2.1.49	m_psdm_pl_sb	441
2.13.4.2.2.1.50	m_pmper	441
2.13.4.2.2.1.51	m_psda	441
2.13.4.2.2.1.52	m_psdb	441
2.13.4.2.2.1.53	m_psdc	441
2.13.4.2.2.2	struct dsp_plat_cme_map_t	441
2.13.4.2.2.2.1	Field Documentation	442
2.13.4.2.2.2.2	cme_ctr	442
2.13.4.2.2.2.3	cme_dcc	442
2.13.4.2.2.2.4	cme_dca	442
2.13.4.2.2.2.5	cme_dcr	442
2.13.4.2.2.2.6	cme_qcr	442
2.13.4.2.2.2.7	cme_dqu1	442
2.13.4.2.2.2.8	cme_dqu2	442
2.13.4.2.2.2.9	cme_qcc	442
2.13.4.2.2.2.10	cme_qca	442
2.13.4.2.2.2.11	cme_qu1	443
2.13.4.2.2.2.12	cme_qu2	443
2.13.4.2.2.2.13	cme_cc	443
2.13.4.2.2.2.14	cme_cs	443
2.13.4.2.2.2.15	cme_ca	443
2.13.4.2.2.2.16	cme_cr	443
2.13.4.2.2.2.17	cme_dst	443
2.13.4.2.2.2.18	cme_der	443
2.13.4.2.2.2.19	cme_din	443
2.13.4.2.2.2.20	cme_dms	443
2.13.4.2.2.2.21	cme_dc	443
2.13.4.2.2.2.22	cme_ds	443
2.13.4.2.2.2.23	cme_da	443
2.13.4.2.2.2.24	cme_pst	444
2.13.4.2.2.2.25	cme_per	444
2.13.4.2.2.2.26	cme_pin	444

Section number	Title	Page
2.13.4.2.2.2.27	cme_pms	444
2.13.4.2.2.2.28	cme_pc	444
2.13.4.2.2.2.29	cme_ps	444
2.13.4.2.2.2.30	cme_pa	444
2.13.4.2.2.3	struct dsp_plat_epic_map_t	444
2.13.4.2.2.3.1	Field Documentation	444
2.13.4.2.2.3.2	p_ipl	444
2.13.4.2.2.3.3	p_disp	444
2.13.4.2.2.3.4	p_trgt	445
2.13.4.2.2.3.5	p_elr	445
2.13.4.2.2.3.6	p_ipr	445
2.13.4.2.2.3.7	p_endis	445
2.13.4.2.2.3.8	p_swii	445
2.13.4.2.2.3.9	p_di	445
2.13.4.2.2.4	struct run_control_map_t	445
2.13.4.2.2.4.1	Field Documentation	446
2.13.4.2.2.4.2	ccr	446
2.13.4.2.2.4.3	cccr	446
2.13.4.2.2.4.4	ccd	446
2.13.4.2.2.4.5	pc_next	446
2.13.4.2.2.4.6	rcr	446
2.13.4.2.2.4.7	dmeer	446
2.13.4.2.2.4.8	dmrsr	446
2.13.4.2.2.4.9	dmcsr	446
2.13.4.2.2.4.10	dhrrr	446
2.13.4.2.2.4.11	dmrrr	446
2.13.4.2.2.4.12	duicr	446
2.13.4.2.2.4.13	drasr	446
2.13.4.2.2.4.14	desr	447
2.13.4.2.2.4.15	sasr	447
2.13.4.2.2.4.16	dturev	447
2.13.4.2.2.5	struct detector_map_t	447
2.13.4.2.2.5.1	Field Documentation	447
2.13.4.2.2.5.2	ardcr	447
2.13.4.2.2.5.3	depcr	447
2.13.4.2.2.5.4	padrra	447
2.13.4.2.2.5.5	padrrb	447
2.13.4.2.2.6	struct address_and_data_detection_map_t	447
2.13.4.2.2.6.1	Field Documentation	448
2.13.4.2.2.6.2	detector	448
2.13.4.2.2.7	struct indirect_event_unit_map_t	448
2.13.4.2.2.7.1	Field Documentation	448
2.13.4.2.2.7.2	iectls	448
2.13.4.2.2.7.3	iect	448
2.13.4.2.2.7.4	ieutr	448

Section number	Title	Page
2.13.4.2.2.8	struct triad_unit_map_t	448
2.13.4.2.2.8.1	Field Documentation	448
2.13.4.2.2.8.2	ptcr	448
2.13.4.2.2.8.3	pcvr	448
2.13.4.2.2.8.4	pcsr	449
2.13.4.2.2.9	struct profiling_unit_map_t	449
2.13.4.2.2.9.1	Field Documentation	449
2.13.4.2.2.9.2	pccsr	449
2.13.4.2.2.9.3	triad	449
2.13.4.2.2.9.4	rccr0	449
2.13.4.2.2.9.5	rcvr0	449
2.13.4.2.2.9.6	rcrr0	449
2.13.4.2.2.9.7	rcsr0	449
2.13.4.2.2.10	struct trace_unit_map_t	449
2.13.4.2.2.10.1	Field Documentation	450
2.13.4.2.2.10.2	tc1	450
2.13.4.2.2.10.3	trsr	450
2.13.4.2.2.10.4	tc3	450
2.13.4.2.2.10.5	tc4	450
2.13.4.2.2.10.6	tpmcr	450
2.13.4.2.2.10.7	twmsk	450
2.13.4.2.2.10.8	tmdati	450
2.13.4.2.2.11	struct dsp_plat_dtu_map_t	450
2.13.4.2.2.12	struct dsp_plat_timer_t	450
2.13.4.2.2.12.1	Field Documentation	451
2.13.4.2.2.12.2	tm_tc	451
2.13.4.2.2.12.3	tm_tp	451
2.13.4.2.2.13	struct dsp_plat_timer_map_t	451
2.13.4.2.2.13.1	Field Documentation	451
2.13.4.2.2.13.2	platform_timer	451
2.13.4.2.2.13.3	tm_tv	451
2.13.4.2.2.13.4	tm_sc	451
2.13.4.2.2.13.5	tm_s	451
2.13.4.2.2.14	struct dsp_plat_map_t	451
2.13.4.2.2.14.1	Field Documentation	452
2.13.4.2.2.14.2	dcache	452
2.13.4.2.2.14.3	icache	452
2.13.4.2.2.14.4	dtu	452
2.13.4.2.2.14.5	mmu	452
2.13.4.2.2.14.6	cme	452
2.13.4.2.2.14.7	epic	452
2.13.4.2.2.14.8	timer	452
2.13.4.3	SC39XX Cluster Memory Maps	452
2.13.4.3.1	Overview	452
2.13.4.3.2	Data Structure Documentation	452

Section number	Title	Page
2.13.4.3.2.1	struct dsp_cluster_wdt_t	452
2.13.4.3.2.1.1	Field Documentation	453
2.13.4.3.2.1.2	wdtc	453
2.13.4.3.2.1.3	wdtpl0	453
2.13.4.3.2.1.4	wdtpl1	453
2.13.4.3.2.1.5	wdtsc	453
2.13.4.3.2.1.6	wdtsv0	453
2.13.4.3.2.1.7	wdtsv1	453
2.13.4.3.2.2	struct l2_partition_id_t	453
2.13.4.3.2.2.1	Field Documentation	453
2.13.4.3.2.2.2	l2pir	453
2.13.4.3.2.2.3	l2par	454
2.13.4.3.2.2.4	l2pwr	454
2.13.4.3.2.3	struct dsp_cluster_l2_map_t	454
2.13.4.3.2.3.1	Field Documentation	454
2.13.4.3.2.3.2	l2csr0	454
2.13.4.3.2.3.3	l2csr1	454
2.13.4.3.2.3.4	l2cfg0	454
2.13.4.3.2.3.5	l2errinjhi	454
2.13.4.3.2.3.6	l2errinjlo	454
2.13.4.3.2.3.7	l2errinjctl	455
2.13.4.3.2.3.8	l2captdatahi	455
2.13.4.3.2.3.9	l2captdatalo	455
2.13.4.3.2.3.10	l2captecc	455
2.13.4.3.2.3.11	l2errdet	455
2.13.4.3.2.3.12	l2errdis	455
2.13.4.3.2.3.13	l2errinten	455
2.13.4.3.2.3.14	l2errattr	455
2.13.4.3.2.3.15	l2erreaddr	455
2.13.4.3.2.3.16	l2erraddr	455
2.13.4.3.2.3.17	l2errctl	455
2.13.4.3.2.4	struct dsp_cluster_map_t	455
2.13.5	B4860 LAW	456
2.13.5.1	Overview	456
2.13.5.2	Data Structure Documentation	456
2.13.5.2.1	struct law_init_params_t	456
2.13.5.2.1.1	Field Documentation	457
2.13.5.2.1.1.1	num_windows	457
2.13.5.3	Enumeration Type Documentation	457
2.13.5.3.1	law_win_size_t	457
2.13.5.3.2	law_win_id_t	458
2.13.5.3.3	law_target_id_t	458
2.13.5.4	Function Documentation	458
2.13.5.4.1	osLawInitialize	458
2.13.5.4.2	b486xLawWinAdd	458

Section number	Title	Page
2.13.5.4.3	b486xLawWinDisable	459
2.13.5.4.4	b486xLawWinFindTid	459
2.13.6	DSP Platform Definitions	459
2.13.6.1	Overview	459
2.13.7	SmartDsp OS Devices	460
2.13.7.1	Overview	460
2.13.8	Hardware Semaphore API	460
2.13.8.1	Overview	460
2.13.8.2	Macro Definition Documentation	460
2.13.8.2.1	hwSemaphoreIsTaken	460
2.13.8.2.2	hwSemaphoreIsFree	461
2.13.8.3	Function Documentation	461
2.13.8.3.1	hwSemaphoreTake	461
2.13.8.3.2	hwSemaphoreIrqTake	461
2.13.8.3.3	hwSemaphoreRelease	461
2.13.8.3.4	hwSemaphoreIrqRelease	462
2.13.8.3.5	hwSemaphoreTryTake	462
2.13.8.3.6	hwSemaphoreGetVal	462
2.13.8.3.7	hwSemaphoreInitialize	463
2.13.9	GPIO (general purpose I/O) API	463
2.13.10	IPC (Inter Process Communication) API	463
2.13.10.1	Overview	463
2.13.11	Heterogeneous API	463
2.13.11.1	Overview	463
2.13.12	HW Interrupts Map	464
2.13.12.1	Overview	464
2.13.12.2	B4860 HW Interrupts	464
2.13.12.2.1	Overview	464
2.13.12.2.2	Macro Definition Documentation	472
2.13.12.2.2.1	OS_INT_FROM_CORE0_0	472
2.13.12.2.2.2	OS_INT_FROM_CORE0_1	472
2.13.12.2.2.3	OS_INT_FROM_CORE1_0	472
2.13.12.2.2.4	OS_INT_FROM_CORE1_1	472
2.13.12.2.2.5	OS_INT_FROM_CORE2_0	472
2.13.12.2.2.6	OS_INT_FROM_CORE2_1	472
2.13.12.2.2.7	OS_INT_FROM_CORE3_0	472
2.13.12.2.2.8	OS_INT_FROM_CORE3_1	472
2.13.12.2.2.9	OS_INT_FROM_CORE4_0	473
2.13.12.2.2.10	OS_INT_FROM_CORE4_1	473
2.13.12.2.2.11	OS_INT_FROM_CORE5_0	473
2.13.12.2.2.12	OS_INT_FROM_CORE5_1	473
2.13.12.2.2.13	OS_INT_VIRQ0	473
2.13.12.2.2.14	OS_INT_VIRQ1	473
2.13.12.2.2.15	OS_INT_VIRQ2	473
2.13.12.2.2.16	OS_INT_VIRQ3	473

Section number	Title	Page
2.13.12.2.2.17	OS_INT_VIRQ4	473
2.13.12.2.2.18	OS_INT_VIRQ5	473
2.13.12.2.2.19	OS_INT_VIRQ6	473
2.13.12.2.2.20	OS_INT_VIRQ7	473
2.13.12.2.2.21	OS_INT_VIRQ8	473
2.13.12.2.2.22	OS_INT_VIRQ9	474
2.13.12.2.2.23	OS_INT_VIRQ10	474
2.13.12.2.2.24	OS_INT_VIRQ11	474
2.13.12.2.2.25	OS_INT_VIRQ12	474
2.13.12.2.2.26	OS_INT_VIRQ13	474
2.13.12.2.2.27	OS_INT_VIRQ14	474
2.13.12.2.2.28	OS_INT_VIRQ15	474
2.13.12.2.2.29	OS_INT_VIRQ16	474
2.13.12.2.2.30	OS_INT_VIRQ17	474
2.13.12.2.2.31	OS_INT_VIRQ18	474
2.13.12.2.2.32	OS_INT_VIRQ19	474
2.13.12.2.2.33	OS_INT_VIRQ20	474
2.13.12.2.2.34	OS_INT_VIRQ21	474
2.13.12.2.2.35	OS_INT_VIRQ22	475
2.13.12.2.2.36	OS_INT_VIRQ23	475
2.13.12.2.2.37	OS_INT_VIRQ24	475
2.13.12.2.2.38	OS_INT_VIRQ25	475
2.13.12.2.2.39	OS_INT_VIRQ26	475
2.13.12.2.2.40	OS_INT_VIRQ27	475
2.13.12.2.2.41	OS_INT_VIRQ28	475
2.13.12.2.2.42	OS_INT_VIRQ29	475
2.13.12.2.2.43	OS_INT_VIRQ30	475
2.13.12.2.2.44	OS_INT_VIRQ31	475
2.13.12.2.2.45	OS_INT_MAPLE_0_CH_0	475
2.13.12.2.2.46	OS_INT_MAPLE_0_CH_1	475
2.13.12.2.2.47	OS_INT_MAPLE_0_CH_2	475
2.13.12.2.2.48	OS_INT_MAPLE_0_CH_3	476
2.13.12.2.2.49	OS_INT_MAPLE_0_CH_4	476
2.13.12.2.2.50	OS_INT_MAPLE_0_CH_5	476
2.13.12.2.2.51	OS_INT_MAPLE_0_CH_6	476
2.13.12.2.2.52	OS_INT_MAPLE_0_CH_7	476
2.13.12.2.2.53	OS_INT_MAPLE_0_CH_8	476
2.13.12.2.2.54	OS_INT_MAPLE_0_CH_9	476
2.13.12.2.2.55	OS_INT_MAPLE_0_CH_10	476
2.13.12.2.2.56	OS_INT_MAPLE_0_CH_11	476
2.13.12.2.2.57	OS_INT_MAPLE_0_CH_12	476
2.13.12.2.2.58	OS_INT_MAPLE_0_CH_13	476
2.13.12.2.2.59	OS_INT_MAPLE_0_CH_14	476
2.13.12.2.2.60	OS_INT_MAPLE_0_CH_15	476
2.13.12.2.2.61	OS_INT_MAPLE_0_CH_16	477

Section number	Title	Page
2.13.12.2.2.62	OS_INT_MAPLE_0_CH_17	477
2.13.12.2.2.63	OS_INT_MAPLE_0_CH_18	477
2.13.12.2.2.64	OS_INT_MAPLE_0_CH_19	477
2.13.12.2.2.65	OS_INT_MAPLE_0_CH_20	477
2.13.12.2.2.66	OS_INT_MAPLE_0_CH_21	477
2.13.12.2.2.67	OS_INT_MAPLE_0_CH_22	477
2.13.12.2.2.68	OS_INT_MAPLE_0_CH_23	477
2.13.12.2.2.69	OS_INT_MAPLE_0_CH_24	477
2.13.12.2.2.70	OS_INT_MAPLE_0_CH_25	477
2.13.12.2.2.71	OS_INT_MAPLE_0_CH_26	477
2.13.12.2.2.72	OS_INT_MAPLE_0_CH_27	477
2.13.12.2.2.73	OS_INT_MAPLE_0_CH_28	477
2.13.12.2.2.74	OS_INT_MAPLE_0_CH_29	478
2.13.12.2.2.75	OS_INT_MAPLE_0_CH_30	478
2.13.12.2.2.76	OS_INT_MAPLE_0_CH_31	478
2.13.12.2.2.77	OS_INT_MAPLE_1_CH_0	478
2.13.12.2.2.78	OS_INT_MAPLE_1_CH_1	478
2.13.12.2.2.79	OS_INT_MAPLE_1_CH_2	478
2.13.12.2.2.80	OS_INT_MAPLE_1_CH_3	478
2.13.12.2.2.81	OS_INT_MAPLE_1_CH_4	478
2.13.12.2.2.82	OS_INT_MAPLE_1_CH_5	478
2.13.12.2.2.83	OS_INT_MAPLE_1_CH_6	478
2.13.12.2.2.84	OS_INT_MAPLE_1_CH_7	478
2.13.12.2.2.85	OS_INT_MAPLE_1_CH_8	478
2.13.12.2.2.86	OS_INT_MAPLE_1_CH_9	478
2.13.12.2.2.87	OS_INT_MAPLE_1_CH_10	479
2.13.12.2.2.88	OS_INT_MAPLE_1_CH_11	479
2.13.12.2.2.89	OS_INT_MAPLE_1_CH_12	479
2.13.12.2.2.90	OS_INT_MAPLE_1_CH_13	479
2.13.12.2.2.91	OS_INT_MAPLE_1_CH_14	479
2.13.12.2.2.92	OS_INT_MAPLE_1_CH_15	479
2.13.12.2.2.93	OS_INT_MAPLE_1_CH_16	479
2.13.12.2.2.94	OS_INT_MAPLE_1_CH_17	479
2.13.12.2.2.95	OS_INT_MAPLE_1_CH_18	479
2.13.12.2.2.96	OS_INT_MAPLE_1_CH_19	479
2.13.12.2.2.97	OS_INT_MAPLE_1_CH_20	479
2.13.12.2.2.98	OS_INT_MAPLE_1_CH_21	479
2.13.12.2.2.99	OS_INT_MAPLE_1_CH_22	479
2.13.12.2.2.100	OS_INT_MAPLE_1_CH_23	480
2.13.12.2.2.101	OS_INT_MAPLE_1_CH_24	480
2.13.12.2.2.102	OS_INT_MAPLE_1_CH_25	480
2.13.12.2.2.103	OS_INT_MAPLE_1_CH_26	480
2.13.12.2.2.104	OS_INT_MAPLE_1_CH_27	480
2.13.12.2.2.105	OS_INT_MAPLE_1_CH_28	480
2.13.12.2.2.106	OS_INT_MAPLE_1_CH_29	480

Section number	Title	Page
2.13.12.2.2.107	OS_INT_MAPLE_1_CH_30	480
2.13.12.2.2.108	OS_INT_MAPLE_1_CH_31	480
2.13.12.2.2.109	OS_INT_MAPLE_2_CH_0	480
2.13.12.2.2.110	OS_INT_MAPLE_2_CH_1	480
2.13.12.2.2.111	OS_INT_MAPLE_2_CH_2	480
2.13.12.2.2.112	OS_INT_MAPLE_2_CH_3	480
2.13.12.2.2.113	OS_INT_MAPLE_2_CH_4	481
2.13.12.2.2.114	OS_INT_MAPLE_2_CH_5	481
2.13.12.2.2.115	OS_INT_MAPLE_2_CH_6	481
2.13.12.2.2.116	OS_INT_MAPLE_2_CH_7	481
2.13.12.2.2.117	OS_INT_MAPLE_2_CH_8	481
2.13.12.2.2.118	OS_INT_MAPLE_2_CH_9	481
2.13.12.2.2.119	OS_INT_MAPLE_2_CH_10	481
2.13.12.2.2.120	OS_INT_MAPLE_2_CH_11	481
2.13.12.2.2.121	OS_INT_MAPLE_2_CH_12	481
2.13.12.2.2.122	OS_INT_MAPLE_2_CH_13	481
2.13.12.2.2.123	OS_INT_MAPLE_2_CH_14	481
2.13.12.2.2.124	OS_INT_MAPLE_2_CH_15	481
2.13.12.2.2.125	OS_INT_MAPLE_2_CH_16	481
2.13.12.2.2.126	OS_INT_MAPLE_2_CH_17	482
2.13.12.2.2.127	OS_INT_MAPLE_2_CH_18	482
2.13.12.2.2.128	OS_INT_MAPLE_2_CH_19	482
2.13.12.2.2.129	OS_INT_MAPLE_2_CH_20	482
2.13.12.2.2.130	OS_INT_MAPLE_2_CH_21	482
2.13.12.2.2.131	OS_INT_MAPLE_2_CH_22	482
2.13.12.2.2.132	OS_INT_MAPLE_2_CH_23	482
2.13.12.2.2.133	OS_INT_MAPLE_2_CH_24	482
2.13.12.2.2.134	OS_INT_MAPLE_2_CH_25	482
2.13.12.2.2.135	OS_INT_MAPLE_2_CH_26	482
2.13.12.2.2.136	OS_INT_MAPLE_2_CH_27	482
2.13.12.2.2.137	OS_INT_MAPLE_2_CH_28	482
2.13.12.2.2.138	OS_INT_MAPLE_2_CH_29	482
2.13.12.2.2.139	OS_INT_MAPLE_2_CH_30	483
2.13.12.2.2.140	OS_INT_MAPLE_2_CH_31	483
2.13.12.2.2.141	OS_INT_MAPLE_2_SYS_ERR	483
2.13.12.2.2.142	OS_INT_MAPLE_2_ECC_ERR	483
2.13.12.2.2.143	OS_INT_MAPLE_0_SYS_ERR	483
2.13.12.2.2.144	OS_INT_MAPLE_0_ECC_ERR	483
2.13.12.2.2.145	OS_INT_MAPLE_1_SYS_ERR	483
2.13.12.2.2.146	OS_INT_MAPLE_1_ECC_ERR	483
2.13.12.2.2.147	OS_INT_CPRI_0	483
2.13.12.2.2.148	OS_INT_CPRI_1	483
2.13.12.2.2.149	OS_INT_CPRI_2	483
2.13.12.2.2.150	OS_INT_CPRI_3	483
2.13.12.2.2.151	OS_INT_CPRI_4	483

Section number	Title	Page
2.13.12.2.2.152	OS_INT_CPRI_5	484
2.13.12.2.2.153	OS_INT_CPRI_6	484
2.13.12.2.2.154	OS_INT_CPRI_7	484
2.13.12.2.2.155	OS_INT_CPRI_8	484
2.13.12.2.2.156	OS_INT_CPRI_9	484
2.13.12.2.2.157	OS_INT_CPRI_10	484
2.13.12.2.2.158	OS_INT_CPRI_11	484
2.13.12.2.2.159	OS_INT_CPRI_12	484
2.13.12.2.2.160	OS_INT_CPRI_13	484
2.13.12.2.2.161	OS_INT_CPRI_14	484
2.13.12.2.2.162	OS_INT_CPRI_15	484
2.13.12.2.2.163	OS_INT_CPRI_0_RX_CTRL	484
2.13.12.2.2.164	OS_INT_CPRI_1_RX_CTRL	484
2.13.12.2.2.165	OS_INT_CPRI_2_RX_CTRL	485
2.13.12.2.2.166	OS_INT_CPRI_3_RX_CTRL	485
2.13.12.2.2.167	OS_INT_CPRI_4_RX_CTRL	485
2.13.12.2.2.168	OS_INT_CPRI_5_RX_CTRL	485
2.13.12.2.2.169	OS_INT_CPRI_6_RX_CTRL	485
2.13.12.2.2.170	OS_INT_CPRI_7_RX_CTRL	485
2.13.12.2.2.171	OS_INT_CPRI_0_TX_CTRL	485
2.13.12.2.2.172	OS_INT_CPRI_1_TX_CTRL	485
2.13.12.2.2.173	OS_INT_CPRI_2_TX_CTRL	485
2.13.12.2.2.174	OS_INT_CPRI_3_TX_CTRL	485
2.13.12.2.2.175	OS_INT_CPRI_4_TX_CTRL	485
2.13.12.2.2.176	OS_INT_CPRI_5_TX_CTRL	485
2.13.12.2.2.177	OS_INT_CPRI_6_TX_CTRL	485
2.13.12.2.2.178	OS_INT_CPRI_7_TX_CTRL	486
2.13.12.2.2.179	OS_INT_CPRI_0_RX_TIMING	486
2.13.12.2.2.180	OS_INT_CPRI_1_RX_TIMING	486
2.13.12.2.2.181	OS_INT_CPRI_2_RX_TIMING	486
2.13.12.2.2.182	OS_INT_CPRI_3_RX_TIMING	486
2.13.12.2.2.183	OS_INT_CPRI_4_RX_TIMING	486
2.13.12.2.2.184	OS_INT_CPRI_5_RX_TIMING	486
2.13.12.2.2.185	OS_INT_CPRI_6_RX_TIMING	486
2.13.12.2.2.186	OS_INT_CPRI_7_RX_TIMING	486
2.13.12.2.2.187	OS_INT_CPRI_0_TX_TIMING	486
2.13.12.2.2.188	OS_INT_CPRI_1_TX_TIMING	486
2.13.12.2.2.189	OS_INT_CPRI_2_TX_TIMING	486
2.13.12.2.2.190	OS_INT_CPRI_3_TX_TIMING	486
2.13.12.2.2.191	OS_INT_CPRI_4_TX_TIMING	487
2.13.12.2.2.192	OS_INT_CPRI_5_TX_TIMING	487
2.13.12.2.2.193	OS_INT_CPRI_6_TX_TIMING	487
2.13.12.2.2.194	OS_INT_CPRI_7_TX_TIMING	487
2.13.12.2.2.195	OS_INT_CPRI_GEN_ERR	487
2.13.12.2.2.196	OS_INT_EPU1	487

Section number	Title	Page
2.13.12.2.2.197	OS_INT_EPU2	487
2.13.12.2.2.198	OS_INT_SOC_TIMER_0_CH_0	487
2.13.12.2.2.199	OS_INT_SOC_TIMER_0_CH_1	487
2.13.12.2.2.200	OS_INT_SOC_TIMER_0_CH_2	487
2.13.12.2.2.201	OS_INT_SOC_TIMER_0_CH_3	487
2.13.12.2.2.202	OS_INT_SOC_TIMER_1_CH_0	487
2.13.12.2.2.203	OS_INT_SOC_TIMER_1_CH_1	487
2.13.12.2.2.204	OS_INT_SOC_TIMER_1_CH_2	488
2.13.12.2.2.205	OS_INT_SOC_TIMER_1_CH_3	488
2.13.12.2.2.206	OS_INT_SOC_TIMER_2_CH_0	488
2.13.12.2.2.207	OS_INT_SOC_TIMER_2_CH_1	488
2.13.12.2.2.208	OS_INT_SOC_TIMER_2_CH_2	488
2.13.12.2.2.209	OS_INT_SOC_TIMER_2_CH_3	488
2.13.12.2.2.210	OS_INT_SOC_TIMER_3_CH_0	488
2.13.12.2.2.211	OS_INT_SOC_TIMER_3_CH_1	488
2.13.12.2.2.212	OS_INT_SOC_TIMER_3_CH_2	488
2.13.12.2.2.213	OS_INT_SOC_TIMER_3_CH_3	488
2.13.12.2.2.214	OS_INT_SOC_TIMER_4_CH_0	488
2.13.12.2.2.215	OS_INT_SOC_TIMER_4_CH_1	488
2.13.12.2.2.216	OS_INT_SOC_TIMER_4_CH_2	488
2.13.12.2.2.217	OS_INT_SOC_TIMER_4_CH_3	489
2.13.12.2.2.218	OS_INT_SOC_TIMER_5_CH_0	489
2.13.12.2.2.219	OS_INT_SOC_TIMER_5_CH_1	489
2.13.12.2.2.220	OS_INT_SOC_TIMER_5_CH_2	489
2.13.12.2.2.221	OS_INT_SOC_TIMER_5_CH_3	489
2.13.12.2.2.222	OS_INT_SOC_TIMER_6_CH_0	489
2.13.12.2.2.223	OS_INT_SOC_TIMER_6_CH_1	489
2.13.12.2.2.224	OS_INT_SOC_TIMER_6_CH_2	489
2.13.12.2.2.225	OS_INT_SOC_TIMER_6_CH_3	489
2.13.12.2.2.226	OS_INT_SOC_TIMER_7_CH_0	489
2.13.12.2.2.227	OS_INT_SOC_TIMER_7_CH_1	489
2.13.12.2.2.228	OS_INT_SOC_TIMER_7_CH_2	489
2.13.12.2.2.229	OS_INT_SOC_TIMER_7_CH_3	489
2.13.12.2.2.230	OS_INT_SOC_TIMER_8_CH_0	490
2.13.12.2.2.231	OS_INT_SOC_TIMER_8_CH_1	490
2.13.12.2.2.232	OS_INT_SOC_TIMER_8_CH_2	490
2.13.12.2.2.233	OS_INT_SOC_TIMER_8_CH_3	490
2.13.12.2.2.234	OS_INT_SOC_TIMER_9_CH_0	490
2.13.12.2.2.235	OS_INT_SOC_TIMER_9_CH_1	490
2.13.12.2.2.236	OS_INT_SOC_TIMER_9_CH_2	490
2.13.12.2.2.237	OS_INT_SOC_TIMER_9_CH_3	490
2.13.12.2.2.238	OS_INT_SOC_TIMER_10_CH_0	490
2.13.12.2.2.239	OS_INT_SOC_TIMER_10_CH_1	490
2.13.12.2.2.240	OS_INT_SOC_TIMER_10_CH_2	490
2.13.12.2.2.241	OS_INT_SOC_TIMER_10_CH_3	490

Section number	Title	Page
2.13.12.2.2.242	OS_INT_SOC_TIMER_11_CH_0	490
2.13.12.2.2.243	OS_INT_SOC_TIMER_11_CH_1	491
2.13.12.2.2.244	OS_INT_SOC_TIMER_11_CH_2	491
2.13.12.2.2.245	OS_INT_SOC_TIMER_11_CH_3	491
2.13.12.2.2.246	OS_INT_SOC_TIMER_12_CH_0	491
2.13.12.2.2.247	OS_INT_SOC_TIMER_12_CH_1	491
2.13.12.2.2.248	OS_INT_SOC_TIMER_12_CH_2	491
2.13.12.2.2.249	OS_INT_SOC_TIMER_12_CH_3	491
2.13.12.2.2.250	OS_INT_SOC_TIMER_13_CH_0	491
2.13.12.2.2.251	OS_INT_SOC_TIMER_13_CH_1	491
2.13.12.2.2.252	OS_INT_SOC_TIMER_13_CH_2	491
2.13.12.2.2.253	OS_INT_SOC_TIMER_13_CH_3	491
2.13.12.2.2.254	OS_INT_SOC_TIMER_14_CH_0	491
2.13.12.2.2.255	OS_INT_SOC_TIMER_14_CH_1	491
2.13.12.2.2.256	OS_INT_SOC_TIMER_14_CH_2	492
2.13.12.2.2.257	OS_INT_SOC_TIMER_14_CH_3	492
2.13.12.2.2.258	OS_INT_SOC_TIMER_15_CH_0	492
2.13.12.2.2.259	OS_INT_SOC_TIMER_15_CH_1	492
2.13.12.2.2.260	OS_INT_SOC_TIMER_15_CH_2	492
2.13.12.2.2.261	OS_INT_SOC_TIMER_15_CH_3	492
2.13.12.2.2.262	OS_INT_GCR_TIMER_16	492
2.13.12.2.2.263	OS_INT_GCR_TIMER_17	492
2.13.12.2.2.264	OS_INT_GCR_TIMER_18	492
2.13.12.2.2.265	OS_INT_GCR_TIMER_19	492
2.13.12.2.2.266	OS_INT_GCR_TIMER_20	492
2.13.12.2.2.267	OS_INT_GCR_TIMER_21	492
2.13.12.2.2.268	OS_INT_GCR_TIMER_22	492
2.13.12.2.2.269	OS_INT_GCR_TIMER_23	493
2.13.12.2.2.270	OS_INT_GCR_TIMER_24	493
2.13.12.2.2.271	OS_INT_GCR_TIMER_25	493
2.13.12.2.2.272	OS_INT_GCR_TIMER_26	493
2.13.12.2.2.273	OS_INT_GCR_TIMER_27	493
2.13.12.2.2.274	OS_INT_GCR_TIMER_28	493
2.13.12.2.2.275	OS_INT_GCR_TIMER_29	493
2.13.12.2.2.276	OS_INT_GCR_TIMER_30	493
2.13.12.2.2.277	OS_INT_GCR_TIMER_31	493
2.13.12.2.2.278	OS_INT_TRAP0	493
2.13.12.2.2.279	OS_INT_TRAP1	493
2.13.12.2.2.280	OS_INT_ILLEGAL	493
2.13.12.2.2.281	OS_INT_IRRECOVERABLE	493
2.13.12.2.2.282	OS_INT_IMMUAE	494
2.13.12.2.2.283	OS_INT_DMMUAE	494
2.13.12.2.2.284	OS_INT_DEBUG0	494
2.13.12.2.2.285	OS_INT_DEBUG1	494
2.13.12.2.2.286	OS_INT_DEBUG2	494

Section number	Title	Page
2.13.12.2.2.287	OS_INT_DEBUG3	494
2.13.12.2.2.288	OS_INT_SWBE	494
2.13.12.2.2.289	OS_INT_RTEST	494
2.13.12.2.2.290	OS_INT_HWDE	494
2.13.12.2.2.291	OS_INT_PHWDE	494
2.13.12.2.2.292	EPIC_INTERRUPTS_OFFSET	494
2.13.12.2.2.293	OS_INT_CBE	494
2.13.12.2.2.294	OS_INT_WE	494
2.13.12.2.2.295	OS_INT_ICGE	495
2.13.12.2.2.296	OS_INT_L2E	495
2.13.12.2.2.297	OS_INT_AXI2ELINK0_ERR	495
2.13.12.2.2.298	OS_INT_AXI2ELINK1_ERR	495
2.13.12.2.2.299	OS_INT_CMS	495
2.13.12.2.2.300	OS_INT_CGMS	495
2.13.12.2.2.301	OS_INT_WDTI	495
2.13.12.2.2.302	OS_INT_MS	495
2.13.12.2.2.303	OS_INT_GMS	495
2.13.12.2.2.304	OS_INT_MMS	495
2.13.12.2.2.305	OS_INT_I_TM0	495
2.13.12.2.2.306	OS_INT_I_TM1	495
2.13.12.2.2.307	OS_INT_I_TM2	495
2.13.12.2.2.308	OS_INT_I_TM3	496
2.13.12.2.2.309	OS_INT_CBC	496
2.13.12.2.2.310	OS_INT_AXI2ELINK0_MON	496
2.13.12.2.2.311	OS_INT_AXI2ELINK1_MON	496
2.13.12.2.2.312	OS_INT_IRQ_0	496
2.13.12.2.2.313	OS_INT_IRQ_1	496
2.13.12.2.2.314	OS_INT_IRQ_2	496
2.13.12.2.2.315	OS_INT_IRQ_3	496
2.13.12.2.2.316	OS_INT_IRQ_4	496
2.13.12.2.2.317	OS_INT_IRQ_5	496
2.13.12.2.2.318	OS_INT_IRQ_6	496
2.13.12.2.2.319	OS_INT_IRQ_7	496
2.13.12.2.2.320	OS_INT_IRQ_8	496
2.13.12.2.2.321	OS_INT_IRQ_9	497
2.13.12.2.2.322	OS_INT_IRQ_10	497
2.13.12.2.2.323	OS_INT_IRQ_11	497
2.13.12.2.2.324	OS_INT_MPIC_CI	497
2.13.12.2.2.325	OS_INT_MPIC	497
2.13.12.2.2.326	EPIC_EDGE_INTERRUPT_START	497
2.13.12.2.2.327	EPIC_EDGE_INTERRUPT_END	497
2.13.12.2.2.328	EPIC_INTERNAL_INTERRUPT_END	497
2.13.12.2.2.329	OS_HWI_DEFAULT_DISPATCHER	497
2.13.12.2.2.330	OS_SWI_DEFALT_DISPATCHER	497
2.13.12.2.2.331	OS_TASK_DEFALT_DISPATCHER	497

Section number	Title	Page
2.13.13	B4860 L1 Defense	498
2.13.13.1	Overview	498
2.13.13.2	Data Structure Documentation	498
2.13.13.2.1	struct maple_allocated_buffers_t	498
2.13.13.2.1.1	Field Documentation	498
2.13.13.2.1.1.1	core_id	498
2.13.13.2.1.1.2	buffer	499
2.13.13.2.2	struct maple_data_t	499
2.13.13.2.3	struct l1d_data_t	499
2.13.13.2.3.1	Field Documentation	499
2.13.13.2.3.1.1	start_validation_value	499
2.13.13.2.3.1.2	maple_data	499
2.13.13.2.3.1.3	end_validation_value	499
2.13.13.2.4	struct l1d_init_params_t	499
2.13.13.2.4.1	Field Documentation	500
2.13.13.2.4.1.1	enabled_modes	500
2.13.13.2.4.1.2	hwi_num	500
2.13.13.2.5	struct l1d_clean_params_t	500
2.13.13.2.5.1	Field Documentation	500
2.13.13.2.5.1.1	l2_cache	500
2.13.13.2.5.1.2	maple	500
2.13.13.2.5.1.3	cpri	500
2.13.13.2.5.1.4	mpic_dsp	500
2.13.13.2.5.1.5	debug_print	500
2.13.13.2.5.1.6	ocn_dma	501
2.13.13.2.6	struct maple3_clean_param_t	501
2.13.13.3	Typedef Documentation	501
2.13.13.3.1	maple3_config_func_t	501
2.13.13.4	Function Documentation	501
2.13.13.4.1	osL1dResetFlowSet	501
2.13.13.4.2	osL1dInitialize	501
2.13.13.4.3	osL1dResetRequest	502
2.13.13.4.4	osL1dHardwareClean	502
2.13.13.4.5	osL1dGetMapleResetStatus	502
2.14	**MPIC Module API**	**502**
2.14.1	Overview	502
2.14.2	MPIC B486x Initialization API	503
2.14.2.1	Overview	503
2.14.2.2	Macro Definition Documentation	511
2.14.2.2.1	NUM_OF_MPIC_IPI_CORES	511
2.14.2.2.2	MPIC_NUM_OF_EXT_INTRS	511
2.14.2.2.3	MPIC_NUM_OF_INT_INTRS	511
2.14.2.2.4	MPIC_NUM_OF_TIMERS	511
2.14.2.2.5	MPIC_NUM_OF_MSG_INTRS	511

Section number	Title	Page
2.14.2.2.6	MPIC_NUM_OF_SMSG_INTRS	511
2.14.2.2.7	MPIC_NUM_OF_IPI_EVENTS	511
2.14.2.2.8	MPIC_NUM_OF_LOW_INT_INTRS	511
2.14.2.2.9	MPIC_MAX_NUM_OF_INTR_SRC	511
2.14.2.2.10	MPIC_MULTI_SRC_INTERNAL_INTERRUPTS	512
2.14.2.2.11	MPIC_MULTI_SRC_INTERNAL_INTERRUPTS_COUNT	512
2.14.2.2.12	MPIC_INTR	512
2.14.2.2.13	MPIC_INTR_TO_IDX	512
2.14.2.3	Enumeration Type Documentation	512
2.14.2.3.1	os_mpic_intr_groups_t	512
2.14.2.3.2	os_mpic_intr_id_t	512
2.14.3	MPIC runtime API	520
2.14.3.1	Overview	520
2.14.3.2	Data Structure Documentation	521
2.14.3.2.1	struct os_mpic_intr_params_t	521
2.14.3.2.1.1	Field Documentation	521
2.14.3.2.1.1.1	priority	521
2.14.3.2.1.1.2	target	521
2.14.3.2.1.1.3	sense	522
2.14.3.2.1.1.4	polarity	522
2.14.3.2.2	struct os_mpic_isr_entry_t	522
2.14.3.2.2.1	Field Documentation	522
2.14.3.2.2.1.1	isr	522
2.14.3.2.2.1.2	arg	522
2.14.3.3	Macro Definition Documentation	522
2.14.3.3.1	OS_VPR_PRIORITY_SET	522
2.14.3.3.2	OS_MPIC_DEFAULT_PRIORITY	522
2.14.3.3.3	OS_MPIC_DEFAULT_POLARITY	522
2.14.3.3.4	OS_MPIC_DEFAULT_SENSE	523
2.14.3.3.5	OS_MPIC_DEFAULT_TARGET	523
2.14.3.3.6	osMpicSetIntr	523
2.14.3.4	Enumeration Type Documentation	523
2.14.3.4.1	os_mpic_intr_target_t	523
2.14.3.4.2	os_mpic_intr_sense_t	523
2.14.3.4.3	os_mpic_intr_polarity_t	523
2.14.3.5	Function Documentation	524
2.14.3.5.1	osMpicSetConfigIntr	524
2.14.3.5.2	osMpicWriteMessage	524
2.14.3.5.3	osMpicReadMessage	525
2.14.3.5.4	osMpicReset	525
2.14.3.5.5	osMpicEnableIntr	526
2.14.3.5.6	osMpicDisableIntr	526
2.14.3.5.7	osMpicWriteMsi	526
2.14.3.5.8	osMpicMsiCoalescingConfig	526
2.14.3.5.9	osMpicReadMsiSignalRegister	527

Section number | Title | Page

2.14.3.5.10 osMpicFreeIntr . . . 527
2.14.3.5.11 osMpicFindMsiIntr . . . 527

Chapter 3
Drivers

3.1 Overview . . . 529

3.2 CPRI Module API . . . 529
3.2.1 Overview . . . 529
3.2.2 CPRI initialization . . . 529
3.2.2.1 Overview . . . 529
3.2.2.2 Data Structure Documentation . . . 534
3.2.2.2.1 struct cpri_multicast_params_t . . . 534
3.2.2.2.1.1 Field Documentation . . . 534
3.2.2.2.1.1.1 multicast_mode . . . 534
3.2.2.2.1.1.2 unicast_receive_transaction_size . . . 534
3.2.2.2.1.1.3 rx_buffer_size . . . 534
3.2.2.2.1.1.4 threshold . . . 534
3.2.2.2.2 struct mapping_table_entry_t . . . 534
3.2.2.2.2.1 Field Documentation . . . 535
3.2.2.2.2.1.1 width0 . . . 535
3.2.2.2.2.1.2 position0 . . . 535
3.2.2.2.2.1.3 axc0 . . . 535
3.2.2.2.2.1.4 enable0 . . . 535
3.2.2.2.2.1.5 width1 . . . 535
3.2.2.2.2.1.6 position1 . . . 535
3.2.2.2.2.1.7 axc1 . . . 535
3.2.2.2.2.1.8 enable1 . . . 535
3.2.2.2.3 struct mapping_params_t . . . 535
3.2.2.2.3.1 Field Documentation . . . 536
3.2.2.2.3.1.1 mapping_mode . . . 536
3.2.2.2.3.1.2 axc_oversampling_factor . . . 536
3.2.2.2.3.1.3 mapping_table_configuration . . . 536
3.2.2.2.3.1.4 rx_mapping_table . . . 536
3.2.2.2.3.1.5 tx_mapping_table . . . 536
3.2.2.2.3.1.6 tx_table_size . . . 536
3.2.2.2.3.1.7 rx_table_size . . . 536
3.2.2.2.4 struct cpri_iq_init_params_t . . . 536
3.2.2.2.4.1 Field Documentation . . . 537
3.2.2.2.4.1.1 tx_axc_number . . . 537
3.2.2.2.4.1.2 rx_axc_number . . . 537
3.2.2.2.4.1.3 participating_cores . . . 538

Section number	Title	Page
3.2.2.2.4.1.4	initializing_core	538
3.2.2.2.4.1.5	axc_tx_active	538
3.2.2.2.4.1.6	axc_rx_active	538
3.2.2.2.4.1.7	axc_tx_summing_active	538
3.2.2.2.4.1.8	axc_rx_forwarding_active	538
3.2.2.2.4.1.9	axc_tx_active_dma	538
3.2.2.2.4.1.10	axc_rx_active_dma	538
3.2.2.2.4.1.11	auxiliary_mode	538
3.2.2.2.4.1.12	transmit_sync_active	538
3.2.2.2.4.1.13	transmit_iq_sync	538
3.2.2.2.4.1.14	receive_iq_sync	538
3.2.2.2.4.1.15	map_tx_hf_resync	539
3.2.2.2.4.1.16	map_rx_hf_resync	539
3.2.2.2.4.1.17	start_tx_hf_resync	539
3.2.2.2.4.1.18	tx_shared_mode	539
3.2.2.2.4.1.19	dual_bandwidth_mode	539
3.2.2.2.4.1.20	mapping_params	539
3.2.2.2.4.1.21	multicast_params	539
3.2.2.2.4.1.22	sample_width	539
3.2.2.2.4.1.23	tx_transaction_size	539
3.2.2.2.4.1.24	rx_transaction_size	539
3.2.2.2.4.1.25	rx_buffer_size	539
3.2.2.2.4.1.26	tx_buffer_size	539
3.2.2.2.4.1.27	tx_first_threshold	539
3.2.2.2.4.1.28	tx_second_threshold	540
3.2.2.2.4.1.29	rx_first_threshold	540
3.2.2.2.4.1.30	rx_second_threshold	540
3.2.2.2.4.1.31	tx_threshold	540
3.2.2.2.4.1.32	rx_threshold	540
3.2.2.2.4.1.33	map_tx_offset_z	540
3.2.2.2.4.1.34	map_tx_offset_x	540
3.2.2.2.4.1.35	map_rx_offset_z	540
3.2.2.2.4.1.36	map_rx_offset_x	540
3.2.2.2.4.1.37	start_tx_offset_z	540
3.2.2.2.4.1.38	start_tx_offset_x	540
3.2.2.2.4.1.39	auxiliary_mask	540
3.2.2.2.4.1.40	auxiliary_mask_size	540
3.2.2.2.4.1.41	rx_symbol_awareness_int	541
3.2.2.2.4.1.42	rx_2nd_dest_symbol_awareness_int	541
3.2.2.2.4.1.43	tx_symbol_awareness_int	541
3.2.2.2.4.1.44	warm_reset_enable	541
3.2.2.2.4.1.45	transmit_iq_sample_count	541
3.2.2.2.5	struct cpri_iq_additional_params_t	541
3.2.2.2.5.1	Field Documentation	541
3.2.2.2.5.1.1	rx_transparent_mode	541

Section number	Title	Page
3.2.2.2.5.1.2	tx_transparent_mode	541
3.2.2.2.6	struct cpri_vss_init_params_t	541
3.2.2.2.6.1	Field Documentation	542
3.2.2.2.6.1.1	initializing_core	542
3.2.2.2.6.1.2	tx_transaction_size	542
3.2.2.2.6.1.3	rx_transaction_size	542
3.2.2.2.6.1.4	tx_buffer_size	542
3.2.2.2.6.1.5	rx_buffer_size	542
3.2.2.2.6.1.6	tx_threshold	542
3.2.2.2.6.1.7	rx_threshold	542
3.2.2.2.7	struct cpri_ethernet_init_params_t	542
3.2.2.2.7.1	Field Documentation	543
3.2.2.2.7.1.1	initializing_core	543
3.2.2.2.7.1.2	tx_crc_enable	543
3.2.2.2.7.1.3	rx_crc_enable	543
3.2.2.2.7.1.4	rx_long_ethernet_frame_ena	543
3.2.2.2.7.1.5	rx_illegal_preamble_dis_ena	543
3.2.2.2.7.1.6	rx_broadcast_enable	543
3.2.2.2.7.1.7	rx_multicast_enable	543
3.2.2.2.7.1.8	rx_mac_address_check	544
3.2.2.2.7.1.9	tx_store_and_forward_enable	544
3.2.2.2.7.1.10	rx_packet_length_chk_enable	544
3.2.2.2.7.1.11	little_endian_selection	544
3.2.2.2.7.1.12	tx_interrupt_en	544
3.2.2.2.7.1.13	tx_coherency_en	544
3.2.2.2.7.1.14	rx_interrupt_en	544
3.2.2.2.7.1.15	mac_address_lsb	544
3.2.2.2.7.1.16	mac_address_msb	544
3.2.2.2.7.1.17	rx_buffer_size	544
3.2.2.2.7.1.18	hash_filter	544
3.2.2.2.7.1.19	tx_bd_ring_size	544
3.2.2.2.7.1.20	rx_bd_ring_size	545
3.2.2.2.7.1.21	tx_coalescing_threshold	545
3.2.2.2.7.1.22	rx_coalescing_threshold	545
3.2.2.2.7.1.23	tx_heap	545
3.2.2.2.7.1.24	rx_heap	545
3.2.2.2.8	struct cpri_hdlc_init_params_t	545
3.2.2.2.8.1	Field Documentation	546
3.2.2.2.8.1.1	initializing_core	546
3.2.2.2.8.1.2	tx_crc_enable	546
3.2.2.2.8.1.3	rx_crc_enable	546
3.2.2.2.8.1.4	tx_store_and_forward_enable	546
3.2.2.2.8.1.5	rx_packet_length_chk_enable	546
3.2.2.2.8.1.6	little_endian_selection	546
3.2.2.2.8.1.7	tx_interrupt_en	546

Section number	Title	Page
3.2.2.2.8.1.8	tx_coherency_en	546
3.2.2.2.8.1.9	rx_interrupt_en	546
3.2.2.2.8.1.10	rx_buffer_size	546
3.2.2.2.8.1.11	tx_bd_ring_size	546
3.2.2.2.8.1.12	rx_bd_ring_size	546
3.2.2.2.8.1.13	tx_heap	547
3.2.2.2.8.1.14	rx_heap	547
3.2.2.2.9	struct cpri_init_params_t	547
3.2.2.2.9.1	Field Documentation	548
3.2.2.2.9.1.1	cpri_num	548
3.2.2.2.9.1.2	loop_mode	548
3.2.2.2.9.1.3	sync_mode	548
3.2.2.2.9.1.4	external_sync_active	548
3.2.2.2.9.1.5	shared_sync_mode_source	548
3.2.2.2.9.1.6	reset_ack_assert_length	548
3.2.2.2.9.1.7	reset_detection_length	548
3.2.2.2.9.1.8	rx_transparent_mode	548
3.2.2.2.9.1.9	tx_transparent_mode	548
3.2.2.2.9.1.10	transmit_sync_output	549
3.2.2.2.9.1.11	sync_state_machine_ena	549
3.2.2.2.9.1.12	shared_sync_mode	549
3.2.2.2.9.1.13	delay_direction	549
3.2.2.2.9.1.14	sync1588	549
3.2.2.2.9.1.15	remote_reset_indication	549
3.2.2.2.9.1.16	number_of_errors	549
3.2.2.2.9.1.17	tx_framer_buffer_size	549
3.2.2.2.9.1.18	vss_eth_portion_in_frame	549
3.2.2.2.9.1.19	tx_ex_delay	549
3.2.2.2.9.1.20	scarmbling_seed	549
3.2.2.2.9.1.21	timer_tolerance_value	550
3.2.2.2.9.1.22	transmit_delay	550
3.2.2.2.9.1.23	sync_delay	550
3.2.2.2.9.1.24	interrupts_table	550
3.2.2.2.9.1.25	extended_features	550
3.2.2.2.9.1.26	cpri_iq_init	550
3.2.2.2.9.1.27	cpri_vss_init	550
3.2.2.2.9.1.28	cpri_ethernet_init	550
3.2.2.2.9.1.29	cpri_hdlc_init	550
3.2.2.2.9.1.30	iq_init_params	550
3.2.2.2.9.1.31	vss_init_params	550
3.2.2.2.9.1.32	ethernet_init_params	551
3.2.2.2.9.1.33	hdlc_init_params	551
3.2.2.2.10	struct cpri_auxiliary_params_t	551
3.2.2.2.10.1	Field Documentation	551
3.2.2.2.10.1.1	mask	551

Section number	Title	Page
3.2.2.2.10.1.2	mask_size	551
3.2.2.3	Macro Definition Documentation	551
3.2.2.3.1	CPRI_IQ_DEV_NAME0	551
3.2.2.3.2	CPRI_IQ_DEV_NAME1	551
3.2.2.3.3	CPRI_IQ_DEV_NAME2	551
3.2.2.3.4	CPRI_IQ_DEV_NAME3	552
3.2.2.3.5	CPRI_IQ_DEV_NAME4	552
3.2.2.3.6	CPRI_IQ_DEV_NAME5	552
3.2.2.3.7	CPRI_VSS_DEV_NAME0	552
3.2.2.3.8	CPRI_VSS_DEV_NAME1	552
3.2.2.3.9	CPRI_VSS_DEV_NAME2	552
3.2.2.3.10	CPRI_VSS_DEV_NAME3	552
3.2.2.3.11	CPRI_VSS_DEV_NAME4	552
3.2.2.3.12	CPRI_VSS_DEV_NAME5	552
3.2.2.3.13	CPRI_ETHERNET_DEV_NAME0	552
3.2.2.3.14	CPRI_ETHERNET_DEV_NAME1	553
3.2.2.3.15	CPRI_ETHERNET_DEV_NAME2	553
3.2.2.3.16	CPRI_ETHERNET_DEV_NAME3	553
3.2.2.3.17	CPRI_ETHERNET_DEV_NAME4	553
3.2.2.3.18	CPRI_ETHERNET_DEV_NAME5	553
3.2.2.3.19	CPRI_HDLC_DEV_NAME0	553
3.2.2.3.20	CPRI_HDLC_DEV_NAME1	553
3.2.2.3.21	CPRI_HDLC_DEV_NAME2	553
3.2.2.3.22	CPRI_HDLC_DEV_NAME3	553
3.2.2.3.23	CPRI_HDLC_DEV_NAME4	553
3.2.2.3.24	CPRI_HDLC_DEV_NAME5	554
3.2.2.3.25	CPRI_IQ_ENABLED	554
3.2.2.3.26	CPRI_VSS_ENABLED	554
3.2.2.3.27	CPRI_ETHERNET_ENABLED	554
3.2.2.3.28	CPRI_HDLC_ENABLED	554
3.2.2.3.29	CPRI_IQ_DISABLED	554
3.2.2.3.30	CPRI_VSS_DISABLED	554
3.2.2.3.31	CPRI_ETHERNET_DISABLED	554
3.2.2.3.32	CPRI_HDLC_DISABLED	554
3.2.2.3.33	CPRI_NOT_ACTIVE	554
3.2.2.3.34	CPRI_ACTIVE	555
3.2.2.3.35	CPRI_NOT_SYNCHONIZED	555
3.2.2.3.36	CPRI_SYNCHONIZED	555
3.2.2.3.37	CPRI_IQ_ACTIVE	555
3.2.2.3.38	CPRI_ETHERNET_ACTIVE	555
3.2.2.3.39	CPRI_HDLC_ACTIVE	555
3.2.2.3.40	CPRI_VSS_ACTIVE	555
3.2.2.3.41	CPRI_ALL_DATA_TYPES_ACTIVE	555
3.2.2.3.42	CPRI_DEVICE_DELAYS_CALCULATE	555
3.2.2.3.43	CPRI_DEVICE_STATISTICS_SET	555

Section number	Title	Page
3.2.2.3.44	CPRI_DEVICE_TX_CONTROL_TABLE_WRITE	556
3.2.2.3.45	CPRI_DEVICE_TX_CONTROL_TABLE_READ	556
3.2.2.3.46	CPRI_DEVICE_RX_CONTROL_TABLE_READ	556
3.2.2.3.47	CPRI_DEVICE_RX_BFN_COUNTER_READ	556
3.2.2.3.48	CPRI_DEVICE_RX_HFN_COUNTER_READ	556
3.2.2.3.49	CPRI_DEVICE_TX_BFN_COUNTER_READ	556
3.2.2.3.50	CPRI_DEVICE_TX_HFN_COUNTER_READ	556
3.2.2.3.51	CPRI_DEVICE_RESET_REQUEST	556
3.2.2.3.52	CPRI_DEVICE_RESET_REQUEST_DISABLE	556
3.2.2.3.53	CPRI_DEVICE_RESET_ENABLE	556
3.2.2.3.54	CPRI_DEVICE_RESET_DISABLE	557
3.2.2.3.55	CPRI_DEVICE_CHECK_RESET_DETECTED	557
3.2.2.3.56	CPRI_DEVICE_TX_BFN_RESET	557
3.2.2.3.57	CPRI_DEVICE_DMA_RESTART	557
3.2.2.3.58	CPRI_DEVICE_RECONFIGURATION_LEVEL0	557
3.2.2.3.59	CPRI_DEVICE_RECONFIGURATION_LEVEL1	557
3.2.2.3.60	CPRI_DEVICE_RECONFIGURATION_LEVEL2	557
3.2.2.3.61	CPRI_STATUS_QUERY	557
3.2.2.3.62	CPRI_AXC_0_ACTIVE	557
3.2.2.3.63	CPRI_AXC_1_ACTIVE	558
3.2.2.3.64	CPRI_AXC_2_ACTIVE	558
3.2.2.3.65	CPRI_AXC_3_ACTIVE	558
3.2.2.3.66	CPRI_AXC_4_ACTIVE	558
3.2.2.3.67	CPRI_AXC_5_ACTIVE	558
3.2.2.3.68	CPRI_AXC_6_ACTIVE	558
3.2.2.3.69	CPRI_AXC_7_ACTIVE	558
3.2.2.3.70	CPRI_AXC_8_ACTIVE	558
3.2.2.3.71	CPRI_AXC_9_ACTIVE	558
3.2.2.3.72	CPRI_AXC_10_ACTIVE	558
3.2.2.3.73	CPRI_AXC_11_ACTIVE	559
3.2.2.3.74	CPRI_AXC_12_ACTIVE	559
3.2.2.3.75	CPRI_AXC_13_ACTIVE	559
3.2.2.3.76	CPRI_AXC_14_ACTIVE	559
3.2.2.3.77	CPRI_AXC_15_ACTIVE	559
3.2.2.3.78	CPRI_AXC_16_ACTIVE	559
3.2.2.3.79	CPRI_AXC_17_ACTIVE	559
3.2.2.3.80	CPRI_AXC_18_ACTIVE	559
3.2.2.3.81	CPRI_AXC_19_ACTIVE	559
3.2.2.3.82	CPRI_AXC_20_ACTIVE	559
3.2.2.3.83	CPRI_AXC_21_ACTIVE	560
3.2.2.3.84	CPRI_AXC_22_ACTIVE	560
3.2.2.3.85	CPRI_AXC_23_ACTIVE	560
3.2.2.3.86	MAXIMUM_NUMBER_OF_AXCS	560
3.2.2.3.87	CPRI_IQ_THRESHOLD_EVENT	560
3.2.2.3.88	CPRI_IQ_FIRST_THRESHOLD_EVENT	560

Section number	Title	Page
3.2.2.3.89	CPRI_IQ_SECOND_THRESHOLD_EVENT	560
3.2.2.3.90	CPRI_VSS_EVENT	560
3.2.2.3.91	CPRI_HDLC_EVENT	560
3.2.2.3.92	CPRI_ETHERNET_EVENT	560
3.2.2.3.93	CPRI_HFN_TIMING_EVENT	561
3.2.2.3.94	CPRI_BFN_TIMING_EVENT	561
3.2.2.3.95	CPRI_MULTICAST_RX_IQ_THRESHOLD_EVENT	561
3.2.2.3.96	CPRI_IQ_TX_BEGIN_EVENT	561
3.2.2.3.97	CPRI_ALL_TER_RER_EVENTS	561
3.2.2.3.98	CPRI_RECEIVE_IQ_OVERRUN_ERROR	561
3.2.2.3.99	CPRI_TRANSMIT_IQ_UNDERRUN_ERROR	561
3.2.2.3.100	CPRI_RECEIVE_ETH_MEMORY_OVERRUN_ERROR	561
3.2.2.3.101	CPRI_TRANSMIT_ETH_UNDERRUN_ERROR	561
3.2.2.3.102	CPRI_RECEIVE_ETH_BUFFER_DESCRIPTOR_UNDERRUN_ERROR	561
3.2.2.3.103	CPRI_RECEIVE_HDLC_OVERRUN_ERROR	562
3.2.2.3.104	CPRI_TRANSMIT_HDLC_UNDERRUN_ERROR	562
3.2.2.3.105	CPRI_RECEIVE_HDLC_BUFFER_DESCRIPTOR_UNDERRUN_ERROR	562
3.2.2.3.106	CPRI_RECEIVE_VSS_OVERRUN_ERROR	562
3.2.2.3.107	CPRI_TRANSMIT_VSS_UNDERRUN_ERROR	562
3.2.2.3.108	CPRI_EXTERNAL_SYNC_SYNCHRONIZATION_LOST_ERROR	562
3.2.2.3.109	CPRI_REMOTE_LOS_ERROR	562
3.2.2.3.110	CPRI_REMOTE_LOF_ERROR	562
3.2.2.3.111	CPRI_REMOTE_RAI_ERROR	562
3.2.2.3.112	CPRI_REMOTE_SDI_ERROR	562
3.2.2.3.113	CPRI_LOCAL_LOS_ERROR	563
3.2.2.3.114	CPRI_LOCAL_LOST_OF_FRAME_ERROR	563
3.2.2.3.115	CPRI_FREQUENCY_ALARM_ERROR	563
3.2.2.3.116	CPRI_REMOTE_RESET_ERROR	563
3.2.2.3.117	CPRI_ECC_CONFIGURATION_MEMORY_ERROR	563
3.2.2.3.118	CPRI_ECC_DATA_MEMORY_ERROR	563
3.2.2.3.119	CPRI_REMOTE_RESET_ACKNOWLEDGE	563
3.2.2.3.120	CPRI_ALL_ERRORS	563
3.2.2.3.121	CPRI_ERROR_MULTIBIT_ECC_SECOND_DESTINATION_DMA_MEMORY	563
3.2.2.3.122	CPRI_ERROR_MULTIBIT_ECC_COMBINER_MEMORY	563
3.2.2.3.123	CPRI_ERROR_MULTIBIT_ECC_FRAME_ELASTIC_BUFFER_MEMORY	564
3.2.2.3.124	CPRI_ERROR_MULTIBIT_ECC_AUX_MEMORY	564
3.2.2.3.125	CPRI_ERROR_MULTIBIT_ECC_FRAMER_TRANSMIT_HDLC_MEMORY	564
3.2.2.3.126	CPRI_ERROR_MULTIBIT_ECC_FRAMER_RECEIVE_HDLC_MEMORY	564

Section number	Title	Page
3.2.2.3.127	CPRI_ERROR_MULTIBIT_ECC_FRAMER_TRANSMIT_ETH_MEMORY	564
3.2.2.3.128	CPRI_ERROR_MULTIBIT_ECC_FRAMER_RECEIVE_ETH_MEMORY	564
3.2.2.3.129	CPRI_ERROR_MULTIBIT_ECC_TRANSMIT_CONTROL_WORD_TABLE_MEMORY	564
3.2.2.3.130	CPRI_ERROR_MULTIBIT_ECC_RECEIVE_CONTROL_WORD_TABLE_MEMORY	564
3.2.2.3.131	CPRI_ERROR_MULTIBIT_ECC_RECEIVE_DMA_MEMORY	565
3.2.2.3.132	CPRI_ERROR_MULTIBIT_ECC_FRAMER_TRANSMIT_IQ_MEMORY	565
3.2.2.3.133	CPRI_ERROR_MULTIBIT_ECC_FRAMER_RECEIVE_IQ_MEMORY	565
3.2.2.3.134	CPRI_ERROR_MULTIBIT_ECC_TRANSMIT_CONFIGURATION_MEMORY	565
3.2.2.3.135	CPRI_ERROR_MULTIBIT_ECC_RECEIVE_CONFIGURATION_MEMORY	565
3.2.2.4	Enumeration Type Documentation	565
3.2.2.4.1	cpri_loop_mode_t	565
3.2.2.4.2	cpri_sync_mode_t	565
3.2.2.4.3	ext_sync_act_t	565
3.2.2.4.4	shared_sync_mode_source_t	565
3.2.2.4.5	reset_ack_assert_length_t	566
3.2.2.4.6	reset_detection_length_t	566
3.2.2.4.7	cpri_protocol_t	566
3.2.2.4.8	cpri_multicast_mode_t	566
3.2.2.4.9	mapping_mode_t	566
3.2.2.4.10	sampling_width_select_t	566
3.2.2.4.11	iq_bus_transaction_size_t	566
3.2.2.4.12	vss_bus_transaction_size_t	566
3.2.2.4.13	oversampling_ratio_t	566
3.2.2.4.14	cpri_interrupt_type_t	567
3.2.2.5	Function Documentation	567
3.2.2.5.1	cpriInitialize	567
3.2.2.5.2	cpriIqInitialize	567
3.2.2.5.3	cpriVssInitialize	568
3.2.2.5.4	cpriEthernetInitialize	568
3.2.2.5.5	cpriHdlcInitialize	569
3.2.3	CPRI runtime	569
3.2.3.1	Overview	569
3.2.3.2	Data Structure Documentation	570
3.2.3.2.1	struct cpri_delays_t	570
3.2.3.2.1.1	Field Documentation	570
3.2.3.2.1.1.1	t_ext	570
3.2.3.2.1.1.2	t_14	570
3.2.3.2.1.1.3	t_offset	570

Section number	Title	Page
3.2.3.2.2	struct cpri_control_table_entry_t	570
3.2.3.2.2.1	Field Documentation	571
3.2.3.2.2.1.1	address	571
3.2.3.2.2.1.2	data_register_0	571
3.2.3.2.2.1.3	data_register_1	571
3.2.3.2.2.1.4	data_register_2	571
3.2.3.2.3	struct cpri_iq_open_params_t	571
3.2.3.2.3.1	Field Documentation	571
3.2.3.2.3.1.1	multicast_buffer_data_base	571
3.2.3.2.3.1.2	hfn_tx	571
3.2.3.2.3.1.3	hfn_rx	572
3.2.3.2.3.1.4	bfn_tx	572
3.2.3.2.3.1.5	bfn_rx	572
3.2.3.2.3.1.6	iq_tx_begin	572
3.2.3.2.3.1.7	iq_multicast_rx	572
3.2.3.2.3.1.8	error_callback	572
3.2.3.2.4	struct cpri_vss_open_params_t	572
3.2.3.2.4.1	Field Documentation	572
3.2.3.2.4.1.1	error_callback	572
3.2.3.2.5	struct cpri_ethernet_open_params_t	573
3.2.3.2.5.1	Field Documentation	573
3.2.3.2.5.1.1	error_callback	573
3.2.3.2.6	struct cpri_ethernet_channel_params_t	573
3.2.3.2.6.1	Field Documentation	573
3.2.3.2.6.1.1	bd_ring_steering_bits	573
3.2.3.2.6.1.2	buffer_steering_bits	573
3.2.3.2.6.1.3	bd_ring_attributes	573
3.2.3.2.6.1.4	buffer_attributes	573
3.2.3.2.7	struct cpri_hdlc_open_params_t	574
3.2.3.2.7.1	Field Documentation	574
3.2.3.2.7.1.1	error_callback	574
3.2.3.2.8	struct cpri_hdlc_channel_params_t	574
3.2.3.2.8.1	Field Documentation	574
3.2.3.2.8.1.1	bd_ring_steering_bits	574
3.2.3.2.8.1.2	buffer_steering_bits	574
3.2.3.2.8.1.3	bd_ring_attributes	574
3.2.3.2.8.1.4	buffer_attributes	574
3.2.3.2.9	struct cpri_iq_int_cb_param_t	575
3.2.3.2.9.1	Field Documentation	575
3.2.3.2.9.1.1	callback_param	575
3.2.3.2.9.1.2	cpri_num	575
3.2.3.2.9.1.3	event_type	575
3.2.3.2.10	struct cpri_channel_params_t	575
3.2.3.2.11	struct cpri_error_param_t	575
3.2.3.2.11.1	Field Documentation	576

Section number	Title	Page
3.2.3.2.11.1.1	cpri_num	576
3.2.3.2.11.1.2	error_events	576
3.2.3.2.11.1.3	error_ecc_multibit_events	576
3.2.3.2.12	struct cpri_reconfiguration_level0_param_t	576
3.2.3.2.12.1	Field Documentation	576
3.2.3.2.12.1.1	maximal_desired_link_rate	576
3.2.3.2.12.1.2	minimal_accepted_link_rate	576
3.2.3.2.12.1.3	cpri_init_params	576
3.2.3.3	Macro Definition Documentation	576
3.2.3.3.1	CPRI_ETHERNET_CMD_RX_POLL	576
3.2.3.3.2	CPRI_ETHERNET_CMD_TX_CONFIRM	577
3.2.3.3.3	CPRI_HDLC_CMD_RX_POLL	577
3.2.3.3.4	CPRI_HDLC_CMD_TX_CONFIRM	577
3.2.4	CPRI B4860 architecture-specific definitions	577
3.2.4.1	Overview	577
3.2.4.2	Data Structure Documentation	578
3.2.4.2.1	struct cpri_global_init_params_t	578
3.2.4.2.1.1	Field Documentation	578
3.2.4.2.1.1.1	initializing_core	578
3.2.4.2.1.1.2	cpri_num_of_used_units	578
3.2.4.2.1.1.3	maximal_desired_link_rate	578
3.2.4.2.1.1.4	minimal_accepted_link_rate	579
3.2.4.2.2	union cpri_buffer_attributes_t	579
3.2.4.2.2.1	Field Documentation	579
3.2.4.2.2.1.1	c	579
3.2.4.2.2.1.2	lru	579
3.2.4.2.2.1.3	w	579
3.2.4.2.2.1.4	m	579
3.2.4.2.2.1.5	g	579
3.2.4.2.2.1.6	l2pid	579
3.2.4.2.2.1.7	enhanced	579
3.2.4.2.2.1.8	liodn	579
3.2.4.3	Macro Definition Documentation	580
3.2.4.3.1	REGISTERS_TYPE	580
3.2.4.3.2	DEVICE_REGISTERS	580
3.2.4.3.3	CPRI_GENERAL_REGISTERS	580
3.2.4.3.4	CPRI_MAX_NUM_OF_CORES	580
3.2.4.3.5	CPRI_ICCR_CLOCKS	580
3.2.4.3.6	CPRI_NUM_OF_TER_RER_INTERRUPTS_PER_UNIT	580
3.2.4.3.7	CPRI_NUM_OF_TX_CONTROL_INTERRUPTS_PER_UNIT	580
3.2.4.3.8	CPRI_NUM_OF_RX_CONTROL_INTERRUPTS_PER_UNIT	580
3.2.4.3.9	CPRI_NUM_OF_TX_TIMING_INTERRUPTS_PER_UNIT	580
3.2.4.3.10	CPRI_NUM_OF_RX_TIMING_INTERRUPTS_PER_UNIT	581
3.2.4.3.11	CPRI_NUM_OF_ERROR_INTERRUPTS_PER_UNIT	581
3.2.4.3.12	CPRI_TOTAL_NUM_OF_TER_RER_INTERRUPTS	581

Section number	Title	Page
3.2.4.3.13	CPRI_TOTAL_NUM_OF_TX_CONTROL_INTERRUPTS	581
3.2.4.3.14	CPRI_TOTAL_NUM_OF_RX_CONTROL_INTERRUPTS	581
3.2.4.3.15	CPRI_TOTAL_NUM_OF_TX_TIMING_INTERRUPTS	581
3.2.4.3.16	CPRI_TOTAL_NUM_OF_RX_TIMING_INTERRUPTS	581
3.2.4.3.17	CPRI_TOTAL_NUM_OF_ERROR_INTERRUPTS	581
3.2.4.3.18	CPRI_NUMBER_OF_CLOCK_DOMAINS	581
3.2.4.3.19	CPRI_GET_CLOCK_DOMAIN_INDEX	582
3.2.4.4	Enumeration Type Documentation	582
3.2.4.4.1	cpri_group_t	582
3.2.4.4.2	cpri_link_rate_t	582
3.2.4.5	Function Documentation	582
3.2.4.5.1	cpriPairNumGet	582
3.2.4.5.2	cpriSerdesDelaysCalculate	582
3.2.4.5.3	cpriArchStatisticsSet	583
3.2.4.5.4	cpriPowerGating	583
3.3	**sRIO Module API**	**583**
3.3.1	Overview	583
3.3.2	sRIO initialization	583
3.3.2.1	Overview	583
3.3.2.2	Data Structure Documentation	584
3.3.2.2.1	struct srio_init_params_t	584
3.3.2.2.1.1	Field Documentation	584
3.3.2.2.1.1.1	num_ports	584
3.3.2.2.1.1.2	ocn_dma_to_use	584
3.3.2.2.2	struct srio_sys_init_param_t	584
3.3.2.2.2.1	Field Documentation	585
3.3.2.2.2.1.1	port_num	585
3.3.2.2.2.1.2	srio_connectivity	585
3.3.2.2.2.1.3	enumerate	585
3.3.2.3	Macro Definition Documentation	585
3.3.2.3.1	SRIO_DEFAULT_HOST_ID	585
3.3.2.3.2	SRIO_ACCEPT_ALL	585
3.3.2.4	Function Documentation	585
3.3.2.4.1	initSrioSystem	585
3.3.2.4.2	srioInitialize	586
3.3.2.4.3	srioDevicesInitialize	586
3.3.2.4.4	srioDeviceAdd	586
3.3.2.4.5	srioRecover	587
3.3.2.4.6	srioAlternateIdSet	587
3.3.2.4.7	srioAlternateIdDisable	587
3.3.2.4.8	srioAcceptAllConfigure	588
3.3.3	sRIO runtime	588
3.3.3.1	Overview	588
3.3.3.2	Data Structure Documentation	590

Section number	Title	Page
3.3.3.2.1	struct srio_device_t	590
3.3.3.2.1.1	Field Documentation	590
3.3.3.2.1.1.1	dev_id	590
3.3.3.2.1.1.2	hop_count	590
3.3.3.2.1.1.3	switch_port_num	591
3.3.3.2.1.1.4	srio_port_num	591
3.3.3.2.2	struct srio_in_win_config_t	591
3.3.3.2.2.1	Field Documentation	591
3.3.3.2.2.1.1	win_num	591
3.3.3.2.2.1.2	base	591
3.3.3.2.2.1.3	translation	591
3.3.3.2.2.1.4	snoop_enable	591
3.3.3.2.2.1.5	win_size	591
3.3.3.2.2.1.6	target_interface	591
3.3.3.2.3	struct srio_out_win_config_t	592
3.3.3.2.3.1	Field Documentation	592
3.3.3.2.3.1.1	win_num	592
3.3.3.2.3.1.2	srio_id	592
3.3.3.2.3.1.3	base	592
3.3.3.2.3.1.4	translation	592
3.3.3.2.3.1.5	win_size	592
3.3.3.2.3.1.6	srio_flow_level	592
3.3.3.2.3.1.7	srio_write_access_type	592
3.3.3.2.4	struct srio_maint_win_config_t	593
3.3.3.2.4.1	Field Documentation	593
3.3.3.2.4.1.1	win_num	593
3.3.3.2.4.1.2	base	593
3.3.3.2.4.1.3	win_size	593
3.3.3.2.4.1.4	srio_id	593
3.3.3.2.4.1.5	hop_count	593
3.3.3.2.4.1.6	offset	593
3.3.3.3	Macro Definition Documentation	593
3.3.3.3.1	SRIO_ATMU_DEFAULT_WINDOW	593
3.3.3.3.2	SRIO_LCFBA_ADDRESS	593
3.3.3.3.3	SRIO_LCF_ATMU_OUT_WIN_SIZE	594
3.3.3.3.4	srioGetID	594
3.3.3.3.5	srioGetAnyID	594
3.3.3.3.6	srioIsSmallSystem	594
3.3.3.4	Typedef Documentation	594
3.3.3.4.1	srio_dev_id_t	594
3.3.3.5	Enumeration Type Documentation	594
3.3.3.5.1	srio_flow_level_t	594
3.3.3.5.2	srio_port_id_t	595
3.3.3.5.3	srio_connectivity_t	595
3.3.3.5.4	srio_write_type_t	595

Section number	Title	Page
3.3.3.5.5	srio_in_target_interface_t	595
3.3.3.5.6	srio_atmu_in_win_sz_t	595
3.3.3.5.7	srio_atmu_out_win_sz_t	596
3.3.3.5.8	srio_traffic_management_t	597
3.3.3.6	Function Documentation	597
3.3.3.6.1	srioOutboundWindowFree	597
3.3.3.6.2	srioOutboundWindowFind	597
3.3.3.6.3	srioOutboundWindowEnable	598
3.3.3.6.4	srioOutboundWindowDisable	598
3.3.3.6.5	srioOutboundWindowOpen	598
3.3.3.6.6	srioInboundWindowFind	599
3.3.3.6.7	srioInboundWindowFree	599
3.3.3.6.8	srioInboundWindowEnable	599
3.3.3.6.9	srioInboundWindowDisable	599
3.3.3.6.10	srioInboundWindowOpen	600
3.3.3.6.11	srioMaintenanceAccess	600
3.3.3.6.12	srioMaintenanceTargetSet	601
3.3.3.6.13	srioMaintenanceAtmuFree	601
3.3.3.6.14	srioMaintenanceAtmuOpen	601
3.3.3.6.15	srioTrafficManagementSupport	602
3.3.3.6.16	srioDeviceIdGetByAttr	602
3.3.3.6.17	srioDeviceGetByIndex	602
3.3.3.6.18	srioDeviceGetById	603
3.3.3.6.19	srioNumDiscoverdDevicesGet	603
3.3.3.6.20	srioClearPortErrors	603
3.4	**I2C Module API**	**604**
3.4.1	Overview	604
3.4.2	Macro Definition Documentation	604
3.4.2.1	RETRY_DEF	604
3.4.2.2	I2C_TIMEOUT	604
3.4.2.3	I2C_CMD_SCAN_BUS	604
3.4.2.4	I2C_CMD_STOP_DEVICE	604
3.4.2.5	I2C_CMD_DEVICE_BUSY	605
3.4.2.6	I2C_CMD_FREE_STUCK_BUS	605
3.4.2.7	I2C_CMD_SET_FREQUENCY_DEVIDER	605
3.4.2.8	I2C_CMD_SET_DIGITAL_FILTER_SAMPLING_RATE	605
3.4.2.9	I2C_CMD_SET_TIMEOUT	605
3.4.2.10	I2C_FREQ_DEF	605
3.4.2.11	I2C_DFSR_DEF	605
3.4.3	I2C Initialization	605
3.4.3.1	Overview	605
3.4.3.2	Data Structure Documentation	606
3.4.3.2.1	struct i2c_init_params_t	606
3.4.3.2.1.1	Field Documentation	606

Section number	Title	Page
3.4.3.2.1.1.1	i2c_id	606
3.4.3.2.1.1.2	retry	606
3.4.3.2.1.1.3	frequency	606
3.4.3.2.1.1.4	dfsr	606
3.4.3.2.1.1.5	owner_core	606
3.4.3.2.2	struct i2c_open_params_t	606
3.4.3.2.2.1	Field Documentation	607
3.4.3.2.2.1.1	error_cb	607
3.4.3.3	Function Documentation	607
3.4.3.3.1	i2cInitialize	607
3.4.4	runtime API	607
3.4.4.1	Overview	607
3.4.4.2	Data Structure Documentation	608
3.4.4.2.1	struct i2c_scan_cmd_params_t	608
3.4.4.2.1.1	Field Documentation	608
3.4.4.2.1.1.1	addresses	608
3.4.4.2.1.1.2	print_func	608
3.4.4.2.1.1.3	start_address	608
3.4.4.2.1.1.4	end_address	608
3.4.4.3	Enumeration Type Documentation	608
3.4.4.3.1	i2c_trans_mode	608
3.4.4.3.2	i2c_next_trans_action	609
3.4.5	I2C B4860-specific definitions	609
3.4.5.1	Overview	609
3.4.5.2	Macro Definition Documentation	609
3.4.5.2.1	I2C_NUM_OF_DEVICES	609
3.4.5.2.2	I2C_1_DEVICE_NAME	609
3.4.5.2.3	I2C_2_DEVICE_NAME	609
3.4.5.2.4	I2C_3_DEVICE_NAME	609
3.4.5.2.5	I2C_4_DEVICE_NAME	610
3.4.5.3	Enumeration Type Documentation	610
3.4.5.3.1	i2c_id_t	610
3.5	**MAPLE-B3 Module API**	**610**
3.5.1	Overview	610
3.5.2	Maple API	610
3.5.2.1	Overview	610
3.5.2.2	Macro Definition Documentation	611
3.5.2.2.1	MAPLE_MMU_REGA_SHIFT	611
3.5.2.2.2	MAPLE_MMU_REGC_SHIFT	611
3.5.2.2.3	MAPLE_MMU_REGM_SHIFT	611
3.5.2.2.4	MAPLE_MMU_ATTR_FLEX_SEGMENT	612
3.5.2.2.5	MAPLE_MMU_ATTR_WPERM	612
3.5.2.2.6	MAPLE_MMU_ATTR_RPERM	612
3.5.2.2.7	MAPLE_MMU_ATTR_ENABLE	612

Section number	Title	Page
3.5.2.2.8	MAPLE_MMU_ATTR_DID	612
3.5.2.2.9	MAPLE_MMU_CHB_ATTR_ETYPE	612
3.5.2.2.10	MAPLE_MMU_CHB_ATTR_ENHANCED	612
3.5.2.2.11	MAPLE_MMU_CHB_ATTR_LIODN	612
3.5.2.2.12	MAPLE_MMU_ATTR_CACHEABLE	613
3.5.2.2.13	MAPLE_MMU_ATTR_GAURDED	613
3.5.2.2.14	MAPLE_MMU_ATTR_COHERENT	613
3.5.2.2.15	MAPLE_MMU_ATTR_WRITETHROUGH	613
3.5.2.2.16	MAPLE_MMU_ATTR_LEAST_RU	613
3.5.2.2.17	MAPLE_MMU_ATTR_L2_PARTITION_ID	613
3.5.2.3	Maple Initialization	613
3.5.2.3.1	Overview	613
3.5.2.3.2	Data Structure Documentation	617
3.5.2.3.2.1	struct maple_depe_wa_params_t	617
3.5.2.3.2.1.1	Field Documentation	617
3.5.2.3.2.1.2	bd	617
3.5.2.3.2.2	struct maple_malloc_t	617
3.5.2.3.2.2.1	Field Documentation	617
3.5.2.3.2.2.2	size	617
3.5.2.3.2.2.3	addr	617
3.5.2.3.2.2.4	alignment	617
3.5.2.3.2.3	struct maple_mmu_seg_t	618
3.5.2.3.2.3.1	Field Documentation	618
3.5.2.3.2.3.2	attr	618
3.5.2.3.2.3.3	virt_addr	618
3.5.2.3.2.3.4	phys_addr	618
3.5.2.3.2.3.5	size	618
3.5.2.3.2.4	struct maple_mmu_seg_update_t	618
3.5.2.3.2.4.1	Field Documentation	618
3.5.2.3.2.4.2	descriptor	618
3.5.2.3.2.4.3	attributes	618
3.5.2.3.2.5	struct maple_mmu_err_t	618
3.5.2.3.2.5.1	Field Documentation	619
3.5.2.3.2.5.2	err_addr	619
3.5.2.3.2.5.3	err_type	619
3.5.2.3.2.5.4	descritptors	619
3.5.2.3.2.5.5	err_event	619
3.5.2.3.2.6	struct maple_mmu_init_param_t	619
3.5.2.3.2.6.1	Field Documentation	619
3.5.2.3.2.6.2	trash_vaddr	619
3.5.2.3.2.6.3	steering_bits	619
3.5.2.3.2.7	union maple_version_t	620
3.5.2.3.2.7.1	Field Documentation	620
3.5.2.3.2.7.2	maple_w	620
3.5.2.3.2.7.3	dev_fw	620

Section number	Title	Page
3.5.2.3.2.7.4	ucode_vesion	620
3.5.2.3.2.8	struct maple_open_params_t	620
3.5.2.3.2.8.1	Field Documentation	620
3.5.2.3.2.8.2	error_callback	620
3.5.2.3.2.8.3	config_mbus_mmu	620
3.5.2.3.2.8.4	mbus_base	620
3.5.2.3.2.8.5	config_sbus_mmu	621
3.5.2.3.2.8.6	sbus_base	621
3.5.2.3.2.8.7	dev_init_params	621
3.5.2.3.2.8.8	maple_init	621
3.5.2.3.2.9	struct maple_init_params_t	621
3.5.2.3.2.9.1	Field Documentation	621
3.5.2.3.2.9.2	num_maples	621
3.5.2.3.3	Macro Definition Documentation	621
3.5.2.3.3.1	MAPLE_B3LW_UCODE	621
3.5.2.3.3.2	MAPLE_B3W_UCODE	621
3.5.2.3.3.3	MAPLE_B3W_INIT	621
3.5.2.3.3.4	MAPLE_B3LW_INIT	622
3.5.2.3.3.5	MAPLE_SYS_ERR	622
3.5.2.3.3.6	MAPLE_MMU_ERR	622
3.5.2.3.3.7	MAPLE_ECC_ERR	622
3.5.2.3.3.8	MAPLE_CMD_MALLOC	622
3.5.2.3.3.9	MAPLE_CMD_GET_UCODE_VERSION	622
3.5.2.3.3.10	MAPLE_CMD_PCR_ACTIVATE_WITH_POLL	622
3.5.2.3.3.11	MAPLE_CMD_PCR_ACTIVATE_NO_POLL	622
3.5.2.3.3.12	MAPLE_CMD_MMU_SEGMENT_FIND	622
3.5.2.3.3.13	MAPLE_CMD_MMU_SEGMENT_UPDATE	622
3.5.2.3.3.14	MAPLE_CMD_MMU_SEGMENT_ENABLE	623
3.5.2.3.3.15	MAPLE_CMD_MMU_SEGMENT_DISABLE	623
3.5.2.3.3.16	MAPLE_CMD_MMU_ENABLE	623
3.5.2.3.3.17	MAPLE_CMD_MMU_DISABLE	623
3.5.2.3.3.18	MAPLE_CMD_MMU_PROTECT_DISABLE	623
3.5.2.3.3.19	MAPLE_CMD_MMU_PROTECT_ENABLE	623
3.5.2.3.3.20	MAPLE_CMD_MMU_ERROR_DETECT	623
3.5.2.3.3.21	MAPLE_CMD_SOFT_RESET	623
3.5.2.3.3.22	MAPLE_CMD_SET_TRACE_EVENTS	623
3.5.2.3.3.23	MAPLE_CMD_CLEAR_TRACE_BUFFER	623
3.5.2.3.3.24	MAPLE_CMD_GET_TRACE_BUFFER	624
3.5.2.3.3.25	MAPLE_CMD_GET_TRACE_BUFFER_WRITE_PTR	624
3.5.2.3.3.26	MAPLE_CMD_ACTIVATE_PDSCH_ACK_INT	624
3.5.2.3.3.27	MAPLE_CMD_DISABLE_ECC_PROTECTION	624
3.5.2.3.3.28	MAPLE_CMD_ENABLE_ECC_PROTECTION	624
3.5.2.3.3.29	MAPLE_CMD_DEPE_WA	624
3.5.2.3.3.30	MAPLE_CMD_PRE_POST_TASK_ID	624
3.5.2.3.3.31	MAPLE_FLG_DEFAULT	624

Section number	Title	Page
3.5.2.3.3.32	MAPLE_0_NAME	624
3.5.2.3.3.33	MAPLE_1_NAME	624
3.5.2.3.3.34	MAPLE_2_NAME	624
3.5.2.3.4	Enumeration Type Documentation	625
3.5.2.3.4.1	anonymous enum	625
3.5.2.3.4.2	maple_pspicer2_flag_t	625
3.5.2.3.4.3	maple_pcr_opcodes_t	626
3.5.2.3.4.4	maple_dev_id_t	626
3.5.2.3.4.5	anonymous enum	627
3.5.2.3.4.6	anonymous enum	627
3.5.2.3.4.7	maple_b3lw_init_power_sch_pe_t	627
3.5.2.3.4.8	maple_b3lw_init_power_sch_t	628
3.5.2.3.4.9	maple_b3lw_init_axi_aloc_bus_type_t	628
3.5.2.3.4.10	maple_b3lw_init_axi_aloc_pe_t	628
3.5.2.3.4.11	maple_b3lw_init_axi_aloc_t	628
3.5.2.3.5	Function Documentation	629
3.5.2.3.5.1	mapleDirectRead	629
3.5.2.3.5.2	mapleDirectWrite	629
3.5.2.3.5.3	mapleInitialize	630
3.5.2.3.5.4	MAPLE_B3LW_INIT_PEx_POWER_SCHEME	630
3.5.2.3.5.5	MAPLE_B3LW_INIT_AXIx_DEFAULT_PRIORITY	631
3.5.2.3.5.6	MAPLE_B3LW_INIT_AXI_PE_ALC_PARAM	631
3.5.2.3.6	Variable Documentation	631
3.5.2.3.6.1	maple3lw_code	631
3.5.2.3.6.2	maple3w_code	632
3.5.2.4	Maple Workarounds API	632
3.5.2.4.1	Overview	632
3.5.2.4.2	Function Documentation	632
3.5.2.4.2.1	maplePgCg	632
3.5.3	Maple PE Generic API	632
3.5.3.1	Overview	632
3.5.3.2	Maple xxPE Runtime	633
3.5.3.2.1	Overview	633
3.5.3.2.2	Data Structure Documentation	633
3.5.3.2.2.1	struct maple_xxpe_ch_info_t	633
3.5.3.2.2.1.1	Field Documentation	633
3.5.3.2.2.1.2	bd_base	633
3.5.3.2.2.1.3	bd_enq	634
3.5.3.2.2.1.4	bd_deq	634
3.5.3.2.2.1.5	bd_ring_size	634
3.5.3.2.2.1.6	num_dispatched_bd	634
3.5.3.2.2.1.7	num_reaped_bd	634
3.5.3.2.3	Macro Definition Documentation	634
3.5.3.2.3.1	MAPLE_PE_CH_CMD_RX_POLL	634
3.5.3.2.3.2	MAPLE_PE_CH_CMD_VIRT_TRANS_ENABLE	634

Section number	Title	Page
3.5.3.2.3.3	MAPLE_PE_CH_CMD_VIRT_TRANS_DISABLE	634
3.5.3.2.3.4	MAPLE_PE_CH_CMD_GO	634
3.5.3.2.3.5	MAPLE_PE_CH_INFO_GET	634
3.5.3.2.3.6	MAPLE_PE_CH_ACTIVATE_JOB	634
3.5.3.3	Maple xxPE Initialization	635
3.5.3.3.1	Overview	635
3.5.3.3.2	Data Structure Documentation	636
3.5.3.3.2.1	struct maple_pe_init_params_t	636
3.5.3.3.2.1.1	Field Documentation	636
3.5.3.3.2.1.2	num_devices	636
3.5.3.3.2.2	struct maple_pe_ch_open_params_t	636
3.5.3.3.2.2.1	Field Documentation	636
3.5.3.3.2.2.2	flags	636
3.5.3.3.2.2.3	high_priority	636
3.5.3.3.2.2.4	int_enable	636
3.5.3.3.2.2.5	no_translation	637
3.5.3.3.2.2.6	no_automatic_reap	637
3.5.3.3.2.2.7	single_channel_to_int	637
3.5.3.3.2.2.8	use_go_function	637
3.5.3.3.2.2.9	channel_location	637
3.5.3.3.2.2.10	int_dispatcher	637
3.5.3.3.2.2.11	int_num	637
3.5.3.3.2.2.12	int_priority	637
3.5.3.3.2.2.13	mmu_task_id	637
3.5.3.3.2.2.14	steering_bits	637
3.5.3.3.3	Macro Definition Documentation	637
3.5.3.3.3.1	MAPLE_MAX_NUM_BD_PTRS	637
3.5.3.3.4	Typedef Documentation	638
3.5.3.3.4.1	maple_pe_ch	638
3.5.3.3.4.2	maple_pe_hw_sem_id	638
3.5.3.3.4.3	maple_xxpe_dispatch	638
3.5.3.3.4.4	maple_xxpe_reap	638
3.5.3.3.5	Enumeration Type Documentation	638
3.5.3.3.5.1	maple_pe_bd_priority_t	638
3.5.3.3.5.2	maple_pe_num_bd_t	638
3.5.3.3.5.3	anonymous enum	639
3.5.3.3.5.4	maple_pe_type_t	639
3.5.3.3.5.5	maple_pe_int_t	639
3.5.3.3.5.6	maple_pe_int_trgt_t	639
3.5.3.3.5.7	anonymous enum	639
3.5.4	Maple CRCPE API	640
3.5.4.1	Overview	640
3.5.4.2	CRCPE Initialization	640
3.5.4.2.1	Overview	640
3.5.4.2.2	Data Structure Documentation	641

Section number	Title	Page
3.5.4.2.2.1	struct crcpe_open_params_t	641
3.5.4.2.2.1.1	Field Documentation	641
3.5.4.2.2.1.2	maple_handle	641
3.5.4.2.2.1.3	maple_pe_bd_priority	641
3.5.4.2.2.1.4	maple_pe_num_bd	641
3.5.4.2.2.1.5	crcpe_params	641
3.5.4.2.3	Macro Definition Documentation	641
3.5.4.2.3.1	CRCPE_DEV_ID_0	641
3.5.4.2.3.2	MAPLE_0_CRCPE_NAME	641
3.5.4.2.3.3	MAPLE_1_CRCPE_NAME	642
3.5.4.2.3.4	MAPLE_2_CRCPE_NAME	642
3.5.4.2.4	Typedef Documentation	642
3.5.4.2.4.1	maple_crcpe_init_params_t	642
3.5.4.2.4.2	maple_crcpe_init_params_s	642
3.5.4.2.4.3	maple_crcpe_ch_open_params_t	642
3.5.4.2.5	Enumeration Type Documentation	642
3.5.4.2.5.1	anonymous enum	642
3.5.4.2.6	Function Documentation	642
3.5.4.2.6.1	mapleCrcpeInitialize	642
3.5.4.3	CRCPE Runtime	643
3.5.4.3.1	Overview	643
3.5.4.3.2	Data Structure Documentation	644
3.5.4.3.2.1	struct maple_crcpe_job_t	644
3.5.4.3.2.1.1	Field Documentation	644
3.5.4.3.2.1.2	flags	644
3.5.4.3.2.1.3	buffer_size	644
3.5.4.3.2.1.4	input	645
3.5.4.3.2.1.5	init_value	645
3.5.4.3.2.1.6	result	645
3.5.4.3.2.1.7	aux	645
3.5.4.3.2.1.8	bd_index	645
3.5.4.3.3	Macro Definition Documentation	645
3.5.4.3.3.1	CRCPE_MAX_NUM_BD_FOR_DISPACTH	645
3.5.4.3.3.2	MAPLE_CRCPE_CMD_RX_POLL	645
3.5.4.3.3.3	CRCPE_BD_MANUAL_ACTIVATION	646
3.5.4.3.3.4	CRCPE_INT_EN	646
3.5.4.3.3.5	CRCPE_UPDATE	646
3.5.4.3.3.6	CRCPE_RVRS_IN	646
3.5.4.3.3.7	CRCPE_RVRS_OUT	646
3.5.4.3.3.8	CRCPE_INV_OUT	646
3.5.4.3.3.9	CRCPE_CHECK_FAIL	646
3.5.4.3.4	Enumeration Type Documentation	646
3.5.4.3.4.1	anonymous enum	646
3.5.4.3.4.2	anonymous enum	646
3.5.4.3.4.3	anonymous enum	647

Section number	Title	Page
3.5.4.3.5	Function Documentation	647
3.5.4.3.5.1	CRCPE_BS	647
3.5.5	Maple TVPE API	647
3.5.5.1	Overview	647
3.5.5.2	Macro Definition Documentation	648
3.5.5.2.1	TVPE_MAX_NUM_BD_FOR_DISPACTH	648
3.5.5.2.2	NUM_VITERBI_POLYNOM_SETS	648
3.5.5.2.3	NUM_VITERBI_PUNC_VECTORS	648
3.5.5.3	TVPE Initialization	648
3.5.5.3.1	Overview	648
3.5.5.3.2	Data Structure Documentation	649
3.5.5.3.2.1	struct tvpe_viterbi_polynomial_set	649
3.5.5.3.2.1.1	Field Documentation	649
3.5.5.3.2.1.2	m_polygen1	649
3.5.5.3.2.1.3	m_polygen0	649
3.5.5.3.2.1.4	m_polygen3	650
3.5.5.3.2.1.5	m_polygen2	650
3.5.5.3.2.2	struct tvpe_turbo_params_t	650
3.5.5.3.2.2.1	Field Documentation	650
3.5.5.3.2.2.2	apq_threshold	650
3.5.5.3.2.3	struct tvpe_viterbi_params_t	650
3.5.5.3.2.3.1	Field Documentation	650
3.5.5.3.2.3.2	punturing_vector	650
3.5.5.3.2.3.3	puncturing_pattern	650
3.5.5.3.2.3.4	polynomial_set	651
3.5.5.3.2.4	struct tvpe_open_params_t	651
3.5.5.3.2.4.1	Field Documentation	651
3.5.5.3.2.4.2	maple_handle	651
3.5.5.3.2.4.3	maple_pe_bd_priority	651
3.5.5.3.2.4.4	maple_pe_num_bd	651
3.5.5.3.2.4.5	tvpe_turbo_params	651
3.5.5.3.2.4.6	tvpe_viterbi_params	651
3.5.5.3.3	Macro Definition Documentation	651
3.5.5.3.3.1	TVPE_DEV_ID_0	651
3.5.5.3.3.2	TVPE_DEV_ID_1	652
3.5.5.3.3.3	MAPLE_0_TVPE_NAME	652
3.5.5.3.3.4	MAPLE_1_TVPE_NAME	652
3.5.5.3.4	Typedef Documentation	652
3.5.5.3.4.1	maple_tvpe_init_params_t	652
3.5.5.3.4.2	maple_tvpe_init_params_s	652
3.5.5.3.4.3	maple_tvpe_ch_open_params_t	652
3.5.5.3.5	Enumeration Type Documentation	652
3.5.5.3.5.1	anonymous enum	652
3.5.5.3.6	Function Documentation	652
3.5.5.3.6.1	mapleTvpeInitialize	652

Section number	Title	Page
3.5.5.4	TVPE Runtime	653
3.5.5.4.1	Overview	653
3.5.5.4.2	Data Structure Documentation	655
3.5.5.4.2.1	struct tvpe_job_e_param_t	655
3.5.5.4.2.1.1	Field Documentation	656
3.5.5.4.2.1.2	e_cb_sd_ini	656
3.5.5.4.2.1.3	e_sd_minus	656
3.5.5.4.2.1.4	e_sd_plus	656
3.5.5.4.2.1.5	e_cb_pfa_ini	656
3.5.5.4.2.1.6	e_pfa_minus	656
3.5.5.4.2.1.7	e_pfa_plus	656
3.5.5.4.2.1.8	e_cb_psa_ini	656
3.5.5.4.2.1.9	e_psa_minus	656
3.5.5.4.2.1.10	e_psa_plus	656
3.5.5.4.2.2	struct maple_tvpe_job_t	656
3.5.5.4.2.2.1	Field Documentation	657
3.5.5.4.2.2.2	first_flags	657
3.5.5.4.2.2.3	second_flags	657
3.5.5.4.2.2.4	third_flags	657
3.5.5.4.2.2.5	hard_output_addr	657
3.5.5.4.2.2.6	inputs	657
3.5.5.4.2.2.7	offsets	658
3.5.5.4.2.2.8	harq_buffer	658
3.5.5.4.2.2.9	harq_weight_flags	658
3.5.5.4.2.2.10	soft_output_addr	658
3.5.5.4.2.2.11	e_param_ptr	658
3.5.5.4.2.2.12	harq_acc_input_addr	658
3.5.5.4.2.2.13	pfs_vec_sizes	658
3.5.5.4.2.2.14	pf_vector_size	659
3.5.5.4.2.2.15	ps_vector_size	659
3.5.5.4.2.2.16	harq_acc_output_addr	659
3.5.5.4.2.2.17	fourth_flags	659
3.5.5.4.2.2.18	bd_index	659
3.5.5.4.3	Macro Definition Documentation	659
3.5.5.4.3.1	MAPLE_TVPE_CMD_SET_VITERBI_POLY	659
3.5.5.4.3.2	MAPLE_TVPE_CMD_GET_PE_ACCUMULATOR	659
3.5.5.4.3.3	MAPLE_TVPE_CMD_RX_POLL	659
3.5.5.4.3.4	TVPE_BD_MANUAL_ACTIVATION	659
3.5.5.4.3.5	TVPE_INT_EN	660
3.5.5.4.3.6	TVPE_TURBO_LMAP_EN	660
3.5.5.4.3.7	TVPE_3GLTE_EN	660
3.5.5.4.3.8	TVPE_VITERBI_TBZE	660
3.5.5.4.3.9	TVPE_ZTTB	660
3.5.5.4.3.10	TVPE_TURBO_CRC_EN	660
3.5.5.4.3.11	TVPE_TURBO_HAOE	660

Section number	Title	Page
3.5.5.4.3.12	TVPE_BLOCK_SIZE	660
3.5.5.4.3.13	TVPE_TURBO_LLMAP_LCF	660
3.5.5.4.3.14	TVPE_HOE	660
3.5.5.4.3.15	TVPE_TURBO_DPNC_EN	660
3.5.5.4.3.16	TVPE_BUF_SIZE	661
3.5.5.4.3.17	TVPE_TURBO_VEC_OFFSET	661
3.5.5.4.3.18	TVPE_HARD_OUT_OFFSET	661
3.5.5.4.3.19	TVPE_TURBO_LLR_OUT_SF	661
3.5.5.4.3.20	TVPE_HARQ_IHBSA	661
3.5.5.4.3.21	TVPE_FIRST_TIME_HARQ	661
3.5.5.4.3.22	TVPE_HARQ_IHBSZ	661
3.5.5.4.3.23	TVPE_TURBO_HARQ_W1_EN	661
3.5.5.4.3.24	TVPE_TURBO_HARQ_W2_EN	661
3.5.5.4.3.25	TVPE_TURBO_HARQ_W3_EN	661
3.5.5.4.3.26	TVPE_TURBO_HARQ_W1	661
3.5.5.4.3.27	TVPE_TURBO_HARQ_W2	662
3.5.5.4.3.28	TVPE_TURBO_HARQ_W3	662
3.5.5.4.3.29	TVPE_CRC_REFLECT	662
3.5.5.4.3.30	TVPE_AQC_AUTOSTOP	662
3.5.5.4.3.31	TVPE_CRC_AUTOSTOP	662
3.5.5.4.3.32	TVPE_CRC_SCRC	662
3.5.5.4.3.33	TVPE_HARQ_EN	662
3.5.5.4.3.34	TVPE_POLARITY	662
3.5.5.4.3.35	TVPE_DOBSY	662
3.5.5.4.3.36	TVPE_DOBSI	662
3.5.5.4.4	Enumeration Type Documentation	663
3.5.5.4.4.1	maple_tvpe_k_t	663
3.5.5.4.4.2	maple_tvpe_rate_t	663
3.5.5.4.4.3	maple_tvpe_type_t	663
3.5.5.4.4.4	maple_tvpe_punc_t	663
3.5.5.4.4.5	maple_tvpe_max_iter_t	663
3.5.5.4.4.6	maple_tvpe_min_iter_t	663
3.5.5.4.4.7	maple_tvpe_data_struct_t	663
3.5.5.4.4.8	maple_tvpe_crc_t	664
3.5.5.4.4.9	maple_tvpe_dre_t	664
3.5.5.4.4.10	maple_tvpe_mbus_prio_t	664
3.5.5.4.4.11	maple_tvpe_soft_output_t	664
3.5.5.4.4.12	maple_tvpe_vit_set_t	665
3.5.5.4.4.13	tvpe_harq_hus_t	665
3.5.6	Maple FTPE API	665
3.5.6.1	Overview	665
3.5.6.2	FTPE Initialization	665
3.5.6.2.1	Overview	665
3.5.6.2.2	Data Structure Documentation	667
3.5.6.2.2.1	struct ftpe_size_params_t	667

Section number	Title	Page
3.5.6.2.2.1.1	Field Documentation	667
3.5.6.2.2.1.2	ds0	667
3.5.6.2.2.1.3	ds1	667
3.5.6.2.2.1.4	ds2	667
3.5.6.2.2.1.5	ds3	667
3.5.6.2.2.1.6	ds4	667
3.5.6.2.2.1.7	ds5	667
3.5.6.2.2.2	struct ftpe_open_params_t	667
3.5.6.2.2.2.1	Field Documentation	667
3.5.6.2.2.2.2	maple_handle	667
3.5.6.2.2.2.3	maple_pe_bd_priority	668
3.5.6.2.2.2.4	maple_pe_num_bd	668
3.5.6.2.2.2.5	ftpe_config	668
3.5.6.2.2.2.6	ftpe_data_size_sets	668
3.5.6.2.3	Macro Definition Documentation	668
3.5.6.2.3.1	FTPE_MAX_NUM_BD_FOR_DISPACTH	668
3.5.6.2.3.2	FTPE_PARAM_DSS_NUM	668
3.5.6.2.3.3	MAPLE_FTPE_BCIS_EN	668
3.5.6.2.3.4	MAPLE_0_FTPE_0_NAME	668
3.5.6.2.3.5	MAPLE_0_FTPE_1_NAME	668
3.5.6.2.3.6	MAPLE_0_FTPE_2_NAME	668
3.5.6.2.3.7	MAPLE_1_FTPE_0_NAME	668
3.5.6.2.3.8	MAPLE_1_FTPE_1_NAME	669
3.5.6.2.3.9	MAPLE_1_FTPE_2_NAME	669
3.5.6.2.3.10	MAPLE_2_FTPE_0_NAME	669
3.5.6.2.3.11	MAPLE_2_FTPE_1_NAME	669
3.5.6.2.4	Typedef Documentation	669
3.5.6.2.4.1	maple_ftpe_init_params_t	669
3.5.6.2.4.2	maple_ftpe_init_params_s	669
3.5.6.2.4.3	maple_ftpe_ch_open_params_t	669
3.5.6.2.5	Enumeration Type Documentation	669
3.5.6.2.5.1	maple_ftpe_buff_config_t	669
3.5.6.2.5.2	anonymous enum	669
3.5.6.2.5.3	anonymous enum	670
3.5.6.2.6	Function Documentation	670
3.5.6.2.6.1	mapleFtpeInitialize	670
3.5.6.3	FTPE Runtime	670
3.5.6.3.1	Overview	670
3.5.6.3.2	Data Structure Documentation	672
3.5.6.3.2.1	struct maple_ftpe_mult_buff_params_t	672
3.5.6.3.2.1.1	Field Documentation	673
3.5.6.3.2.1.2	pre_buff_ptr	673
3.5.6.3.2.1.3	pst_buff_ptr	673
3.5.6.3.2.1.4	pre_buff_size	673
3.5.6.3.2.1.5	pst_buff_size	673

Section number	Title	Page
3.5.6.3.2.1.6	pst_steering_bits	673
3.5.6.3.2.1.7	pre_steering_bits	673
3.5.6.3.2.2	union maple_ftpe_job_extension_t	673
3.5.6.3.2.2.1	Field Documentation	674
3.5.6.3.2.2.2	extension_header	674
3.5.6.3.2.2.3	fcg	674
3.5.6.3.2.2.4	cg	674
3.5.6.3.2.2.5	fc_rpt	674
3.5.6.3.2.2.6	fc_pre	674
3.5.6.3.2.2.7	fc_in_sc	674
3.5.6.3.2.2.8	fcb_ptr	674
3.5.6.3.2.2.9	fc_base_real	674
3.5.6.3.2.2.10	fc_base_img	674
3.5.6.3.2.2.11	fc_size	674
3.5.6.3.2.2.12	fc_step_size	675
3.5.6.3.2.2.13	fc_shift_real	675
3.5.6.3.2.2.14	next_num_pi	675
3.5.6.3.2.2.15	chip_offset	675
3.5.6.3.2.2.16	curr_num_pi	675
3.5.6.3.2.2.17	slot_gen_off	675
3.5.6.3.2.2.18	lss	675
3.5.6.3.2.2.19	ssn	675
3.5.6.3.2.2.20	code_length	675
3.5.6.3.2.2.21	slot_num	675
3.5.6.3.2.2.22	preas_scl_ptr	675
3.5.6.3.2.2.23	fc_shift_img	675
3.5.6.3.2.3	struct maple_ftpe_job_t	675
3.5.6.3.2.3.1	Field Documentation	676
3.5.6.3.2.3.2	first_flags	676
3.5.6.3.2.3.3	second_flags	676
3.5.6.3.2.3.4	third_flags	676
3.5.6.3.2.3.5	fourth_flags	677
3.5.6.3.2.3.6	fifth_flags	677
3.5.6.3.2.3.7	buffer_size	677
3.5.6.3.2.3.8	post_multiplication	677
3.5.6.3.2.3.9	post_mult_real	677
3.5.6.3.2.3.10	post_mult_img	677
3.5.6.3.2.3.11	input	677
3.5.6.3.2.3.12	output	677
3.5.6.3.2.3.13	bd_rpt_stat_ptr	677
3.5.6.3.2.3.14	pre_operation_ptr	677
3.5.6.3.2.3.15	post_multiplier_ptr	678
3.5.6.3.2.3.16	in_exp_ptr	678
3.5.6.3.2.3.17	input_exponent	678
3.5.6.3.2.3.18	per_resb_exp_ptr	678

Section number	Title	Page
3.5.6.3.2.3.19	extension_ptr	678
3.5.6.3.2.3.20	bd_index	678
3.5.6.3.3	Macro Definition Documentation	678
3.5.6.3.3.1	MAPLE_FTPE_CMD_RX_POLL	678
3.5.6.3.3.2	MAPLE_FTPE_CMD_PRE_POST_UPDATE	678
3.5.6.3.3.3	MAPLE_FTPE_CMD_GET_PE_ACCUMULATOR	678
3.5.6.3.3.4	FTPE_BD_MANUAL_ACTIVATION	679
3.5.6.3.3.5	FTPE_BD_CFG0_FC_CG	679
3.5.6.3.3.6	FTPE_BD_CFG0_DSTZP	679
3.5.6.3.3.7	FTPE_BD_CFG0_OFFKB	679
3.5.6.3.3.8	FTPE_BD_CFG0_OFF16	679
3.5.6.3.3.9	FTPE_BD_PACKED_EXP_STAT	679
3.5.6.3.3.10	FTPE_BD_CFG0_PRBIE	679
3.5.6.3.3.11	FTPE_BD_CFG0_SOUS	679
3.5.6.3.3.12	FTPE_BD_COHERENCY_MODE	679
3.5.6.3.3.13	FTPE_BD_CFG0_INT_EN	679
3.5.6.3.3.14	FTPE_BD_CFG1_AIUS	679
3.5.6.3.3.15	FTPE_BD_CFG1_PROUR	680
3.5.6.3.3.16	FTPE_BD_CFG1_PSMUR	680
3.5.6.3.3.17	FTPE_BD_CFG1_ZERO_PAD	680
3.5.6.3.3.18	FTPE_BD_CFG1_SINS	680
3.5.6.3.3.19	FTPE_BD_CFG1_CPRE	680
3.5.6.3.3.20	FTPE_BD_CFG1_GRDC	680
3.5.6.3.3.21	FTPE_BD_CFG1_AFS	680
3.5.6.3.3.22	FTPE_BD_CFG1_PSTME	680
3.5.6.3.3.23	FTPE_BD_CFG1_CPIE	680
3.5.6.3.3.24	FTPE_BD_CFG1_GR_EN	681
3.5.6.3.3.25	FTPE_BD_CFG1_OVA_SCL	681
3.5.6.3.3.26	FTPE_BD_CFG1_SCL_TYPE	681
3.5.6.3.3.27	FTPE_BD_CFG1_ITE	681
3.5.6.3.3.28	FTPE_BD_CFG1_GI_EN	681
3.5.6.3.3.29	FTPE_BD_CFG3_PM_SCL	681
3.5.6.3.3.30	FTPE_BD_CFG3_USR_SCL_DWN_STG0	681
3.5.6.3.3.31	FTPE_BD_CFG3_USR_SCL_DWN_STG1	681
3.5.6.3.3.32	FTPE_BD_CFG3_USR_SCL_DWN_STG2	681
3.5.6.3.3.33	FTPE_BD_CFG3_USR_SCL_DWN_STG3	681
3.5.6.3.3.34	FTPE_BD_CFG3_USR_SCL_DWN_STG4	681
3.5.6.3.3.35	FTPE_BD_CFG3_USR_SCL_DWN_STG5	682
3.5.6.3.3.36	FTPE_BD_CFG3_IN_SCL	682
3.5.6.3.3.37	FTPE_BD_CFG3_ADP_OVA_SCL	682
3.5.6.3.3.38	FTPE_BD_CFG4_CPS	682
3.5.6.3.3.39	FTPE_BD_CFG4_BD_RPT	682
3.5.6.3.4	Enumeration Type Documentation	682
3.5.6.3.4.1	anonymous enum	682
3.5.6.3.4.2	maple_ftpe_post_vec_t	682

Section number	Title	Page
3.5.6.3.4.3	maple_ftpe_pre_vec_t	682
3.5.6.3.4.4	maple_ftpe_post_scale_t	683
3.5.6.3.4.5	maple_ftpe_extra_scale_t	683
3.5.6.3.4.6	maple_ftpe_data_size_set_t	683
3.5.6.3.4.7	maple_ftpe_axi_pr_t	683
3.5.7	Maple EQPE API	683
3.5.7.1	Overview	683
3.5.7.2	EQPE Initialization	684
3.5.7.2.1	Overview	684
3.5.7.2.2	Data Structure Documentation	684
3.5.7.2.2.1	struct eqpe_open_params_t	684
3.5.7.2.2.1.1	Field Documentation	685
3.5.7.2.2.1.2	maple_handle	685
3.5.7.2.2.1.3	maple_pe_bd_priority	685
3.5.7.2.2.1.4	maple_pe_num_bd	685
3.5.7.2.3	Macro Definition Documentation	685
3.5.7.2.3.1	EQPE_DEV_ID_0	685
3.5.7.2.3.2	EQPE_DEV_ID_1	685
3.5.7.2.3.3	EQPE_DEV_ID_2	685
3.5.7.2.3.4	EQPE_DEV_ID_3	685
3.5.7.2.3.5	EQPE_DEV_NAME_0	685
3.5.7.2.3.6	EQPE_DEV_NAME_1	685
3.5.7.2.3.7	EQPE_DEV_NAME_2	685
3.5.7.2.3.8	EQPE_DEV_NAME_3	686
3.5.7.2.4	Typedef Documentation	686
3.5.7.2.4.1	maple_eqpe_init_params_t	686
3.5.7.2.4.2	maple_eqpe_init_params_s	686
3.5.7.2.4.3	maple_eqpe_ch_open_params_t	686
3.5.7.2.5	Enumeration Type Documentation	686
3.5.7.2.5.1	anonymous enum	686
3.5.7.2.6	Function Documentation	686
3.5.7.2.6.1	mapleEqpeInitialize	686
3.5.7.3	EQPE Runtime	687
3.5.7.3.1	Overview	687
3.5.7.3.2	Macro Definition Documentation	688
3.5.7.3.2.1	EQPE_MAX_NUM_BD_FOR_DISPACTH	688
3.5.7.3.2.2	MAPLE_EQPE_CMD_GET_PE_ACCUMULATOR	688
3.5.7.3.2.3	MAPLE_EQPE_CMD_GET_ECC_EVENT_REG	688
3.5.7.3.2.4	MAPLE_EQPE_CMD_RX_POLL	688
3.5.7.3.3	Enumeration Type Documentation	688
3.5.7.3.3.1	maple_eqpe_ecc_event_flag_t	688
3.5.7.3.4	MAPLE EQPE BD	689
3.5.7.3.4.1	Overview	689
3.5.7.3.4.2	Data Structure Documentation	691
3.5.7.3.4.2.1	struct maple_eqpe_job_t	691

Section number	Title	Page
3.5.7.3.4.2.2	Field Documentation	692
3.5.7.3.4.2.3	first_flags	692
3.5.7.3.4.2.4	second_flags	692
3.5.7.3.4.2.5	matrix_dim	692
3.5.7.3.4.2.6	low_rr	693
3.5.7.3.4.2.7	high_rr	693
3.5.7.3.4.2.8	align_val	693
3.5.7.3.4.2.9	axi_pr	693
3.5.7.3.4.2.10	y_ptr	693
3.5.7.3.4.2.11	y_params	693
3.5.7.3.4.2.12	h_ptr	693
3.5.7.3.4.2.13	h_params	693
3.5.7.3.4.2.14	s_ptr	693
3.5.7.3.4.2.15	w_ptr	693
3.5.7.3.4.2.16	w_params	694
3.5.7.3.4.2.17	f_ptr	694
3.5.7.3.4.2.18	f_params	694
3.5.7.3.4.2.19	nb_ptr	694
3.5.7.3.4.2.20	c_ptr	694
3.5.7.3.4.2.21	scl_ba_ptr	694
3.5.7.3.4.2.22	scl_offsets	694
3.5.7.3.4.2.23	y_scl_ptr	694
3.5.7.3.4.2.24	out_ptr	694
3.5.7.3.4.2.25	out_scl_ptr	694
3.5.7.3.4.2.26	bd_index	695
3.5.7.3.4.3	Macro Definition Documentation	695
3.5.7.3.4.3.1	EQPE_BD_MANUAL_ACTIVATION	695
3.5.7.3.4.3.2	EQPE_BD_INT_EN	695
3.5.7.3.4.3.3	EQPE_BD_C_EN	695
3.5.7.3.4.3.4	EQPE_BD_W_PER_RX	695
3.5.7.3.4.3.5	EQPE_BD_RR2	695
3.5.7.3.4.3.6	EQPE_BD_RR3	695
3.5.7.3.4.3.7	EQPE_CONSECUTIVE_H	695
3.5.7.3.4.3.8	EQPE_NUM_H_PTRS	695
3.5.7.3.4.3.9	EQPE_NUM_F_PTRS	695
3.5.7.3.4.3.10	EQPE_NUM_OUT_PTRS	696
3.5.7.3.4.4	Enumeration Type Documentation	696
3.5.7.3.4.4.1	anonymous enum	696
3.5.7.3.4.4.2	anonymous enum	696
3.5.7.3.4.4.3	anonymous enum	696
3.5.7.3.4.4.4	anonymous enum	696
3.5.7.3.4.4.5	anonymous enum	697
3.5.7.3.4.4.6	anonymous enum	697
3.5.7.3.4.4.7	anonymous enum	697
3.5.7.3.4.4.8	anonymous enum	697

Section number	Title	Page
3.5.7.3.4.4.9	anonymous enum	698
3.5.7.3.4.4.10	anonymous enum	698
3.5.7.3.4.4.11	anonymous enum	698
3.5.7.3.4.5	Function Documentation	698
3.5.7.3.4.5.1	EQPE_BD_LX	698
3.5.7.3.4.5.2	EQPE_BD_RX	700
3.5.7.3.4.5.3	EQPE_BD_COLS	700
3.5.7.3.4.5.4	EQPE_BD_ROWS	700
3.5.7.3.4.5.5	EQPE_BD_C_SCL	701
3.5.7.3.4.5.6	EQPE_BD_W_COL	701
3.5.7.3.4.5.7	EQPE_BD_W_ROW	701
3.5.7.3.4.5.8	EQPE_BD_S_COL	702
3.5.7.3.4.5.9	EQPE_BD_S_ROW	702
3.5.7.3.4.5.10	EQPE_RR_SET_COLUMN	702
3.5.7.3.4.5.11	EQPE_BD_RR0	703
3.5.7.3.4.5.12	EQPE_BD_RR1	703
3.5.7.3.4.5.13	EQPE_BD_ALIGN_VAL	703
3.5.7.3.4.5.14	EQPE_BD_W_SIZE	704
3.5.7.3.4.5.15	EQPE_BD_S_SIZE	704
3.5.7.3.4.5.16	EQPE_BD_Y_OFFSET	704
3.5.7.3.4.5.17	EQPE_BD_Y_GAP	705
3.5.7.3.4.5.18	EQPE_BD_OUT_OFFSET0	705
3.5.7.3.4.5.19	EQPE_BD_OUT_OFFSET1	705
3.5.7.3.4.5.20	EQPE_BD_OUT_OFFSET2	706
3.5.7.3.4.5.21	EQPE_BD_OUT_OFFSET3	706
3.5.7.3.4.5.22	EQPE_BD_H_OFFSET	706
3.5.7.3.4.5.23	EQPE_BD_H_GAP	706
3.5.7.3.4.5.24	EQPE_BD_W_OFFSET	707
3.5.7.3.4.5.25	EQPE_BD_W_GAP	707
3.5.7.3.4.5.26	EQPE_BD_F_OFFSET	707
3.5.7.3.4.5.27	EQPE_BD_F_GAP	708
3.5.8	Maple DEPE API	709
3.5.8.1	Overview	709
3.5.8.2	DEPE Initialization	709
3.5.8.2.1	Overview	709
3.5.8.2.2	Data Structure Documentation	710
3.5.8.2.2.1	struct depe_open_params_t	710
3.5.8.2.2.1.1	Field Documentation	710
3.5.8.2.2.1.2	maple_handle	710
3.5.8.2.2.1.3	maple_pe_bd_priority	710
3.5.8.2.2.1.4	maple_pe_num_bd	710
3.5.8.2.3	Macro Definition Documentation	710
3.5.8.2.3.1	DEPE_DEV_ID_0	710
3.5.8.2.3.2	DEPE_DEV_ID_1	710
3.5.8.2.3.3	MAPLE_0_DEPE_NAME	710

Section number	Title	Page
3.5.8.2.3.4	MAPLE_1_DEPE_NAME	710
3.5.8.2.4	Typedef Documentation	711
3.5.8.2.4.1	maple_depe_init_params_t	711
3.5.8.2.4.2	maple_depe_init_params_s	711
3.5.8.2.4.3	maple_depe_ch_open_params_t	711
3.5.8.2.5	Enumeration Type Documentation	711
3.5.8.2.5.1	anonymous enum	711
3.5.8.2.6	Function Documentation	711
3.5.8.2.6.1	mapleDepeInitialize	711
3.5.8.3	DEPE Runtime	712
3.5.8.3.1	Overview	712
3.5.8.3.2	Macro Definition Documentation	712
3.5.8.3.2.1	DEPE_MAX_NUM_BD_FOR_DISPACTH	712
3.5.8.3.2.2	MAPLE_DEPE_CMD_GET_PE_ACCUMULATOR	712
3.5.8.3.2.3	MAPLE_DEPE_CMD_RX_POLL	713
3.5.8.3.3	MAPLE DEPE BD	713
3.5.8.3.3.1	Overview	713
3.5.8.3.3.2	Data Structure Documentation	714
3.5.8.3.3.2.1	struct maple_depe_job_t	714
3.5.8.3.3.2.2	Field Documentation	714
3.5.8.3.3.2.3	first_flags	714
3.5.8.3.3.2.4	input	714
3.5.8.3.3.2.5	output	715
3.5.8.3.3.2.6	header	715
3.5.8.3.3.2.7	sts	715
3.5.8.3.3.2.8	aux_flags	715
3.5.8.3.3.2.9	sep_vector	715
3.5.8.3.3.2.10	rmna	715
3.5.8.3.3.2.11	headers	715
3.5.8.3.3.2.12	bd_index	715
3.5.8.3.3.3	Macro Definition Documentation	716
3.5.8.3.3.3.1	DEPE_BD_MANUAL_ACTIVATION	716
3.5.8.3.3.3.2	DEPE_BD_INT_EN	716
3.5.8.3.3.3.3	DEPE_BD_COE	716
3.5.8.3.3.3.4	DEPE_TRANSPORT_BLOCK	716
3.5.8.3.3.3.5	DFPE_SVE_SVE	716
3.5.8.3.3.4	Enumeration Type Documentation	716
3.5.8.3.3.4.1	anonymous enum	716
3.5.8.3.3.4.2	anonymous enum	716
3.5.8.3.3.4.3	anonymous enum	717
3.5.8.3.3.4.4	anonymous enum	718
3.5.8.3.3.5	Function Documentation	718
3.5.8.3.3.5.1	DEPE_IBS	718
3.5.8.3.3.5.2	DEPE_NOH	718
3.5.8.3.3.5.3	DEPE_OBS	719

Section number	Title	Page
3.5.8.3.3.5.4	DEPE_PFS	719
3.5.8.3.3.5.5	DEPE_PSS	720
3.5.8.3.3.5.6	DEPE_OBO	720
3.5.8.3.3.5.7	MAPLE DEPE Headers	720
3.5.8.3.3.5.8	Overview	720
3.5.8.3.3.5.9	Data Structure Documentation	721
3.5.8.3.3.5.10	union depe_3glte_header_t	721
3.5.8.3.3.5.11	Field Documentation	721
3.5.8.3.3.5.12	nof	721
3.5.8.3.3.5.13	obo	721
3.5.8.3.3.5.14	ibo	721
3.5.8.3.3.5.15	cbsi	721
3.5.8.3.3.5.16	cm	721
3.5.8.3.3.5.17	lh	722
3.5.8.3.3.5.18	cd	722
3.5.8.3.3.5.19	noh	722
3.5.8.3.3.5.20	rmno	722
3.5.8.3.3.5.21	rms	722
3.5.8.3.3.5.22	rmni	722
3.5.8.3.3.5.23	union depe_wimax_header_t	722
3.5.8.3.3.5.24	Field Documentation	723
3.5.8.3.3.5.25	rmno	723
3.5.8.3.3.5.26	rms	723
3.5.8.3.3.5.27	cbsi	723
3.5.8.3.3.5.28	rd	723
3.5.8.3.3.5.29	lh	723
3.5.8.3.3.5.30	cd	723
3.5.8.3.3.5.31	noh	723
3.5.8.3.3.5.32	union depe_wimax16m_header_t	723
3.5.8.3.3.5.33	Field Documentation	724
3.5.8.3.3.5.34	clsb	724
3.5.8.3.3.5.35	rmno	724
3.5.8.3.3.5.36	clsy	724
3.5.8.3.3.5.37	clsw	724
3.5.8.3.3.5.38	cbsi	724
3.5.8.3.3.5.39	rd	724
3.5.8.3.3.5.40	lh	724
3.5.8.3.3.5.41	cd	724
3.5.8.3.3.5.42	noh	724
3.5.8.3.3.5.43	union depe_umts_header_t	725
3.5.8.3.3.5.44	Field Documentation	725
3.5.8.3.3.5.45	pfem2	725
3.5.8.3.3.5.46	cbs	725
3.5.8.3.3.5.47	bct	725
3.5.8.3.3.5.48	obo	725

Section number	Title	Page
3.5.8.3.3.5.49	ibo	725
3.5.8.3.3.5.50	cbc	725
3.5.8.3.3.5.51	p	726
3.5.8.3.3.5.52	bcm	726
3.5.8.3.3.5.53	rep	726
3.5.8.3.3.5.54	rrm	726
3.5.8.3.3.5.55	sd	726
3.5.8.3.3.5.56	lh	726
3.5.8.3.3.5.57	cd	726
3.5.8.3.3.5.58	noh	726
3.5.8.3.3.5.59	psem2	727
3.5.8.3.3.5.60	pem1	727
3.5.8.3.3.5.61	sei	727
3.5.8.3.3.5.62	sem	727
3.5.8.3.3.5.63	psei2	727
3.5.8.3.3.5.64	pfei2	727
3.5.8.3.3.5.65	psei1	727
3.5.8.3.3.5.66	pfei1	727
3.5.8.3.3.5.67	pso	728
3.5.8.3.3.5.68	pfo	728
3.5.8.3.3.5.69	ncb	728
3.5.8.3.3.5.70	epfei2	728
3.5.8.3.3.5.71	epsei2	728
3.5.8.3.3.5.72	epsei1	728
3.5.8.3.3.5.73	epfei1	728
3.5.8.3.3.5.74	epsem2	728
3.5.8.3.3.5.75	epem1	728
3.5.8.3.3.5.76	esei	729
3.5.8.3.3.5.77	esem	729
3.5.8.3.3.5.78	Enumeration Type Documentation	729
3.5.8.3.3.5.79	anonymous enum	729
3.5.9	Maple PUSCH API	729
3.5.9.1	Overview	729
3.5.9.1.1	PUSCH Initialization	730
3.5.9.1.1.1	Overview	730
3.5.9.1.1.2	Data Structure Documentation	730
3.5.9.1.1.2.1	struct pusch_open_params_t	730
3.5.9.1.1.2.2	Field Documentation	730
3.5.9.1.1.2.3	maple_handle	730
3.5.9.1.1.2.4	maple_pe_bd_priority	731
3.5.9.1.1.2.5	maple_pe_max_num_bd_rings	731
3.5.9.1.1.2.6	struct maple_pusch_ch_open_additional_interrupt	731
3.5.9.1.1.2.7	Field Documentation	731
3.5.9.1.1.2.8	int_enable	731
3.5.9.1.1.2.9	multiplexed_int	731

Section number	Title	Page
3.5.9.1.1.2.10	int_dispatcher	731
3.5.9.1.1.2.11	int_callback	731
3.5.9.1.1.2.12	int_num	731
3.5.9.1.1.2.13	int_priority	731
3.5.9.1.1.2.14	struct maple_pusch_ch_open_term_additional_interrupt	731
3.5.9.1.1.2.15	Field Documentation	732
3.5.9.1.1.2.16	int_enable	732
3.5.9.1.1.2.17	multiplexed_int	732
3.5.9.1.1.2.18	int_dispatcher	732
3.5.9.1.1.2.19	int_callback	732
3.5.9.1.1.2.20	int_num	732
3.5.9.1.1.2.21	int_priority	732
3.5.9.1.1.2.22	struct maple_pusch_ch_open_params_t	732
3.5.9.1.1.2.23	Field Documentation	733
3.5.9.1.1.2.24	pe_ch_params	733
3.5.9.1.1.2.25	ctrl_interrupt	733
3.5.9.1.1.2.26	done_cb	733
3.5.9.1.1.2.27	all_ctrl_done_cb	733
3.5.9.1.1.2.28	eq_term_interrupt	733
3.5.9.1.1.2.29	pu_term_interrupt	733
3.5.9.1.1.3	Macro Definition Documentation	733
3.5.9.1.1.3.1	PUSCH_DEV_ID_0	733
3.5.9.1.1.3.2	PUSCH_DEV_NAME_0	733
3.5.9.1.1.3.3	PUSCH_DEV_NAME_1	733
3.5.9.1.1.4	Typedef Documentation	733
3.5.9.1.1.4.1	maple_pusch_init_params_s	733
3.5.9.1.1.4.2	maple_pusch_init_params_t	734
3.5.9.1.1.4.3	add_int_cb	734
3.5.9.1.1.4.4	maple_pusch_int_cb_t	734
3.5.9.1.1.5	Enumeration Type Documentation	734
3.5.9.1.1.5.1	anonymous enum	734
3.5.9.1.1.6	Function Documentation	734
3.5.9.1.1.6.1	maplePuschInitialize	734
3.5.9.1.2	PUSCH Runtime	735
3.5.9.1.2.1	Overview	735
3.5.9.1.2.2	Macro Definition Documentation	740
3.5.9.1.2.2.1	MAPLE_PUSCH_CMD_CTRL_POLL	740
3.5.9.1.2.2.2	PUSCH_CH_CMD_GET_EQPE_STATUS	740
3.5.9.1.2.2.3	PUSCH_CH_CMD_GET_TERM_STATUS	740
3.5.9.1.2.2.4	PUSCH_CH_CMD_SET_USER_NEIGHBOR_READY	740
3.5.9.1.2.2.5	PUSCH_MAX_NUM_BD_FOR_DISPACTH	740
3.5.9.1.2.2.6	PUSCH_BD_MANUAL_ACTIVATION	740
3.5.9.1.2.2.7	PUSCH_BD_ALL_USERS_DONE_INT_EN	740
3.5.9.1.2.2.8	PUSCH_BD_ALL_CTRL_INT_EN	741
3.5.9.1.2.2.9	PUSCH_BD_ALL_EQ_TERM_INT_EN	741

Section number	Title	Page
3.5.9.1.2.2.10	PUSCH_BD_ALL_PU_TERM_INT_EN	741
3.5.9.1.2.2.11	PUSCH_BD_EXTENDED_JOB	741
3.5.9.1.2.2.12	PUSCH_NV_BETA_OUT_EN	741
3.5.9.1.2.2.13	PUSCH_COMP_EN	741
3.5.9.1.2.2.14	PUSCH_JOINT_EQ_EN	741
3.5.9.1.2.2.15	PUSCH_BD_EXCP_EN	741
3.5.9.1.2.2.16	PUSCH_BD_NV_BETA_SRC	741
3.5.9.1.2.2.17	PUSCH_BD_DECOUPLE	742
3.5.9.1.2.2.18	PUSCH_NUM_OF_ROWS	742
3.5.9.1.2.2.19	OUT_CQI_RI_DECODED	742
3.5.9.1.2.2.20	OUT_CQI_AMOUNT1	742
3.5.9.1.2.2.21	MODULATION_QPSK	742
3.5.9.1.2.2.22	MODULATION_16QAM	742
3.5.9.1.2.2.23	MODULATION_64QAM	742
3.5.9.1.2.2.24	PUSCH_SEGMENT_HEADER_LV0	742
3.5.9.1.2.2.25	PUSCH_SEGMENT_HEADER_LV1	742
3.5.9.1.2.2.26	PUSCH_SEGMENT_HEADER_LV2	742
3.5.9.1.2.2.27	PUSCH_SEGMENT_HEADER_LV3	742
3.5.9.1.2.2.28	PUSCH_SEGMENT_HEADER_C_EN	743
3.5.9.1.2.2.29	PUSCH_SEGMENT_HEADER_SN	743
3.5.9.1.2.2.30	PUSCH_SEGMENT_HEADER_LTC_0	743
3.5.9.1.2.2.31	PUSCH_SEGMENT_HEADER_LTC_1	743
3.5.9.1.2.2.32	PUSCH_SEGMENT_HEADER_LTC_2	743
3.5.9.1.2.2.33	PUSCH_SEGMENT_HEADER_LTC_3	743
3.5.9.1.2.2.34	PUSCH_SEGMENT_HEADER_USERS_DONE_0	743
3.5.9.1.2.2.35	PUSCH_SEGMENT_HEADER_USERS_DONE_1	743
3.5.9.1.2.2.36	PUSCH_SEGMENT_HEADER_USERS_DONE_2	743
3.5.9.1.2.2.37	PUSCH_SEGMENT_HEADER_USERS_DONE_3	743
3.5.9.1.2.2.38	PUSCH_SEGMENT_HEADER_USERS_DONE_BITMASK	744
3.5.9.1.2.2.39	PUSCH_SEGMENT_HEADER_W_OFFSET	744
3.5.9.1.2.2.40	PUSCH_SEGMENT_HEADER_W_GAP	744
3.5.9.1.2.2.41	PUSCH_SEGMENT_HEADER_F_OFFSET	744
3.5.9.1.2.2.42	PUSCH_SEGMENT_HEADER_F_GAP	744
3.5.9.1.2.2.43	PUSCH_SEGMENT_HEADER_SRS_EN	744
3.5.9.1.2.2.44	PUSCH_SEGMENT_HEADER_USER_RB_START	744
3.5.9.1.2.2.45	PUSCH_SEGMENT_HEADER_USER_RB_AMOUNT	744
3.5.9.1.2.2.46	PUSCH_SEGMENT_HEADER_USER_ID_IN_LAYER	745
3.5.9.1.2.2.47	PUSCH_SEGMENT_HEADER_GROUP0_MASK	745
3.5.9.1.2.2.48	PUSCH_SEGMENT_HEADER_GROUP1_MASK	745
3.5.9.1.2.2.49	PUSCH_SEGMENT_HEADER_GROUP2_MASK	745
3.5.9.1.2.2.50	PUSCH_SEGMENT_HEADER_GROUP3_MASK	745
3.5.9.1.2.2.51	PUSCH_SEGMENT_HEADER_GROUP4_MASK	745
3.5.9.1.2.2.52	PUSCH_SEGMENT_HEADER_GROUP5_MASK	745
3.5.9.1.2.2.53	PUSCH_SEGMENT_HEADER_GROUP6_MASK	745
3.5.9.1.2.2.54	PUSCH_SEGMENT_HEADER_GROUP7_MASK	745

Section number	Title	Page
3.5.9.1.2.2.55	PUSCH_USER_HEADER_DONE_INT_EN	746
3.5.9.1.2.2.56	PUSCH_USER_HEADER_CTRL_INT_EN	746
3.5.9.1.2.2.57	PUSCH_USER_HEADER_DISCARD_CW1	746
3.5.9.1.2.2.58	PUSCH_USER_HEADER_DISCARD_CW0	746
3.5.9.1.2.2.59	PUSCH_USER_HEADER_FH_ENABLE	746
3.5.9.1.2.2.60	PUSCH_USER_NV_BETA_OUT_EN	746
3.5.9.1.2.2.61	PUSCH_USER_HEADER_LAYER_0	746
3.5.9.1.2.2.62	PUSCH_USER_HEADER_LAYER_1	746
3.5.9.1.2.2.63	PUSCH_USER_HEADER_LAYER_2	746
3.5.9.1.2.2.64	PUSCH_USER_HEADER_LAYER_3	746
3.5.9.1.2.2.65	PUSCH_USER_HEADER_SRS_EN	746
3.5.9.1.2.2.66	PUSCH_USER_HEADER_CARRIER_0	746
3.5.9.1.2.2.67	PUSCH_USER_HEADER_CARRIER_1	747
3.5.9.1.2.2.68	PUSCH_USER_HEADER_CARRIER_2	747
3.5.9.1.2.2.69	PUSCH_USER_HEADER_CARRIER_3	747
3.5.9.1.2.2.70	PUSCH_USER_HEADER_CARRIER_4	747
3.5.9.1.2.2.71	PUSCH_USER_HEADER_HOE_CW_EN	747
3.5.9.1.2.2.72	PUSCH_USER_HEADER_LLMAP_CW_HYBRID	747
3.5.9.1.2.2.73	PUSCH_USER_HEADER_HAOE_CW_EN	747
3.5.9.1.2.2.74	PUSCH_USER_HEADER_FTH_CW_EN	747
3.5.9.1.2.2.75	PUSCH_USER_HEADER_CRC_A_STP_CW_EN	747
3.5.9.1.2.2.76	PUSCH_USER_HEADER_W1E_CW	747
3.5.9.1.2.2.77	PUSCH_USER_HEADER_W2E_CW	747
3.5.9.1.2.2.78	PUSCH_USER_HEADER_CQI_MAX_OUT_EN	747
3.5.9.1.2.3	Enumeration Type Documentation	748
3.5.9.1.2.3.1	anonymous enum	748
3.5.9.1.2.3.2	pusch_num_of_cols	748
3.5.9.1.2.3.3	pusch_num_ack_bits_t	748
3.5.9.1.2.3.4	pusch_segment_header_user_type_t	749
3.5.9.1.2.3.5	pusch_segment_header_cols_t	749
3.5.9.1.2.3.6	pusch_segment_header_rows_t	749
3.5.9.1.2.3.7	pusch_segment_header_weights_shared_columns_t	750
3.5.9.1.2.3.8	pusch_segment_header_w_per_rx_t	750
3.5.9.1.2.3.9	pusch_segment_header_w_row_t	750
3.5.9.1.2.3.10	pusch_segment_header_shared_columns_t	751
3.5.9.1.2.3.11	pusch_segment_header_shared_rows_t	751
3.5.9.1.2.3.12	pusch_segment_header_s_scl_type_t	751
3.5.9.1.2.3.13	pusch_segment_header_intrp_type_t	751
3.5.9.1.2.4	Function Documentation	752
3.5.9.1.2.4.1	PUSCH_SEGMENT_HEADER_USER_TYPE	752
3.5.9.1.2.4.2	PUSCH_SEGMENT_RB_START	752
3.5.10	Maple PUFFT API	752
3.5.10.1	Overview	752
3.5.10.2	PUFFT Initialization	753
3.5.10.2.1	Overview	753

Section number	Title	Page
3.5.10.2.2	Data Structure Documentation	754
3.5.10.2.2.1	struct pufft_open_params_t	754
3.5.10.2.2.1.1	Field Documentation	754
3.5.10.2.2.1.2	maple_handle	754
3.5.10.2.2.1.3	maple_pe_bd_priority	754
3.5.10.2.2.1.4	maple_pe_num_bd	754
3.5.10.2.2.1.5	antx_in_addr	754
3.5.10.2.2.1.6	antx_in_size	754
3.5.10.2.2.1.7	sector_config	755
3.5.10.2.2.1.8	pre_mult_addr	755
3.5.10.2.2.1.9	ack_addr	755
3.5.10.2.2.1.10	ack_data	755
3.5.10.2.2.1.11	soc_timer_conf	755
3.5.10.2.2.1.12	pre_mult_steering_bits	755
3.5.10.2.2.1.13	post_mult_addr	755
3.5.10.2.2.1.14	fc_shift_real	755
3.5.10.2.2.1.15	fc_shift_imag	755
3.5.10.2.2.1.16	fc_base_real	755
3.5.10.2.2.1.17	fc_base_imag	756
3.5.10.2.2.2	struct maple_pufft_ch_add_int_params_t	756
3.5.10.2.2.2.1	Field Documentation	756
3.5.10.2.2.2.2	int_enable	756
3.5.10.2.2.2.3	multiplexed_int	756
3.5.10.2.2.2.4	int_dispatcher	756
3.5.10.2.2.2.5	int_callback	756
3.5.10.2.2.2.6	int_num	756
3.5.10.2.2.2.7	int_priority	756
3.5.10.2.2.3	struct maple_pufft_ch_open_params_t	756
3.5.10.2.3	Macro Definition Documentation	757
3.5.10.2.3.1	PUFFT_INIT_OVFL_CNT	757
3.5.10.2.3.2	PUFFT_INIT_FTPE_INDX	757
3.5.10.2.3.3	PUFFT_DEV_ID_0	757
3.5.10.2.3.4	MAPLE_0_PUFFT_NAME	757
3.5.10.2.3.5	MAPLE_1_PUFFT_NAME	757
3.5.10.2.3.6	MAPLE_2_PUFFT_NAME	757
3.5.10.2.4	Typedef Documentation	757
3.5.10.2.4.1	maple_pufft_int_cb_t	757
3.5.10.2.4.2	maple_pufft_init_params_s	757
3.5.10.2.4.3	maple_pufft_init_params_t	757
3.5.10.2.5	Enumeration Type Documentation	757
3.5.10.2.5.1	anonymous enum	757
3.5.10.2.5.2	anonymous enum	758
3.5.10.2.6	Function Documentation	758
3.5.10.2.6.1	maplePufftInitialize	758
3.5.10.3	PUFFT Runtime	758

Section number	Title	Page
3.5.10.3.1	Overview	758
3.5.10.3.2	Data Structure Documentation	760
3.5.10.3.2.1	struct maple_pufft_job_t	760
3.5.10.3.2.1.1	Field Documentation	760
3.5.10.3.2.1.2	first_flags	760
3.5.10.3.2.1.3	ant_status	760
3.5.10.3.2.1.4	ant_status_short	760
3.5.10.3.2.1.5	timing_adj	760
3.5.10.3.2.1.6	status_ptr	760
3.5.10.3.2.1.7	second_flags	760
3.5.10.3.2.1.8	antx_input	761
3.5.10.3.2.1.9	output	761
3.5.10.3.2.1.10	output_gap	761
3.5.10.3.2.1.11	bd_index	761
3.5.10.3.2.2	union maple_pufft_status_t	761
3.5.10.3.2.3	struct pufft_mpisr_ch_params_t	761
3.5.10.3.3	Macro Definition Documentation	761
3.5.10.3.3.1	PUFFT_SET_MPISR_CHANNELS	761
3.5.10.3.3.2	PUFFT_SET_MPISR_TID	761
3.5.10.3.3.3	PUFFT_MAX_NUM_BD_FOR_DISPACTH	761
3.5.10.3.3.4	PUFFT_BD_MANUAL_ACTIVATION	761
3.5.10.3.3.5	PUFFT_INT_EN	761
3.5.10.3.3.6	PUFFT_FC_GEN_EN	762
3.5.10.3.3.7	PUFFT_OVA_SCL	762
3.5.10.3.3.8	PUFFT_ACK_INT_EN	762
3.5.10.3.3.9	PUFFT_ANT_MASK	762
3.5.10.3.3.10	PUFFT_ADP_OVA_SCL	762
3.5.10.3.4	Enumeration Type Documentation	762
3.5.10.3.4.1	anonymous enum	762
3.5.10.3.4.2	pufft_axi_priority_t	762
3.5.10.3.4.3	maple_pufft_pre_vec_t	763
3.5.10.3.4.4	maple_pufft_post_vec_t	763
3.5.10.3.4.5	anonymous enum	763
3.5.10.3.4.6	maple_pufft_alt_ubw_t	763
3.5.11	Maple PDSCH API	764
3.5.11.1	Overview	764
3.5.11.2	Data Structure Documentation	764
3.5.11.2.1	struct pdsch_sector_queue_param_t	764
3.5.11.2.1.1	Field Documentation	765
3.5.11.2.1.1.1	queue	765
3.5.11.2.1.1.2	write_ptr	765
3.5.11.2.1.1.3	read_ptr	765
3.5.11.2.2	struct pdsch_dbg_status_param_t	765
3.5.11.2.2.1	Field Documentation	765
3.5.11.2.2.1.1	status	765

Section number	Title	Page
3.5.11.2.2.1.2	error	765
3.5.11.2.2.1.3	sec_num	765
3.5.11.2.3	struct maple_pdsch_pid_t	766
3.5.11.2.3.1	Field Documentation	766
3.5.11.2.3.1.1	reserved	766
3.5.11.2.3.1.2	cw_data_offsets	766
3.5.11.2.3.1.3	data	766
3.5.11.3	Macro Definition Documentation	766
3.5.11.3.1	PDSCH_MAX_NUM_BD_FOR_DISPACTH	766
3.5.11.4	Enumeration Type Documentation	766
3.5.11.4.1	pdsch_sbw_to_rb	766
3.5.11.5	MAPLE PDSCH BD	767
3.5.11.5.1	Overview	767
3.5.11.5.2	Data Structure Documentation	771
3.5.11.5.2.1	struct maple_pdsch_csi_rs_header_t	771
3.5.11.5.2.1.1	Field Documentation	771
3.5.11.5.2.1.2	csi_rs_header	771
3.5.11.5.2.1.3	csi_cge	771
3.5.11.5.2.1.4	csi_cgm	771
3.5.11.5.2.1.5	csi_rs_ports	771
3.5.11.5.2.1.6	cs_rs_ontop	771
3.5.11.5.2.1.7	nidcsi	772
3.5.11.5.2.1.8	csi_rs_conf	772
3.5.11.5.2.2	struct maple_pdsch_csrs_paramters_t	772
3.5.11.5.2.2.1	Field Documentation	772
3.5.11.5.2.2.2	cs_rs_header	772
3.5.11.5.2.2.3	cs_rs_ontop	772
3.5.11.5.2.2.4	cs_rs_ports	772
3.5.11.5.2.2.5	cs_cge	772
3.5.11.5.2.2.6	cs_cgm	772
3.5.11.5.2.2.7	cell_id	772
3.5.11.5.2.3	union maple_pdsch_job_extension_t	772
3.5.11.5.2.3.1	Field Documentation	773
3.5.11.5.2.3.2	extension_header	773
3.5.11.5.2.3.3	import_addr	773
3.5.11.5.2.3.4	noit	773
3.5.11.5.2.3.5	csrs_header_addr	773
3.5.11.5.2.3.6	__pad0__	773
3.5.11.5.2.3.7	fc_address	773
3.5.11.5.2.3.8	nprsrben0	773
3.5.11.5.2.3.9	nprsrben1	774
3.5.11.5.2.4	struct maple_pdsch_job_t	774
3.5.11.5.2.4.1	Field Documentation	774
3.5.11.5.2.4.2	first_flags	774
3.5.11.5.2.4.3	second_flags	774

Section number	Title	Page
3.5.11.5.2.4.4	third_flags	774
3.5.11.5.2.4.5	hdr_data_ptrs	775
3.5.11.5.2.4.6	hdr_config	775
3.5.11.5.2.4.7	bd_index	775
3.5.11.5.2.4.8	syms_done	775
3.5.11.5.2.4.9	extension_ptr	775
3.5.11.5.2.5	struct maple_pdsch_user_header_t	775
3.5.11.5.2.5.1	Field Documentation	776
3.5.11.5.2.5.2	first_flags	776
3.5.11.5.2.5.3	second_flags	776
3.5.11.5.2.5.4	third_flags	776
3.5.11.5.2.5.5	tb_size	776
3.5.11.5.2.5.6	num_out_bits	776
3.5.11.5.2.5.7	tb_input	776
3.5.11.5.2.6	struct maple_pdsch_cw_header_t	776
3.5.11.5.2.6.1	Field Documentation	777
3.5.11.5.2.6.2	cw_init	777
3.5.11.5.2.6.3	cw_mod_type	777
3.5.11.5.2.6.4	cw_trgt_layer	777
3.5.11.5.2.6.5	gain_b_exp	777
3.5.11.5.2.6.6	gain_a_exp	777
3.5.11.5.2.6.7	gain_b_mantisa	777
3.5.11.5.2.6.8	gain_a_mantisa	777
3.5.11.5.2.7	struct maple_pdsch_sub_frm_ctrl_t	777
3.5.11.5.2.7.1	Field Documentation	778
3.5.11.5.2.7.2	bf_offset	778
3.5.11.5.2.7.3	nrbf	778
3.5.11.5.2.7.4	roa	778
3.5.11.5.2.7.5	bfe	778
3.5.11.5.2.7.6	ta	778
3.5.11.5.2.7.7	long_elm	778
3.5.11.5.2.7.8	gain_exp	778
3.5.11.5.2.7.9	gain_mantisa	778
3.5.11.5.2.8	struct maple_pdsch_ofdm_sym_ctrl_t	778
3.5.11.5.2.8.1	Field Documentation	779
3.5.11.5.2.8.2	long_elm	779
3.5.11.5.2.8.3	nor	779
3.5.11.5.2.8.4	dp	779
3.5.11.5.2.9	struct maple_pdsch_long_re_ctrl_t	779
3.5.11.5.2.9.1	Field Documentation	779
3.5.11.5.2.9.2	real	779
3.5.11.5.2.9.3	k	779
3.5.11.5.2.9.4	imaginary	779
3.5.11.5.2.9.5	same_k	779
3.5.11.5.2.10	struct maple_pdsch_short_re_ctrl_t	779

Section number	Title	Page
3.5.11.5.2.10.1	Field Documentation	780
3.5.11.5.2.10.2	real	780
3.5.11.5.2.10.3	k	780
3.5.11.5.2.11	struct maple_pdsch_res_block_en_t	780
3.5.11.5.2.12	struct maple_pdsch_ctrl_gain_rnti_t	780
3.5.11.5.2.12.1	Field Documentation	780
3.5.11.5.2.12.2	cge0	780
3.5.11.5.2.12.3	cgm0	780
3.5.11.5.2.12.4	cge1	781
3.5.11.5.2.12.5	cgm1	781
3.5.11.5.2.12.6	uospms	781
3.5.11.5.2.12.7	uospmf	781
3.5.11.5.2.12.8	epdcch	781
3.5.11.5.2.12.9	roa	781
3.5.11.5.2.12.10	nop	781
3.5.11.5.2.12.11	sauet	781
3.5.11.5.2.12.12	sp_ant	781
3.5.11.5.2.12.13	ue5_nprb	781
3.5.11.5.2.12.14	dmrsid	781
3.5.11.5.2.12.15	rnti	782
3.5.11.5.2.12.16	scid	782
3.5.11.5.2.13	struct maple_pdsch_sm_w_t	782
3.5.11.5.2.14	struct maple_pdsch_ss_header_t	782
3.5.11.5.2.14.1	Field Documentation	782
3.5.11.5.2.14.2	data_re	782
3.5.11.5.2.14.3	k0_re	782
3.5.11.5.2.14.4	data_img	782
3.5.11.5.2.14.5	k0_img	782
3.5.11.5.3	Macro Definition Documentation	782
3.5.11.5.3.1	PDSCH_BD_MANUAL_ACTIVATION	782
3.5.11.5.3.2	PDSCH_BD_INT_EN	782
3.5.11.5.3.3	PDSCH_BD_SYM_INT_EN	783
3.5.11.5.3.4	PDSCH_BD_EXTND_EN	783
3.5.11.5.3.5	PDSCH_BD_PH_EN	783
3.5.11.5.3.6	PDSCH_BD_N_USERS	783
3.5.11.5.3.7	PDSCH_USERS_NUM	783
3.5.11.5.3.8	PDSCH_BD_SSF_EN	783
3.5.11.5.3.9	PDSCH_SSFC	783
3.5.11.5.3.10	PDSCH_BD_SF_NUM	783
3.5.11.5.3.11	PDSCH_BD_NT	783
3.5.11.5.3.12	PDSCH_BD_EXT_SYM_IN_SCALE	783
3.5.11.5.3.13	PDSCH_BD_UE_EN	783
3.5.11.5.3.14	PDSCH_BD_CS_EN	784
3.5.11.5.3.15	PDSCH_BD_MBSFN_EN	784
3.5.11.5.3.16	PDSCH_BD_POS_EN	784

Section number	Title	Page
3.5.11.5.3.17	PDSCH_BD_PSS_EN	784
3.5.11.5.3.18	PDSCH_BD_SSS_EN	784
3.5.11.5.3.19	PDSCH_BD_PBCH_EN	784
3.5.11.5.3.20	PDSCH_BD_SPM_EN	784
3.5.11.5.3.21	PDSCH_BD_CSI_EN	784
3.5.11.5.3.22	PDSCH_BD_POD_EN	784
3.5.11.5.3.23	PDSCH_BD_PID_EN	784
3.5.11.5.3.24	PDSCH_BD_PRM_EN	784
3.5.11.5.3.25	PDSCH_BD_PSM_EN	785
3.5.11.5.3.26	PDSCH_BD_EXT_PAD_EN	785
3.5.11.5.3.27	PDSCH_BD_NUM_SYMS	785
3.5.11.5.3.28	PDSCH_BD_ADP_OVA_SCL	785
3.5.11.5.3.29	PDSCH_BD_GAIN_EXPONENT	785
3.5.11.5.3.30	PDSCH_BD_GAIN_MANTISSA	785
3.5.11.5.3.31	PDSCH_PDPE_OUTPUT_SCALE	785
3.5.11.5.3.32	PDSCH_BD_NUM_DATA_OFDM_SYM	785
3.5.11.5.3.33	PDSCH_BD_NUM_CTRL_OFDM_SYM	785
3.5.11.5.3.34	PDSCH_BD_NUM_UE_SPECIFIC_RBEN	785
3.5.11.5.3.35	PDSCH_BD_NUM_UE_SPECIFIC_PAR	785
3.5.11.5.3.36	PDSCH_BD_N_PRS_RB	786
3.5.11.5.3.37	PDSCH_BD_CSI_NUM_HDR	786
3.5.11.5.3.38	PDSCH_BD_CSI_NUM_RBEN	786
3.5.11.5.3.39	PDSCH_BD_CSRS_NUM_HDR	786
3.5.11.5.3.40	PDSCH_BD_CSRS_NUM_RBEN	786
3.5.11.5.3.41	PDSCH_CSI_HDR_STRUCT_GEN	786
3.5.11.5.3.42	PDSCH_UE_RS_HDR_STRUCT_GEN	786
3.5.11.5.3.43	PDSCH_UE_RS_HDR_STRUCT_GEN	786
3.5.11.5.3.44	PDSCH_SOSD	786
3.5.11.5.3.45	PDSCH_USER_RB_START	786
3.5.11.5.3.46	PDSCH_USER_ANT_EN	787
3.5.11.5.3.47	PDSCH_USER_NUM_PHYS_RB	787
3.5.11.5.3.48	PDSCH_USER_OSPMS	787
3.5.11.5.3.49	PDSCH_USER_OSPMF	787
3.5.11.5.3.50	PDSCH_USER_LARGE_DELAY_CDD	787
3.5.11.5.3.51	PDSCH_USER_CSI_OVAL_ACT	787
3.5.11.5.3.52	PDSCH_USER_NIR	787
3.5.11.5.3.53	PDSCH_USER_RV_IDX_TB0	787
3.5.11.5.3.54	PDSCH_USER_RV_IDX_TB1	787
3.5.11.5.3.55	PDSCH_RB_MAP_STRUCT_GEN	787
3.5.11.5.3.56	PDSCH_HDR_RES_BLOCK_EN	787
3.5.11.5.3.57	PDSCH_CGAIN_UOSPMS	788
3.5.11.5.3.58	PDSCH_CGAIN_UOSPMF	788
3.5.11.5.3.59	PDSCH_CGAIN_NUM_PORTS	788
3.5.11.5.3.60	PDSCH_CGAIN_SP_ANT	788
3.5.11.5.3.61	PDSCH_CGAIN_UE5NPRB	788

Section number	Title	Page
3.5.11.5.3.62	PDSCH_SM_W_RNLV	788
3.5.11.5.3.63	PDSCH_SM_W_INLV	788
3.5.11.5.3.64	PDSCH_SM_W_SWLV	788
3.5.11.5.3.65	PDSCH_SM_W_CNPLV	788
3.5.11.5.4	Enumeration Type Documentation	788
3.5.11.5.4.1	anonymous enum	788
3.5.11.5.4.2	anonymous enum	789
3.5.11.5.4.3	anonymous enum	790
3.5.11.5.4.4	anonymous enum	790
3.5.11.5.4.5	anonymous enum	790
3.5.11.5.4.6	anonymous enum	790
3.5.11.5.4.7	anonymous enum	790
3.5.11.5.4.8	anonymous enum	791
3.5.11.5.4.9	anonymous enum	791
3.5.11.5.4.10	anonymous enum	791
3.5.11.5.4.11	anonymous enum	791
3.5.11.5.4.12	anonymous enum	791
3.5.11.5.4.13	anonymous enum	791
3.5.11.5.4.14	anonymous enum	792
3.5.11.5.4.15	anonymous enum	792
3.5.11.5.5	Function Documentation	792
3.5.11.5.5.1	PDSCH_RB_MAP_TABLE_SIZE	792
3.5.11.5.5.2	PDSCH_RB_MAP_SLOT_SIZE	792
3.5.12	Maple TCPE API	793
3.5.12.1	Overview	793
3.5.12.2	TCPE Initialization	793
3.5.12.2.1	Overview	793
3.5.12.2.2	Data Structure Documentation	794
3.5.12.2.2.1	struct tcpe_open_params_t	794
3.5.12.2.2.1.1	Field Documentation	794
3.5.12.2.2.1.2	maple_handle	794
3.5.12.2.2.1.3	maple_pe_bd_priority	794
3.5.12.2.2.1.4	maple_pe_num_bd	794
3.5.12.2.3	Macro Definition Documentation	794
3.5.12.2.3.1	MAPLE_TCPE_NAME	794
3.5.12.2.3.2	TCPE_DEV_ID	794
3.5.12.2.4	Typedef Documentation	794
3.5.12.2.4.1	maple_tcpe_init_params_t	794
3.5.12.2.4.2	maple_tcpe_init_params_s	794
3.5.12.2.4.3	maple_tcpe_ch_open_params_t	794
3.5.12.2.5	Enumeration Type Documentation	795
3.5.12.2.5.1	anonymous enum	795
3.5.12.2.6	Function Documentation	795
3.5.12.2.6.1	mapleTcpeInitialize	795
3.5.12.3	TCPE Runtime	795

Section number	Title	Page
3.5.12.3.1	Overview	795
3.5.12.3.2	Data Structure Documentation	798
3.5.12.3.2.1	struct maple_tcpe_ucb_t	798
3.5.12.3.2.1.1	Field Documentation	798
3.5.12.3.2.1.2	nchip	798
3.5.12.3.2.1.3	pilot_id	799
3.5.12.3.2.1.4	mode	799
3.5.12.3.2.1.5	fc_en	799
3.5.12.3.2.1.6	sw	799
3.5.12.3.2.1.7	cmbl	799
3.5.12.3.2.1.8	last	799
3.5.12.3.2.1.9	acc_scl	799
3.5.12.3.2.1.10	nca_scl	799
3.5.12.3.2.1.11	ca_scl	799
3.5.12.3.2.1.12	acc_init	799
3.5.12.3.2.1.13	chip_offset	799
3.5.12.3.2.1.14	combining_strategy	799
3.5.12.3.2.1.15	sso	799
3.5.12.3.2.1.16	sst	800
3.5.12.3.2.1.17	ssn	800
3.5.12.3.2.1.18	fc_init_real	800
3.5.12.3.2.1.19	fc_init_imag	800
3.5.12.3.2.1.20	fc_step_real	800
3.5.12.3.2.1.21	fc_step_imag	800
3.5.12.3.2.1.22	ep	800
3.5.12.3.2.1.23	ccn	800
3.5.12.3.2.1.24	fc_step_size	800
3.5.12.3.2.1.25	nca_weight0	800
3.5.12.3.2.1.26	nca_weight1	800
3.5.12.3.2.1.27	nca_weight2	800
3.5.12.3.2.1.28	nca_weight3	800
3.5.12.3.2.2	struct maple_tcpe_uiobc_t	801
3.5.12.3.2.2.1	Field Documentation	801
3.5.12.3.2.2.2	u_out_addr	801
3.5.12.3.2.2.3	out_steer	801
3.5.12.3.2.2.4	u_acc_addr	801
3.5.12.3.2.2.5	acc_steer	801
3.5.12.3.2.2.6	u_out_size	801
3.5.12.3.2.2.7	u_acc_size	801
3.5.12.3.2.3	struct maple_tcpe_job_t	801
3.5.12.3.2.3.1	Field Documentation	802
3.5.12.3.2.3.2	axi_priority	802
3.5.12.3.2.3.3	num_users	802
3.5.12.3.2.3.4	ant_fetch_size	802
3.5.12.3.2.3.5	antenna_buff_size	802

Section number	Title	Page
3.5.12.3.2.3.6	antenna_fetch_off	802
3.5.12.3.2.3.7	antenna0_base_addr	802
3.5.12.3.2.3.8	antenna1_base_addr	802
3.5.12.3.2.3.9	user_conf_buffer	802
3.5.12.3.2.3.10	user_init_out_buff_conf	802
3.5.12.3.2.3.11	bd_index	802
3.5.12.3.3	Macro Definition Documentation	803
3.5.12.3.3.1	TCPE_MAX_NUM_BD_FOR_DISPACTH	803
3.5.12.3.3.2	TCPE_BD_MANUAL_ACTIVATION	803
3.5.12.3.3.3	TCPE_BD_INT_EN	803
3.5.12.3.3.4	TCPE_MODE_PATH_SEARCH	803
3.5.12.3.3.5	TCPE_MODE_RACH	803
3.5.12.3.3.6	TCPE_FC_DISABLE	803
3.5.12.3.3.7	TCPE_FC_ENABLE	803
3.5.12.3.3.8	TCPE_LAST_USER	803
3.5.12.3.3.9	TCPE_CS	803
3.5.12.3.3.10	TCPE_SSO_DONT_ADD_4096_CG	804
3.5.12.3.3.11	TCPE_SSO_ADD_4096_CG	804
3.5.12.3.3.12	TCPE_SST_SHORT	804
3.5.12.3.3.13	TCPE_SST_LONG	804
3.5.12.3.3.14	TCPE_EP	804
3.5.12.3.3.15	TCPE_EP_IQ_MAP_I	804
3.5.12.3.3.16	TCPE_EP_IQ_MAP_Q	804
3.5.12.3.3.17	TCPE_FC_STEP_SIZE	804
3.5.12.3.4	Enumeration Type Documentation	804
3.5.12.3.4.1	anonymous enum	804
3.5.12.3.4.2	anonymous enum	805
3.5.12.3.4.3	anonymous enum	805
3.5.12.3.4.4	anonymous enum	805
3.5.12.3.4.5	anonymous enum	807
3.5.12.3.4.6	anonymous enum	807
3.5.12.3.4.7	anonymous enum	808
3.5.12.3.4.8	anonymous enum	808
3.5.13	Maple Trace Events API	808
3.5.13.1	Overview	808
3.5.13.2	Maple Trace Events Runtime	809
3.5.13.2.1	Overview	809
3.5.13.2.2	Data Structure Documentation	809
3.5.13.2.2.1	struct maple_trace_event_t	809
3.5.13.2.2.1.1	Field Documentation	809
3.5.13.2.2.1.2	trc_id	809
3.5.13.2.2.1.3	__pad0__	810
3.5.13.2.2.1.4	time	810
3.5.13.2.2.2	struct maple_pdsch_trace_event_t	810
3.5.13.2.2.2.1	Field Documentation	810

Section number	Title	Page
3.5.13.2.2.2.2	trace_id	810
3.5.13.2.2.2.3	task_id	810
3.5.13.2.2.2.4	symbol_ix	810
3.5.13.2.2.2.5	status	810
3.5.13.2.2.2.6	__pad0__	810
3.5.13.2.2.2.7	time	810
3.5.13.2.2.3	struct maple_trace_read_buffers_param_t	811
3.5.13.2.2.3.1	Field Documentation	811
3.5.13.2.2.3.2	event_queue	811
3.5.13.2.2.3.3	offset	811
3.5.13.2.2.3.4	num_of_events	811
3.5.13.2.2.4	struct maple_pe_load_param_t	811
3.5.13.2.2.4.1	Field Documentation	811
3.5.13.2.2.4.2	maple_clk_counter	811
3.5.13.2.2.4.3	pe_load	811
3.5.13.2.3	Enumeration Type Documentation	811
3.5.13.2.3.1	maple_trace_event_enable_t	811
3.5.13.2.3.2	maple_trace_id_t	812
3.5.13.2.3.3	maple_pe_load_type_t	812
3.6	**OCeaN DMA Module API**	**812**
3.6.1	Overview	812
3.6.2	OCeaN DMA Module API Initialization	813
3.6.2.1	Overview	813
3.6.2.2	Data Structure Documentation	813
3.6.2.2.1	struct ocn_dma_atmu_config_t	813
3.6.2.2.1.1	Field Documentation	814
3.6.2.2.1.1.1	win_base_addr	814
3.6.2.2.1.1.2	win_size	814
3.6.2.2.1.1.3	win_interface	814
3.6.2.2.2	struct ocn_dma_init_params_t	814
3.6.2.2.2.1	Field Documentation	814
3.6.2.2.2.1.1	dma_id	814
3.6.2.2.2.1.2	irq_priority	814
3.6.2.2.2.1.3	dma_channel_enable	814
3.6.2.2.2.1.4	reserved	814
3.6.2.3	Enumeration Type Documentation	815
3.6.2.3.1	ocn_dma_id_t	815
3.6.2.3.2	ocn_dma_win_sz_t	815
3.6.2.3.3	ocn_dma_interface_t	816
3.6.2.4	Function Documentation	816
3.6.2.4.1	ocnDmaInitialize	816
3.6.3	OCeaN DMA Module API Runtime	816
3.6.3.1	Overview	816
3.6.3.2	Data Structure Documentation	818

Section number	Title	Page
3.6.3.2.1	struct ocn_dma_stride_t	818
3.6.3.2.1.1	Field Documentation	818
3.6.3.2.1.1.1	stride_size	818
3.6.3.2.1.1.2	stride_distance	819
3.6.3.2.2	struct ocn_dma_chain_config_t	819
3.6.3.2.2.1	Field Documentation	819
3.6.3.2.2.1.1	max_num_of_transfers	819
3.6.3.2.2.1.2	source_stride	819
3.6.3.2.2.1.3	destination_stride	819
3.6.3.2.2.1.4	heap	819
3.6.3.2.3	struct ocn_dma_channel_config_t	819
3.6.3.2.3.1	Field Documentation	820
3.6.3.2.3.1.1	priority	820
3.6.3.2.3.1.2	ocn_dma_callback	820
3.6.3.2.3.1.3	interrupt_en	820
3.6.3.2.4	union ocn_dma_desc_t	820
3.6.3.2.4.1	Field Documentation	820
3.6.3.2.4.1.1	satr	820
3.6.3.2.4.1.2	sar	820
3.6.3.2.4.1.3	datr	820
3.6.3.2.4.1.4	dar	820
3.6.3.2.4.1.5	enlndar	820
3.6.3.2.4.1.6	nlndar	820
3.6.3.2.4.1.7	bcr	821
3.6.3.2.4.1.8	ext_next_list_desc	821
3.6.3.2.4.1.9	next_list_desc	821
3.6.3.2.4.1.10	ext_first_link_descriptor	821
3.6.3.2.4.1.11	first_link_descriptor	821
3.6.3.2.4.1.12	source_stride	821
3.6.3.2.4.1.13	destination_stride	821
3.6.3.2.5	struct ocn_dma_buff_t	821
3.6.3.2.5.1	Field Documentation	821
3.6.3.2.5.1.1	full_addr	821
3.6.3.2.5.1.2	high_addr	822
3.6.3.2.5.1.3	low_addr	822
3.6.3.2.5.1.4	addr	822
3.6.3.2.5.1.5	stride_enable	822
3.6.3.2.5.1.6	snoop_enable	822
3.6.3.2.5.1.7	enhanced_enable	822
3.6.3.2.6	struct ocn_dma_t	822
3.6.3.2.6.1	Field Documentation	822
3.6.3.2.6.1.1	device_id	822
3.6.3.2.6.1.2	ocn_dma_ch_map	822
3.6.3.2.6.1.3	channel_state	823
3.6.3.2.6.1.4	bound_chain	823

Section number	Title	Page
3.6.3.2.6.1.5	ocn_dma_callback	823
3.6.3.2.6.1.6	channel_mr	823
3.6.3.3	Macro Definition Documentation	823
3.6.3.3.1	OCN_DMA_WRITE	823
3.6.3.3.2	OCN_DMA_READ	823
3.6.3.3.3	DMA_LIST_DESC_ADDR_MASK	823
3.6.3.3.4	DMA_LINK_DESC_LAST	823
3.6.3.3.5	OCN_ATMU_DEFAULT_WIN	823
3.6.3.3.6	OCN_DMA_MR_CC	823
3.6.3.3.7	OCN_DMA_MR_CS	824
3.6.3.3.8	OCN_DMA_XFER_ERROR	824
3.6.3.3.9	OCN_DMA_HALTED	824
3.6.3.3.10	OCN_DMA_PROG_ERROR	824
3.6.3.3.11	OCN_DMA_END_LINK	824
3.6.3.3.12	OCN_DMA_BUSY	824
3.6.3.3.13	OCN_DMA_END_SEGMENT	824
3.6.3.3.14	OCN_DMA_END_LIST	824
3.6.3.3.15	OCN_DMA_COMPLETED	824
3.6.3.3.16	OCN_DMA_HAS_ERROR	825
3.6.3.3.17	DMA_END_SEG_INT_EN	825
3.6.3.3.18	DMA_END_LINK_INT_EN	825
3.6.3.3.19	DMA_END_LIST_INT_EN	825
3.6.3.3.20	DMA_ERROR_INT_EN	825
3.6.3.3.21	ocn_buff_addr	825
3.6.3.3.22	ocnDmaChannelBound	825
3.6.3.4	Enumeration Type Documentation	826
3.6.3.4.1	ocn_dma_ch_state_t	826
3.6.3.4.2	ocn_dma_bwc_t	826
3.6.3.4.3	ocn_dma_response_t	826
3.6.3.5	Function Documentation	827
3.6.3.5.1	ocnDmaControllerOpen	827
3.6.3.5.2	ocnDmaChainCreate	827
3.6.3.5.3	ocnDmaTransfer	828
3.6.3.5.4	ocnDmaChainTransferAdd	828
3.6.3.5.5	ocnDmaChannelStart	829
3.6.3.5.6	ocnDmaChannelIsActive	830
3.6.3.5.7	ocnDmaChannelOpen	830
3.6.3.5.8	ocnDmaChannelClose	831
3.6.3.5.9	ocnDmaChannelBind	831
3.6.3.5.10	ocnDmaChannelUnbind	832
3.6.3.5.11	ocnDmaTransferSizeSet	832
3.6.3.5.12	ocnDmaTransferLowSourceSet	832
3.6.3.5.13	ocnDmaTransferLowDestinationSet	833
3.6.3.5.14	ocnDmaTransferHighDestinationSet	834
3.6.3.5.15	ocnDmaChainNextSet	834

Section number	Title	Page
3.6.3.5.16	ocnDmaChainLastSet	834
3.6.3.5.17	ocnDmaChainReset	834
3.6.3.5.18	ocnDmaTransferWait	835
3.6.3.5.19	ocnDmaChannelContinue	835
3.7	**DPAA Module API**	**835**
3.7.1	Overview	835
3.7.2	Data Structure Documentation	839
3.7.2.1	struct dpaa_fd_t	839
3.7.3	Macro Definition Documentation	839
3.7.3.1	QM_MAX_NUM_OF_POOL_CHANNELS	839
3.7.3.2	QM_MAX_NUM_OF_WQ	839
3.7.3.3	QM_MAX_NUM_OF_CGS	840
3.7.3.4	QM_MAX_NUM_OF_FQIDS	840
3.7.3.5	DPAA_FD_DD_MASK	840
3.7.3.6	DPAA_FD_PID_MASK	840
3.7.3.7	DPAA_FD_ELIODN_MASK	840
3.7.3.8	DPAA_FD_BPID_MASK	840
3.7.3.9	DPAA_FD_ADDRH_MASK	840
3.7.3.10	DPAA_FD_ADDRL_MASK	840
3.7.3.11	DPAA_FD_FORMAT_MASK	840
3.7.3.12	DPAA_FD_OFFSET_MASK	840
3.7.3.13	DPAA_FD_LENGTH_MASK	841
3.7.3.14	DPAA_FD_GET_DD	841
3.7.3.15	DPAA_FD_GET_PID	841
3.7.3.16	DPAA_FD_GET_BPID	841
3.7.3.17	DPAA_FD_GET_ADDRH	841
3.7.3.18	DPAA_FD_GET_ADDRL	841
3.7.3.19	DPAA_FD_GET_PHYS_ADDR	841
3.7.3.20	DPAA_FD_GET_FORMAT	841
3.7.3.21	DPAA_FD_GET_OFFSET	842
3.7.3.22	DPAA_FD_GET_LENGTH	842
3.7.3.23	DPAA_FD_GET_STATUS	842
3.7.3.24	DPAA_FD_GET_ADDR	842
3.7.3.25	DPAA_FD_SET_DD	842
3.7.3.26	DPAA_FD_SET_BPID	842
3.7.3.27	DPAA_FD_SET_ADDRH	842
3.7.3.28	DPAA_FD_SET_ADDRL	843
3.7.3.29	DPAA_FD_SET_ADDR	843
3.7.3.30	DPAA_FD_SET_FORMAT	843
3.7.3.31	DPAA_FD_SET_OFFSET	843
3.7.3.32	DPAA_FD_SET_LENGTH	843
3.7.3.33	DPAA_FD_SET_STATUS	843
3.7.4	Enumeration Type Documentation	843
3.7.4.1	dpaa_sw_portal_t	843

Section number	Title	Page
3.7.4.2	rx_store_response_t	843
3.7.4.3	dpaa_dc_portal_t	844
3.7.5	Variable Documentation	844
3.7.5.1	id	844
3.7.5.2	addrl	844
3.7.5.3	length	844
3.7.5.4	status	844
3.7.5.5	major_rev	844
3.7.5.6	minor_rev	844
3.7.5.7	rejection_code	845
3.7.5.8	cg_id	845
3.7.5.9	cg_id	845
3.7.5.10	cg	845
3.7.5.11	cg	845
3.7.5.12	res	845
3.7.5.13	guest_id	845
3.7.5.14	sw_portals_base_address	845
3.7.5.15	liodn	845
3.7.5.16	total_num_of_fqids	845
3.7.5.17	fqd_mem_partition_id	846
3.7.5.18	pfdr_mem_partition_id	846
3.7.5.19	f_exception	846
3.7.5.20	h_app	846
3.7.5.21	err_irq	846
3.7.5.22	part_fqid_base	846
3.7.5.23	part_num_of_fqids	846
3.7.5.24	part_cgs_base	846
3.7.5.25	part_num_of_cgs	846
3.7.5.26	stash_dest_queue	847
3.7.5.27	eqcr	847
3.7.5.28	eqcr_high_pri	847
3.7.5.29	dqrr	847
3.7.5.30	dqrr_liodn	847
3.7.5.31	dqrr_high_pri	847
3.7.5.32	fd_fq	847
3.7.5.33	fd_fq_liodn	847
3.7.5.34	fd_fq_high_pri	848
3.7.5.35	fd_fq_drop	848
3.7.5.36	ce_base_address	848
3.7.5.37	ci_base_address	848
3.7.5.38	h_qm	848
3.7.5.39	sw_portal_id	848
3.7.5.40	irq	848
3.7.5.41	fd_liodn_offset	848
3.7.5.42	f_dflt_frame	849

Section number	Title	Page
3.7.5.43	f_rejected_frame	849
3.7.5.44	h_app	849
3.7.5.45	h_qm_cg	849
3.7.5.46	overhead_accounting_length	849
3.7.5.47	fq_tail_drop_threshold	849
3.7.5.48	h_qm	849
3.7.5.49	h_qm_portal	849
3.7.5.50	init_parked	849
3.7.5.51	hold_active	849
3.7.5.52	avoid_blocking	850
3.7.5.53	prefer_in_cache	850
3.7.5.54	use_context_a_for_stash	850
3.7.5.55	frame_annotation_size	850
3.7.5.56	frame_annotation_size	850
3.7.5.57	frame_data_size	850
3.7.5.58	frame_data_size	850
3.7.5.59	fq_context_size	850
3.7.5.60	fq_context_size	850
3.7.5.61	fq_context_addr	851
3.7.5.62	fq_context_addr	851
3.7.5.63	context_a	851
3.7.5.64	context_a	851
3.7.5.65	context_b	851
3.7.5.66	channel	851
3.7.5.67	wq	851
3.7.5.68	shadow_mode	851
3.7.5.69	num_of_fqids	851
3.7.5.70	use_force	851
3.7.5.71	align	852
3.7.5.72	align	852
3.7.5.73	fqid	852
3.7.5.74	fqid	852
3.7.5.75	congestion_avoidance_enable	852
3.7.5.76	congestion_avoidance_params	852
3.7.5.77	mod_options	852
3.7.5.78	hold_active	852
3.7.5.79	avoid_blocking	853
3.7.5.80	prefer_in_cache	853
3.7.5.81	use_context_a_for_stash	853
3.7.5.82	frame_annotation_size	853
3.7.5.83	frame_annotation_size	853
3.7.5.84	frame_data_size	853
3.7.5.85	frame_data_size	853
3.7.5.86	fq_context_size	853
3.7.5.87	fq_context_size	853

Section number	Title	Page
3.7.5.88	fq_context_addr	854
3.7.5.89	fq_context_addr	854
3.7.5.90	context_a	854
3.7.5.91	context_a	854
3.7.5.92	context_b	854
3.7.5.93	channel	854
3.7.5.94	wq	854
3.7.5.95	congestion_avoidance_enable	854
3.7.5.96	congestion_avoidance_params	854
3.7.5.97	f_completion_cb	855
3.7.5.98	deliver_frame	855
3.7.5.99	f_callback	855
3.7.5.100	h_app	855
3.7.5.101	max_th	855
3.7.5.102	min_th	855
3.7.5.103	probability_denominator	855
3.7.5.104	h_qm	855
3.7.5.105	h_qm_portal	855
3.7.5.106	frame_count	856
3.7.5.107	wred_enable	856
3.7.5.108	wred_params	856
3.7.5.109	tail_drop_enable	856
3.7.5.110	threshold	856
3.7.5.111	notify_dc_portal	856
3.7.5.112	dc_portal_id	856
3.7.5.113	f_exception	856
3.7.5.114	h_app	857
3.7.6	Buffer Manager Module API	857
3.7.6.1	Overview	857
3.7.6.2	Buffer Manager General	857
3.7.6.2.1	Overview	857
3.7.6.2.2	Buffer Manager General Runtime	857
3.7.6.2.2.1	Overview	857
3.7.6.2.2.2	Enumeration Type Documentation	858
3.7.6.2.2.2.1	bm_counters_t	858
3.7.6.2.2.3	Function Documentation	858
3.7.6.2.2.3.1	bmCounterGet	858
3.7.6.2.3	Buffer Manager General Initialization	858
3.7.6.2.3.1	Overview	858
3.7.6.2.3.2	Data Structure Documentation	859
3.7.6.2.3.2.1	struct bm_param_t	859
3.7.6.2.3.2.2	Field Documentation	859
3.7.6.2.3.2.3	guest_id	859
3.7.6.2.3.2.4	liodn	859
3.7.6.2.3.2.5	total_num_of_buffers	859

Section number	Title	Page
3.7.6.2.3.2.6	fbpr_mem_partition_id	860
3.7.6.2.3.2.7	f_exception	860
3.7.6.2.3.2.8	h_app	860
3.7.6.2.3.2.9	err_irq	860
3.7.6.2.3.2.10	part_bpid_base	860
3.7.6.2.3.2.11	part_num_of_pools	860
3.7.6.2.3.2.12	struct bm_revision_info_t	860
3.7.6.2.3.2.13	Field Documentation	860
3.7.6.2.3.2.14	major_rev	860
3.7.6.2.3.2.15	minor_rev	861
3.7.6.2.3.3	Typedef Documentation	861
3.7.6.2.3.3.1	bmExceptionsCallback	861
3.7.6.2.3.4	Enumeration Type Documentation	861
3.7.6.2.3.4.1	bm_exceptions_t	861
3.7.6.2.3.5	Function Documentation	861
3.7.6.2.3.5.1	bmConfig	861
3.7.6.2.3.5.2	bmInit	862
3.7.6.2.3.5.3	bmFree	862
3.7.6.2.3.5.4	bmRevisionGet	862
3.7.6.2.3.5.5	bmErrorIsr	863
3.7.6.2.3.5.6	bmFbprThresholdConfig	863
3.7.6.3	BM-Pool API	864
3.7.6.3.1	Overview	864
3.7.6.3.2	BM-Pool Runtime Control Unit	864
3.7.6.3.2.1	Overview	864
3.7.6.3.2.2	Enumeration Type Documentation	864
3.7.6.3.2.2.1	bm_pool_counters_t	864
3.7.6.3.2.3	Function Documentation	864
3.7.6.3.2.3.1	bmPoolIdGet	864
3.7.6.3.2.3.2	bmPoolBufferContextGet	866
3.7.6.3.2.3.3	bmPoolPhysToVirt	866
3.7.6.3.2.3.4	bmPoolVirtToPhys	866
3.7.6.3.2.3.5	bmPOOLCounterGet	867
3.7.6.3.2.3.6	bmPoolBufGet	867
3.7.6.3.2.3.7	bmPoolBufPut	867
3.7.6.3.3	BM-Pool Initialization Unit	868
3.7.6.3.3.1	Overview	868
3.7.6.3.3.2	Data Structure Documentation	869
3.7.6.3.3.2.1	struct buffer_pool_info_t	869
3.7.6.3.3.2.2	Field Documentation	869
3.7.6.3.3.2.3	buffer_size	869
3.7.6.3.3.2.4	phys_to_virt	869
3.7.6.3.3.2.5	virt_to_phys	869
3.7.6.3.3.2.6	struct bm_pool_param_t	869
3.7.6.3.3.2.7	Field Documentation	869

Section number	Title	Page
3.7.6.3.3.2.8	bm	869
3.7.6.3.3.2.9	bm_portal	869
3.7.6.3.3.2.10	h_app	869
3.7.6.3.3.2.11	shadow_mode	870
3.7.6.3.3.2.12	bpid	870
3.7.6.3.3.3	Macro Definition Documentation	870
3.7.6.3.3.3.1	BM_POOL_DEP_THRESH_SW_ENTRY	870
3.7.6.3.3.3.2	BM_POOL_DEP_THRESH_SW_EXIT	870
3.7.6.3.3.3.3	BM_POOL_DEP_THRESH_HW_ENTRY	870
3.7.6.3.3.3.4	BM_POOL_DEP_THRESH_HW_EXIT	870
3.7.6.3.3.3.5	MAX_DEPLETION_THRESHOLDS	870
3.7.6.3.3.4	Typedef Documentation	870
3.7.6.3.3.4.1	phys_to_virt_t	870
3.7.6.3.3.4.2	virt_to_phys_t	870
3.7.6.3.3.4.3	bm_depletion_callback_t	872
3.7.6.3.3.5	Function Documentation	872
3.7.6.3.3.5.1	bmPoolConfig	872
3.7.6.3.3.5.2	bmPoolInit	872
3.7.6.3.3.5.3	bmPoolBpidConfig	873
3.7.6.3.3.5.4	bmPoolDepletionConfig	873
3.7.6.3.3.5.5	bmPoolStockpileConfig	874
3.7.6.3.3.5.6	bmPoolBuffContextModeConfig	874
3.7.6.3.3.5.7	bmPoolFree	874
3.7.6.4	BMAN portal Unit	875
3.7.6.4.1	Overview	875
3.7.6.4.2	BM-Portal Initialization Unit	875
3.7.6.4.2.1	Overview	875
3.7.6.4.2.2	Data Structure Documentation	875
3.7.6.4.2.2.1	struct bm_portal_params_t	875
3.7.6.4.2.2.2	Field Documentation	876
3.7.6.4.2.2.3	ce_base_address	876
3.7.6.4.2.2.4	ci_base_address	876
3.7.6.4.2.2.5	h_bm	876
3.7.6.4.2.2.6	sw_portal_id	876
3.7.6.4.2.2.7	irq	876
3.7.6.4.2.3	Function Documentation	876
3.7.6.4.2.3.1	bmPortalConfig	876
3.7.6.4.2.3.2	bmPortalInit	876
3.7.6.4.2.3.3	bmPortalFree	877
3.7.6.4.2.3.4	bmPortalConfigMemAttr	877
3.7.7	Queue Manager Module API	878
3.7.7.1	Overview	878
3.7.7.2	QM General API	878
3.7.7.2.1	Overview	878
3.7.7.2.2	QM General Initialization	878

Section number	Title	Page
3.7.7.2.2.1	Overview	878
3.7.7.2.2.2	Data Structure Documentation	879
3.7.7.2.2.2.1	struct qm_dc_portal_params_t	879
3.7.7.2.2.2.2	struct qm_revision_info_t	879
3.7.7.2.2.2.3	struct qm_rejected_frame_info_t	879
3.7.7.2.2.2.4	struct qm_context_a_t	880
3.7.7.2.2.2.5	struct qm_param_t	880
3.7.7.2.2.3	Typedef Documentation	880
3.7.7.2.2.3.1	qm_exceptions_callback_t	880
3.7.7.2.2.3.2	qm_received_frame_callback_t	880
3.7.7.2.2.3.3	qm_fqr_drained_completion_cb_t	881
3.7.7.2.2.3.4	qm_rejected_frame_callback_t	881
3.7.7.2.2.3.5	qm_context_b_t	882
3.7.7.2.2.4	Enumeration Type Documentation	882
3.7.7.2.2.4.1	qm_rejection_code_t	882
3.7.7.2.2.4.2	qm_exceptions_t	882
3.7.7.2.2.5	Function Documentation	883
3.7.7.2.2.5.1	qmConfig	883
3.7.7.2.2.5.2	qmInit	883
3.7.7.2.2.5.3	qmFree	883
3.7.7.2.2.5.4	qmConfigRTFramesDepth	884
3.7.7.2.2.5.5	qmConfigPfdrThreshold	884
3.7.7.2.2.5.6	qmConfigSfdrReservationThreshold	885
3.7.7.2.3	QM General Runtime	886
3.7.7.2.3.1	Overview	886
3.7.7.2.3.2	Data Structure Documentation	886
3.7.7.2.3.2.1	struct qm_rsrv_fqr_params_t	886
3.7.7.2.3.2.2	Field Documentation	887
3.7.7.2.3.2.3	use_force	887
3.7.7.2.3.2.4	num_of_fqids	887
3.7.7.2.3.2.5	align	887
3.7.7.2.3.2.6	non_frc_qs	887
3.7.7.2.3.2.7	fqid	887
3.7.7.2.3.2.8	frc_q	887
3.7.7.2.3.2.9	qs	887
3.7.7.2.3.2.10	struct qm_error_info_t	887
3.7.7.2.3.2.11	Field Documentation	888
3.7.7.2.3.2.12	portal_valid	888
3.7.7.2.3.2.13	hw_portal	888
3.7.7.2.3.2.14	sw_portal_id	888
3.7.7.2.3.2.15	dcp_id	888
3.7.7.2.3.2.16	fqid_valid	888
3.7.7.2.3.2.17	fqid	888
3.7.7.2.3.2.18	struct qm_wq_class_scheduler_params_t	888
3.7.7.2.3.2.19	Field Documentation	889

Section number	Title	Page
3.7.7.2.3.2.20	channel_type	889
3.7.7.2.3.2.21	dcp_id	889
3.7.7.2.3.2.22	low_tier_over_medium_tier_ratio	889
3.7.7.2.3.2.23	wq_2_weight	889
3.7.7.2.3.2.24	wq_3_weight	889
3.7.7.2.3.2.25	wq_4_weight	889
3.7.7.2.3.2.26	wq_5_weight	889
3.7.7.2.3.2.27	wq_6_weight	889
3.7.7.2.3.2.28	wq_7_weight	889
3.7.7.2.3.3	Enumeration Type Documentation	889
3.7.7.2.3.3.1	qm_counters_t	889
3.7.7.2.3.3.2	qm_channel_type_t	890
3.7.7.2.3.4	Function Documentation	890
3.7.7.2.3.4.1	qmReserveQueues	890
3.7.7.2.3.4.2	qmUnreserveQueues	890
3.7.7.2.3.4.3	qmSetException	891
3.7.7.2.3.4.4	qmErrorIsr	891
3.7.7.2.3.4.5	qmGetErrorInformation	892
3.7.7.2.3.4.6	qmGetCounter	892
3.7.7.2.3.4.7	qmGetRevision	893
3.7.7.2.3.4.8	qmSetWqClassScheduler	893
3.7.7.3	QM-Portal API	894
3.7.7.3.1	Overview	894
3.7.7.3.2	QM-Portal Initialization	894
3.7.7.3.2.1	Overview	894
3.7.7.3.2.2	Data Structure Documentation	894
3.7.7.3.2.2.1	struct qm_portal_stash_param_t	894
3.7.7.3.2.2.2	struct qm_portal_param_t	895
3.7.7.3.2.3	Enumeration Type Documentation	895
3.7.7.3.2.3.1	qm_portal_poll_source_t	895
3.7.7.3.2.4	Function Documentation	895
3.7.7.3.2.4.1	qmPortalConfig	895
3.7.7.3.2.4.2	qmPortalInit	896
3.7.7.3.2.4.3	qmPortalFree	896
3.7.7.3.2.4.4	qmPortalConfigDcaMode	896
3.7.7.3.2.4.5	qmPortalConfigStash	897
3.7.7.3.2.4.6	qmPortalConfigDequeueRingSize	897
3.7.7.3.2.4.7	qmPortalConfigPullMode	897
3.7.7.3.3	QM-Portal Runtime	898
3.7.7.3.3.1	Overview	898
3.7.7.3.3.2	Data Structure Documentation	898
3.7.7.3.3.2.1	struct qm_portal_frame_info_t	898
3.7.7.3.3.3	Function Documentation	898
3.7.7.3.3.3.1	qmPortalSetDequeueInterruptCoalescing	898
3.7.7.3.3.3.2	qmPortalPoll	899

Section number	Title	Page
3.7.7.3.3.3.3	qmPortalPollFrame	899
3.7.7.4	QM Frame-Queue-Range API	900
3.7.7.4.1	Overview	900
3.7.7.4.2	Macro Definition Documentation	900
3.7.7.4.2.1	QM_FQR_ALL_QUEUES	900
3.7.7.4.3	Enumeration Type Documentation	900
3.7.7.4.3.1	qm_fq_channel_t	900
3.7.7.4.4	QM-FQR Initialization	901
3.7.7.4.4.1	Overview	901
3.7.7.4.4.2	Data Structure Documentation	902
3.7.7.4.4.2.1	struct qm_fqr_congestion_avoidance_params_t	902
3.7.7.4.4.2.2	struct qm_fqr_params_t	902
3.7.7.4.4.2.3	struct qm_fqr_modify_params_t	902
3.7.7.4.4.3	Macro Definition Documentation	903
3.7.7.4.4.3.1	QM_FQR_MOD_OPT_HOLD_ACTIVE	903
3.7.7.4.4.3.2	QM_FQR_MOD_OPT_AVOID_BLOCKING	903
3.7.7.4.4.3.3	QM_FQR_MOD_OPT_PREFER_IN_CACHE	903
3.7.7.4.4.3.4	QM_FQR_MOD_OPT_STASH_PARAMS	903
3.7.7.4.4.3.5	QM_FQR_MOD_OPT_CTXA	903
3.7.7.4.4.3.6	QM_FQR_MOD_OPT_CTXB	903
3.7.7.4.4.3.7	QM_FQR_MOD_OPT_CH	903
3.7.7.4.4.3.8	QM_FQR_MOD_OPT_WQ	903
3.7.7.4.4.3.9	QM_FQR_MOD_OPT_CG	903
3.7.7.4.4.4	Typedef Documentation	904
3.7.7.4.4.4.1	qm_fqr_mod_opt_t	904
3.7.7.4.4.5	Enumeration Type Documentation	904
3.7.7.4.4.5.1	qm_fq_channel_t	904
3.7.7.4.4.6	Function Documentation	904
3.7.7.4.4.6.1	qmFqrCreate	904
3.7.7.4.4.6.2	qmFqrFree	905
3.7.7.4.4.6.3	qmFqrFreeWDrain	905
3.7.7.4.4.6.4	qmFqrModify	906
3.7.7.4.5	QM-FQR Runtime	906
3.7.7.4.5.1	Overview	906
3.7.7.4.5.2	Enumeration Type Documentation	906
3.7.7.4.5.2.1	qm_fqr_counters_t	906
3.7.7.4.5.3	Function Documentation	907
3.7.7.4.5.3.1	qmFqrRegisterCB	907
3.7.7.4.5.3.2	qmFqrGetFqid	907
3.7.7.4.5.3.3	qmFqrGetCounter	907
3.7.7.4.5.3.4	qmFqrEnqueue	908
3.7.7.4.5.4	QM Congestion Group API	908
3.7.7.4.5.4.1	Overview	908
3.7.7.4.5.4.2	QM-Congestion Group Initialization	909
3.7.7.4.5.4.3	Overview	909

Section number	Title	Page
3.7.7.4.5.4.4	Data Structure Documentation	909
3.7.7.4.5.4.5	struct qm_cg_wred_curve_t	909
3.7.7.4.5.4.6	struct qm_cg_wred_params_t	909
3.7.7.4.5.4.7	struct qm_cg_params_t	909
3.7.7.4.5.4.8	Function Documentation	910
3.7.7.4.5.4.9	qmCgCreate	910
3.7.7.4.5.4.10	qmCgFree	910
3.7.7.4.5.4.11	QM-CG Runtime	910
3.7.7.4.5.4.12	Overview	910
3.7.7.4.5.4.13	Data Structure Documentation	911
3.7.7.4.5.4.14	struct qm_cg_modify_wred_params_t	911
3.7.7.4.5.4.15	Enumeration Type Documentation	911
3.7.7.4.5.4.16	qm_cg_color_t	911
3.7.7.4.5.4.17	Function Documentation	911
3.7.7.4.5.4.18	qmCgSetException	911
3.7.7.4.5.4.19	qmCgModifyWredCurve	911
3.7.7.4.5.4.20	qmCgModifyTailDropThreshold	913
3.7.7.5	PDSCH Initialization	913
3.7.7.5.1	Overview	913
3.7.7.5.2	Data Structure Documentation	917
3.7.7.5.2.1	struct pdsch_config_param_t	917
3.7.7.5.2.1.1	Field Documentation	917
3.7.7.5.2.1.2	mnos	917
3.7.7.5.2.1.3	cell_id	917
3.7.7.5.2.1.4	mbsfn_id	917
3.7.7.5.2.2	struct pdsch_open_params_t	917
3.7.7.5.2.2.1	Field Documentation	917
3.7.7.5.2.2.2	maple_handle	917
3.7.7.5.2.2.3	antx_out_addr	918
3.7.7.5.2.2.4	antx_out_size	918
3.7.7.5.2.2.5	cell_config	918
3.7.7.5.2.2.6	bf_config	918
3.7.7.5.2.3	struct maple_pdsch_ch_add_int_params_t	918
3.7.7.5.2.3.1	Field Documentation	918
3.7.7.5.2.3.2	int_enable	918
3.7.7.5.2.3.3	multiplexed_int	918
3.7.7.5.2.3.4	int_dispatcher	918
3.7.7.5.2.3.5	int_callback	918
3.7.7.5.2.3.6	int_num	919
3.7.7.5.2.3.7	int_priority	919
3.7.7.5.2.4	struct maple_pdsch_ch_open_params_t	919
3.7.7.5.2.4.1	Field Documentation	919
3.7.7.5.2.4.2	flags	919
3.7.7.5.2.4.3	late_pcr_en	919
3.7.7.5.3	Macro Definition Documentation	919

Section number	Title	Page
3.7.7.5.3.1	PDSCH_MAX_USER_PER_SLOT	919
3.7.7.5.3.2	PDSCH_CW_PER_USER	919
3.7.7.5.3.3	PDSCH_VANT_LAYERS	919
3.7.7.5.3.4	MAPLE_PDSCH_CONFIG_BF_EN	919
3.7.7.5.3.5	MAPLE_PDSCH_CONFIG_BF_OFFSET	919
3.7.7.5.3.6	PDSCH_SYNC_ENALE	920
3.7.7.5.3.7	PDSCH_IFFT_DISABLE	920
3.7.7.5.3.8	PDSCH_DNOSD	920
3.7.7.5.3.9	PDSCH_MNOSEM_EN	920
3.7.7.5.3.10	PDSCH_MODE_ALT_DBW_EN	920
3.7.7.5.3.11	PDSCH_CYCLIC_ENALE	920
3.7.7.5.3.12	PDSCH_CYCLIC_ENABLE	920
3.7.7.5.3.13	PDSCH_BF_ENABLE	920
3.7.7.5.4	Typedef Documentation	920
3.7.7.5.4.1	maple_pdsch_int_cb_t	920
3.7.7.5.4.2	maple_pdsch_init_params_t	920
3.7.7.5.4.3	maple_pdsch_init_params_s	921
3.7.7.5.5	Enumeration Type Documentation	921
3.7.7.5.5.1	anonymous enum	921
3.7.7.5.5.2	anonymous enum	921
3.7.7.5.5.3	anonymous enum	922
3.7.7.5.5.4	anonymous enum	923
3.7.7.5.5.5	anonymous enum	923
3.7.7.5.5.6	anonymous enum	923
3.7.7.5.5.7	pdsch_mode_alt_dbw_t	923
3.7.7.5.5.8	anonymous enum	924
3.7.7.5.5.9	anonymous enum	924
3.7.7.5.5.10	anonymous enum	924
3.7.7.5.5.11	anonymous enum	925
3.7.7.5.5.12	anonymous enum	925
3.7.7.5.5.13	anonymous enum	925
3.7.7.5.5.14	anonymous enum	925
3.7.7.5.5.15	pdsch_mode_sys_bw_t	926
3.7.7.5.6	Function Documentation	926
3.7.7.5.6.1	PDSCH_SECTOR_BA	926
3.7.7.5.6.2	maplePdschInitialize	926
3.7.7.6	PDSCH Runtime	927
3.7.7.6.1	Overview	927
3.7.7.6.2	Data Structure Documentation	928
3.7.7.6.2.1	struct pdsch_sec_reconfig_param_t	928
3.7.7.6.2.1.1	Field Documentation	928
3.7.7.6.2.1.2	cell_config	928
3.7.7.6.2.1.3	sec_num	929
3.7.7.6.2.1.4	ant_out_size	929
3.7.7.6.2.1.5	antx_out_addr	929

Section number	Title	Page
3.7.7.6.2.1.6	sss_tgt	929
3.7.7.6.2.1.7	pss_tgt	929
3.7.7.6.2.1.8	alt_dbw_en	929
3.7.7.6.2.1.9	alt_dbw	929
3.7.7.6.2.1.10	alt_bw_conf	929
3.7.7.6.2.1.11	dnosiba	929
3.7.7.6.2.1.12	dnosid	929
3.7.7.6.2.1.13	pos_tgt	929
3.7.7.6.2.1.14	mbsfn_tgt	930
3.7.7.6.2.1.15	sr	930
3.7.7.6.2.1.16	s_ba	930
3.7.7.6.2.1.17	cs_rs_ports	930
3.7.7.6.2.1.18	cyclic_out	930
3.7.7.6.2.1.19	efsel	930
3.7.7.6.2.1.20	pbch_ports	930
3.7.7.6.2.1.21	s_nva	930
3.7.7.6.2.1.22	s_npa	930
3.7.7.6.2.1.23	s_bfe	930
3.7.7.6.2.1.24	pdsch_api	930
3.7.7.6.2.1.25	bf_config	931
3.7.7.6.2.2	struct maple_pdsch_alt_dbw_conf_t	931
3.7.7.6.2.2.1	Field Documentation	931
3.7.7.6.2.2.2	alt_dbw_en	931
3.7.7.6.2.2.3	alt_dbw	931
3.7.7.6.2.2.4	alt_dbw_conf	931
3.7.7.6.2.2.5	sec_num	931
3.7.7.6.2.2.6	cell_set	931
3.7.7.6.3	Macro Definition Documentation	932
3.7.7.6.3.1	PDSCH_DEV_CMD_RECONF_SEC	932
3.7.7.6.3.2	PDSCH_DEV_CMD_RECONF_SEC_NON_BLOCKING	932
3.7.7.6.3.3	PDSCH_DEV_CMD_FULL_RECONF_SEC	932
3.7.7.6.3.4	PDSCH_DEV_CMD_FULL_RECONF_SEC_NON_BLOCKING	932
3.7.7.6.3.5	PDSCH_DEV_CMD_PAD_DATA_SET	932
3.7.7.6.3.6	PDSCH_DEV_CMD_PAD_DATA_SET_NON_BLOCKING	932
3.7.7.6.3.7	PDSCH_DEV_CMD_DBG_GET_SECTOR_QUEUES	932
3.7.7.6.3.8	PDSCH_DEV_CMD_SET_CELL_CONFIG_0	932
3.7.7.6.3.9	PDSCH_DEV_CMD_SET_CELL_CONFIG_1	933
3.7.7.6.3.10	PDSCH_DEV_CMD_CONFIG_ALT_DBW	933
3.7.7.6.3.11	PDSCH_DEV_CMD_DBG_GET_SECTOR_STATUS	933
3.7.7.6.3.12	PDSCH_DEV_CMD_PAD_DATA1_SET	933
3.7.7.6.3.13	PDSCH_DEV_CMD_PAD_DATA2_SET	933
3.7.7.6.3.14	PDSCH_CH_CMD_RX_POLL	933
3.7.7.6.3.15	PDSCH_CH_CMD_VIRT_TRANS_ENABLE	933
3.7.7.6.3.16	PDSCH_CH_CMD_VIRT_TRANS_DISABLE	933
3.7.7.6.3.17	PDSCH_CH_CMD_GO	933

Section number	Title	Page
3.7.7.6.3.18	PDSCH_CH_CMD_INFO_GET	933
3.7.7.6.3.19	PDSCH_CH_CMD_GET_NUM_SYMS_DONE	934
3.7.7.6.3.20	PDSCH_CH_CMD_UPDATE_BEFORE_LATE_PCR	934
3.7.7.6.3.21	PDSCH_CH_CMD_UPDATE_BEFORE_EXT	934
3.7.7.6.3.22	PDSCH_DEV_CMD_RECONF_SEC_MASK	934
3.7.7.7	VSG Memory Map	934
3.7.7.7.1	Overview	934
3.7.7.7.2	Data Structure Documentation	934
3.7.7.7.2.1	struct vint_map_t	934
3.7.7.7.2.1.1	Field Documentation	935
3.7.7.7.2.1.2	vigr	935
3.7.7.7.2.1.3	visr	935
3.7.7.7.2.2	struct hs_map_t	935
3.7.7.7.2.2.1	Field Documentation	935
3.7.7.7.2.2.2	hsmpr	935
3.7.7.7.2.3	struct gpio_map_t	935
3.7.7.7.2.3.1	Field Documentation	935
3.7.7.7.2.3.2	podr	935
3.7.7.7.2.3.3	pdat	935
3.7.7.7.2.3.4	pdir	935
3.7.7.7.2.3.5	par	936
3.7.7.7.2.3.6	psor	936
3.7.7.8	B486x Heterogeneous API	936
3.7.7.8.1	Overview	936
3.7.7.8.2	Data Structure Documentation	936
3.7.7.8.2.1	struct os_het_control_t	936
3.7.7.8.2.1.1	Field Documentation	937
3.7.7.8.2.1.2	start_validation_value	937
3.7.7.8.2.1.3	initialized	937
3.7.7.8.2.1.4	pa_shared_mem	937
3.7.7.8.2.1.5	sc_shared_mem	937
3.7.7.8.2.1.6	ipc	937
3.7.7.8.2.1.7	l1_defense	937
3.7.7.8.2.1.8	smartdsp_debug	937
3.7.7.8.2.1.9	het_debug_print	937
3.7.7.8.2.1.10	shared_ctrl_size	937
3.7.7.8.2.1.11	num_ipc_regions	937
3.7.7.8.2.1.12	end_validation_value	937
3.7.7.8.3	Macro Definition Documentation	938
3.7.7.8.3.1	OS_HET_IPC_HW_SEMAPHORE_NUM	938
3.7.7.8.3.2	OS_HET_BOOT_HW_SEMAPHORE_NUM	938
3.7.7.8.4	Variable Documentation	938
3.7.7.8.4.1	g_os_het_control	938
3.7.7.9	B486x Heterogeneous Common API	938
3.7.7.9.1	Overview	938

Section number	Title	Page
3.7.7.9.2	Data Structure Documentation	939
3.7.7.9.2.1	struct os_het_init_t	939
3.7.7.9.2.1.1	Field Documentation	939
3.7.7.9.2.1.2	pa_initialized	939
3.7.7.9.2.1.3	sc_initialized	939
3.7.7.9.2.2	struct os_het_tracker_t	939
3.7.7.9.2.2.1	Field Documentation	939
3.7.7.9.2.2.2	producer_num	939
3.7.7.9.2.2.3	consumer_num	939
3.7.7.9.3	Macro Definition Documentation	940
3.7.7.9.3.1	FALSE	940
3.7.7.9.3.2	TRUE	940
3.7.7.9.3.3	OS_HET_INITIALIZED	940
3.7.7.9.3.4	OS_HET_INITIALIZED_MULTIMODE	940
3.7.7.9.3.5	OS_HET_UNINITIALIZED	940
3.7.7.9.3.6	OS_HET_PA_SEMAPHORE_VAL	940
3.7.7.9.3.7	OS_HET_SC_SEMAPHORE_VAL	940
3.7.7.9.3.8	OS_HET_FREE_SEMAPHORE_VAL	940
3.7.7.9.3.9	OS_HET_CALCULATE_ADDR	940
3.7.7.10	B486x Heterogeneous Debug API	940
3.7.7.10.1	Overview	940
3.7.7.10.2	Data Structure Documentation	941
3.7.7.10.2.1	struct os_het_smartdsp_log_t	941
3.7.7.10.2.1.1	Field Documentation	941
3.7.7.10.2.1.2	start_validation_value	941
3.7.7.10.2.1.3	base_address	941
3.7.7.10.2.1.4	size	941
3.7.7.10.2.1.5	last_error	941
3.7.7.10.2.1.6	end_validation_value	941
3.7.7.11	B486x Heterogeneous Debug print API	941
3.7.7.11.1	Overview	941
3.7.7.11.2	Data Structure Documentation	942
3.7.7.11.2.1	struct os_het_debug_print_sc_t	942
3.7.7.11.2.1.1	Field Documentation	942
3.7.7.11.2.1.2	buffer_location	942
3.7.7.11.2.1.3	segment_size	942
3.7.7.11.2.1.4	num_of_segments	942
3.7.7.11.2.1.5	tracker	942
3.7.7.11.2.1.6	segment_clock	942
3.7.7.11.2.1.7	overflow	942
3.7.7.11.3	Macro Definition Documentation	942
3.7.7.11.3.1	MAX_NUM_OF_SEGMENT	942
3.7.7.12	B486x Heterogeneous IPC API	943
3.7.7.12.1	Overview	943
3.7.7.12.2	Data Structure Documentation	943

Section number	Title	Page
3.7.7.12.2.1	struct os_het_ipc_bd_t	943
3.7.7.12.2.1.1	Field Documentation	943
3.7.7.12.2.1.2	msg_ptr	943
3.7.7.12.2.1.3	msg_len	943
3.7.7.12.2.2	struct os_het_ipc_channel_t	944
3.7.7.12.2.2.1	Field Documentation	944
3.7.7.12.2.2.2	start_validation_value	944
3.7.7.12.2.2.3	producer_initialized	944
3.7.7.12.2.2.4	consumer_initialized	944
3.7.7.12.2.2.5	id	944
3.7.7.12.2.2.6	tracker	944
3.7.7.12.2.2.7	bd_ring_size	945
3.7.7.12.2.2.8	max_msg_size	945
3.7.7.12.2.2.9	ch_type	945
3.7.7.12.2.2.10	bd_base	945
3.7.7.12.2.2.11	ipc_ind	945
3.7.7.12.2.2.12	ind_offset	945
3.7.7.12.2.2.13	ind_value	945
3.7.7.12.2.2.14	pa_reserved	945
3.7.7.12.2.2.15	semaphore_pointer	945
3.7.7.12.2.2.16	end_validation_value	945
3.7.7.12.2.3	struct os_het_ipc_t	945
3.7.7.12.2.3.1	Field Documentation	946
3.7.7.12.2.3.2	start_validation_value	946
3.7.7.12.2.3.3	num_ipc_channels	946
3.7.7.12.2.3.4	ipc_max_bd_size	946
3.7.7.12.2.3.5	ipc_channels	946
3.7.7.12.2.3.6	end_validation_value	946
3.7.7.12.3	Macro Definition Documentation	946
3.7.7.12.3.1	OS_HET_CH_EMPTY	946
3.7.7.12.3.2	OS_HET_CH_FREE_BDS	946
3.7.7.12.3.3	OS_HET_CH_FULL	947
3.7.7.12.3.4	OS_HET_INCREMENT_INDEX	947
3.7.7.12.3.5	OS_HET_UNSPECIFIED_LEN	947
3.7.7.12.4	Enumeration Type Documentation	947
3.7.7.12.4.1	os_het_ipc_ind_t	947
3.7.7.12.4.2	os_het_ipc_ch_types_t	947
3.7.7.13	B486x L1 defence heterogeneous API	947
3.7.7.13.1	Overview	947
3.7.7.13.2	Data Structure Documentation	948
3.7.7.13.2.1	struct os_het_l1_defense_t	948
3.7.7.13.2.1.1	Field Documentation	948
3.7.7.13.2.1.2	start_validation_value	948
3.7.7.13.2.1.3	warm_reset_mode	948
3.7.7.13.2.1.4	reset_maple	948

Section number	Title	Page
3.7.7.13.2.1.5	reset_status	948
3.7.7.13.2.1.6	end_validation_value	948
3.7.7.13.3	Enumeration Type Documentation	949
3.7.7.13.3.1	os_het_l1d_mode_t	949
3.7.7.13.3.2	os_het_l1d_reset_maple_t	949
3.7.7.13.3.3	os_het_l1d_status	949
3.7.7.14	B486x Heterogeneous Memory Descriptor structure API	949
3.7.7.14.1	Overview	949
3.7.7.14.2	Data Structure Documentation	950
3.7.7.14.2.1	struct os_het_mem_t	950
3.7.7.14.2.1.1	Field Documentation	950
3.7.7.14.2.1.2	start_addr	950
3.7.7.14.2.1.3	size	950
3.7.7.15	B486x IPC (Inter Process Communication) API	950
3.7.7.15.1	Overview	950
3.7.7.15.2	Data Structure Documentation	951
3.7.7.15.2.1	struct os_ipc_channel_consumer_open_params_t	951
3.7.7.15.2.1.1	Field Documentation	951
3.7.7.15.2.1.2	ch	951
3.7.7.15.2.1.3	channel_type	951
3.7.7.15.2.1.4	indication_type	951
3.7.7.15.2.1.5	int_priority	951
3.7.7.15.2.1.6	int_argument	951
3.7.7.15.2.1.7	bd_ring_size	951
3.7.7.15.2.1.8	max_msg_size	952
3.7.7.15.2.1.9	callback	952
3.7.7.15.2.1.10	buffers_pool	952
3.7.7.15.2.1.11	cache_policy	952
3.7.7.15.2.2	struct os_ipc_channel_producer_open_params_t	952
3.7.7.15.2.2.1	Field Documentation	952
3.7.7.15.2.2.2	ch	952
3.7.7.15.2.2.3	channel_type	952
3.7.7.15.2.2.4	additional_params	952
3.7.7.15.2.2.5	cache_policy	952
3.7.7.15.2.3	struct os_ipc_dsp_channel_t	953
3.7.7.15.2.3.1	Field Documentation	953
3.7.7.15.2.3.2	id	953
3.7.7.15.2.3.3	bd_ring_size	953
3.7.7.15.2.3.4	bd_base	953
3.7.7.15.2.4	struct os_ipc_dsp_t	953
3.7.7.15.2.4.1	Field Documentation	953
3.7.7.15.2.4.2	num_ipc_channels	953
3.7.7.15.2.4.3	ipc_max_bd_size	953
3.7.7.15.2.4.4	ipc_dsp_channels	954
3.7.7.15.3	Function Documentation	954

Section number	Title	Page
3.7.7.15.3.1	osIpcMessagePtrGet	954
3.7.7.15.3.2	osIpcMessageSendPtr	954
3.7.7.15.3.3	osIpcMessageReceiveCb	955
3.7.7.15.3.4	osIpcMpicMsiMessageReceiveCb	955
3.7.7.15.3.5	osIpcChannelPeek	956
3.7.7.15.3.6	osIpcMessageChannelBufferReplace	956
3.7.7.15.3.7	osIpcMessageChannelBufferRelease	956
3.7.7.15.3.8	osIpcIndicationSend	957
3.7.7.15.3.9	osIpcConsumerIndexGet	957
3.7.7.15.3.10	osIpcChannelNumGet	958
3.7.7.15.3.11	osIpcInit	958
3.7.7.15.3.12	osIpcMultimodeChannelIdFind	959
3.7.7.15.3.13	osIpcChannelIdFind	960
3.7.7.15.3.14	osIpcDspChannelIdFind	960
3.7.7.15.3.15	osIpcChannelConsumerOpen	961
3.7.7.15.3.16	osIpcChannelProducerOpen	961
3.7.7.15.3.17	osIpcMpicMsiCoalescingSet	962
3.7.7.15.3.18	osIpcChannelInitializedVerify	962
3.7.7.16	B4860 SmartDsp OS Devices	962
3.7.7.16.1	Overview	962
3.7.7.16.2	Macro Definition Documentation	964
3.7.7.16.2.1	OS_MEM_RESERVED	964
3.7.7.16.2.2	g_dsp_ccsr_map	965
3.7.7.16.2.3	INTERRUPTS_PER_SEC_1000	965
3.7.7.16.2.4	INTERRUPTS_PER_SEC_100	965
3.7.7.16.2.5	SOC_TICK_001MS	965
3.7.7.16.2.6	SOC_TICK_010MS	965
3.7.7.16.2.7	SOC_TICK_DEFAULT	965
3.7.7.16.2.8	OS_SOC_DSP_NUM_CLUSTERS	965
3.7.7.16.2.9	OS_SOC_DSP_CORES_IN_CLUSTER	965
3.7.7.16.2.10	OS_SOC_MAX_NUM_OF_CORES	965
3.7.7.16.2.11	OS_SOC_DSP_LOWEST_CORENUM	965
3.7.7.16.2.12	OS_SOC_PA_NUM_CLUSTERS	965
3.7.7.16.2.13	OS_SOC_PA_CORES_IN_CLUSER	965
3.7.7.16.2.14	OS_SOC_PA_THREADS_PER_CORE	966
3.7.7.16.2.15	OS_SOC_MAX_NUM_OF_PA_CORES	966
3.7.7.16.2.16	OS_SOC_MAX_NUM_OF_PA_THREADS	966
3.7.7.16.2.17	OS_SOC_PA_LOWEST_CORENUM	966
3.7.7.16.2.18	OS_SOC_MAX_NUM_OF_MESSAGES	966
3.7.7.16.2.19	CORE0_ID	966
3.7.7.16.2.20	CORE1_ID	966
3.7.7.16.2.21	CORE2_ID	966
3.7.7.16.2.22	CORE3_ID	966
3.7.7.16.2.23	CORE4_ID	966
3.7.7.16.2.24	CORE5_ID	966

Section number	Title	Page
3.7.7.16.2.25	ANY_CORE_ID	966
3.7.7.16.2.26	EXTERNAL_MASTER_ID	967
3.7.7.16.2.27	CLUSTER0_ID	967
3.7.7.16.2.28	CLUSTER1_ID	967
3.7.7.16.2.29	CLUSTER2_ID	967
3.7.7.16.2.30	ANY_CLUSTER_ID	967
3.7.7.16.2.31	DSP_CLUSTER_REGS_DEFAULT	967
3.7.7.16.2.32	DSP_CCSR_REGS_DEFAULT	967
3.7.7.16.2.33	SOC_DSP_MAPLE_DEFAULT	967
3.7.7.16.2.34	SOC_DSP_MAPLE_MBUS_DEFAULT	967
3.7.7.16.2.35	NUM_OF_MEMMAP_CPRI_UNITS	967
3.7.7.16.2.36	MAX_NUM_OF_VIRTS	967
3.7.7.16.2.37	NUMBER_OF_MMU_PROG_MATT	967
3.7.7.16.2.38	NUMBER_OF_MMU_DATA_MATT	968
3.7.7.16.2.39	SUPPORTS_MMU_FSM	968
3.7.7.16.2.40	SUPPORTS_MMU_NEXT_LINE_PREFETCH	968
3.7.7.16.2.41	NUMBER_OF_CME_DATA_CHANNELS	968
3.7.7.16.2.42	NUMBER_OF_CME_PROG_CHANNELS	968
3.7.7.16.2.43	PRIVATE_L2_CACHE	968
3.7.7.16.2.44	NUMBER_L2_CACHE_PARTITIONS	968
3.7.7.16.2.45	NUMBER_OF_WDTS	968
3.7.7.16.2.46	MAPLE_IN_PLATFORM	968
3.7.7.16.2.47	NUM_OF_MAPLES	968
3.7.7.16.2.48	NUM_OF_HW_TIMER_32b_GROUPS	968
3.7.7.16.2.49	NUM_OF_HW_TIMERS_32b_PER_MODULE	968
3.7.7.16.2.50	NUM_OF_HW_TIMERS_32b_MODULES_PER_GROUP	969
3.7.7.16.2.51	NUM_OF_HW_TIMERS_32b	969
3.7.7.16.2.52	NUM_OF_HW_TIMER_32b_MODULES	969
3.7.7.16.2.53	SC39XX_NUM_OF_TIMERS	969
3.7.7.16.2.54	NUMBER_OF_HW_SEMAPHORES	969
3.7.7.16.2.55	NUMBER_OF_OCN_DMA	969
3.7.7.16.2.56	NUMBER_OF_OCN_DMA_CH	969
3.7.7.16.2.57	NUMBER_OF_OCN_DMA_ATMU_WIN	969
3.7.7.16.2.58	NUMBER_OF_HSSI_PHY_PORTS	969
3.7.7.16.2.59	MAX_NUM_SRIO_PORTS	969
3.7.7.16.2.60	NUMBER_OF_SRIO_PHY_PORTS	969
3.7.7.16.2.61	NUMBER_OF_RIO_ATMU_IN_WIN	970
3.7.7.16.2.62	NUMBER_OF_RIO_ATMU_OUT_WIN	970
3.7.7.16.2.63	SRIO_SUPPORTS_ACCEPT_ALL	970
3.7.7.16.2.64	MAPLE_DIRECT_ACCESSES	970
3.7.7.16.2.65	MAX_NUM_MAPLE	970
3.7.7.16.2.66	MAX_NUM_OF_MAPLE_PE	970
3.7.7.16.2.67	MAX_NUM_OF_CHANNELS_PER_PE	970
3.7.7.16.2.68	CLEAR_LEVEL_INTERRUPT	970
3.7.7.17	SC39XX Definitions	971

Section number	Title	Page
3.7.7.17.1	Overview	971
3.7.7.17.2	Macro Definition Documentation	972
3.7.7.17.2.1	SR_PRIORITIES_NUMBER	972
3.7.7.17.2.2	SR_PRIORITY_SHIFT	972
3.7.7.17.2.3	SR_PRIORITY_MASK	972
3.7.7.17.2.4	OS_HWI_PRIORITY0	972
3.7.7.17.2.5	OS_HWI_PRIORITY1	972
3.7.7.17.2.6	OS_HWI_PRIORITY2	972
3.7.7.17.2.7	OS_HWI_PRIORITY3	973
3.7.7.17.2.8	OS_HWI_PRIORITY4	973
3.7.7.17.2.9	OS_HWI_PRIORITY5	973
3.7.7.17.2.10	OS_HWI_PRIORITY6	973
3.7.7.17.2.11	OS_HWI_PRIORITY7	973
3.7.7.17.2.12	OS_HWI_PRIORITY8	973
3.7.7.17.2.13	OS_HWI_PRIORITY9	973
3.7.7.17.2.14	OS_HWI_PRIORITY10	973
3.7.7.17.2.15	OS_HWI_PRIORITY11	973
3.7.7.17.2.16	OS_HWI_PRIORITY12	973
3.7.7.17.2.17	OS_HWI_PRIORITY13	973
3.7.7.17.2.18	OS_HWI_PRIORITY14	973
3.7.7.17.2.19	OS_HWI_PRIORITY15	973
3.7.7.17.2.20	OS_HWI_PRIORITY16	974
3.7.7.17.2.21	OS_HWI_PRIORITY17	974
3.7.7.17.2.22	OS_HWI_PRIORITY18	974
3.7.7.17.2.23	OS_HWI_PRIORITY19	974
3.7.7.17.2.24	OS_HWI_PRIORITY20	974
3.7.7.17.2.25	OS_HWI_PRIORITY21	974
3.7.7.17.2.26	OS_HWI_PRIORITY22	974
3.7.7.17.2.27	OS_HWI_PRIORITY23	974
3.7.7.17.2.28	OS_HWI_PRIORITY24	974
3.7.7.17.2.29	OS_HWI_PRIORITY25	974
3.7.7.17.2.30	OS_HWI_PRIORITY26	974
3.7.7.17.2.31	OS_HWI_PRIORITY27	974
3.7.7.17.2.32	OS_HWI_PRIORITY28	974
3.7.7.17.2.33	OS_HWI_PRIORITY29	975
3.7.7.17.2.34	OS_HWI_PRIORITY30	975
3.7.7.17.2.35	OS_HWI_LAST_PRIORITY	975
3.7.7.17.2.36	NUMBER_OF_CORE_INTERRUPTS	975
3.7.7.17.2.37	ARCH_CONTEXT_SIZE	975
3.7.7.17.2.38	OS_HWI_PRIORITY_ALL	975
3.7.7.17.2.39	OS_HWI_PRIORITY_NMI	975
3.7.7.17.2.40	ARCH_CACHE_LINE_SIZE	975
3.7.7.17.2.41	DSP_CORE_NUM_DID	975
3.7.7.17.2.42	DSP_CORE_NUM_PID	975
3.7.7.17.2.43	OS_SYSTEM_DID	975

Section number	Title	Page
3.7.7.17.2.44	OS_SYSTEM_PID	975
3.7.7.17.2.45	OS_SHARED_DID	975
3.7.7.17.2.46	OS_SHARED_PID	976
3.7.7.17.2.47	ARCH_STACK_MAX_SIZE	976
3.7.7.17.2.48	ARCH_SYS_STACK_VIRT_BASE	976
3.7.7.17.2.49	ARCH_APP_STACK_VIRT_BASE	976
3.7.7.17.2.50	TASK_STATUS_REG	976
3.7.7.17.2.51	TASK_STATUS2_REG	976
3.7.7.17.2.52	STACK_DEFAULT	976
3.7.7.17.2.53	TASK_PC_REG_PLACE	976
3.7.7.17.2.54	TASK_STATUS_REG_PLACE	976
3.7.7.17.2.55	TASK_STATUS2_REG_PLACE	976
3.7.7.17.2.56	TASK_ARG_REG_PLACE	976
3.7.7.17.2.57	TASK_ARG_PROCID_REG_PLACE	976
3.7.7.17.3	Enumeration Type Documentation	977
3.7.7.17.3.1	anonymous enum	977
3.7.7.17.3.2	register_offset_in_stack	977
3.7.7.17.4	Tasks Runtime API.	977
3.7.7.17.4.1	Overview	977
3.7.7.17.4.2	Data Structure Documentation	978
3.7.7.17.4.2.1	struct os_task_init_param_t	978
3.7.7.17.4.2.2	Field Documentation	979
3.7.7.17.4.2.3	task_function	979
3.7.7.17.4.2.4	task_arg	979
3.7.7.17.4.2.5	top_of_stack	979
3.7.7.17.4.2.6	stack_size	979
3.7.7.17.4.2.7	task_flags	979
3.7.7.17.4.2.8	task_priority	979
3.7.7.17.4.2.9	task_name	979
3.7.7.17.4.2.10	private_data	979
3.7.7.17.4.2.11	init_stack	979
3.7.7.17.4.3	Macro Definition Documentation	980
3.7.7.17.4.3.1	TASK_NAME_LEN	980
3.7.7.17.4.3.2	TASK_IDLE_ID	980
3.7.7.17.4.3.3	TASK_NULL_ID	980
3.7.7.17.4.3.4	OS_TASK_PRIORITY_HIGHEST	980
3.7.7.17.4.3.5	OS_TASK_PRIORITY_01	980
3.7.7.17.4.3.6	OS_TASK_PRIORITY_02	980
3.7.7.17.4.3.7	OS_TASK_PRIORITY_03	980
3.7.7.17.4.3.8	OS_TASK_PRIORITY_04	980
3.7.7.17.4.3.9	OS_TASK_PRIORITY_05	980
3.7.7.17.4.3.10	OS_TASK_PRIORITY_06	980
3.7.7.17.4.3.11	OS_TASK_PRIORITY_07	980
3.7.7.17.4.3.12	OS_TASK_PRIORITY_08	980
3.7.7.17.4.3.13	OS_TASK_PRIORITY_09	981

Section number	Title	Page
3.7.7.17.4.3.14	OS_TASK_PRIORITY_10	981
3.7.7.17.4.3.15	OS_TASK_PRIORITY_11	981
3.7.7.17.4.3.16	OS_TASK_PRIORITY_12	981
3.7.7.17.4.3.17	OS_TASK_PRIORITY_13	981
3.7.7.17.4.3.18	OS_TASK_PRIORITY_14	981
3.7.7.17.4.3.19	OS_TASK_PRIORITY_15	981
3.7.7.17.4.3.20	OS_TASK_PRIORITY_16	981
3.7.7.17.4.3.21	OS_TASK_PRIORITY_17	981
3.7.7.17.4.3.22	OS_TASK_PRIORITY_18	981
3.7.7.17.4.3.23	OS_TASK_PRIORITY_19	981
3.7.7.17.4.3.24	OS_TASK_PRIORITY_20	981
3.7.7.17.4.3.25	OS_TASK_PRIORITY_21	981
3.7.7.17.4.3.26	OS_TASK_PRIORITY_22	982
3.7.7.17.4.3.27	OS_TASK_PRIORITY_23	982
3.7.7.17.4.3.28	OS_TASK_PRIORITY_24	982
3.7.7.17.4.3.29	OS_TASK_PRIORITY_25	982
3.7.7.17.4.3.30	OS_TASK_PRIORITY_26	982
3.7.7.17.4.3.31	OS_TASK_PRIORITY_27	982
3.7.7.17.4.3.32	OS_TASK_PRIORITY_28	982
3.7.7.17.4.3.33	OS_TASK_PRIORITY_29	982
3.7.7.17.4.3.34	OS_TASK_PRIORITY_30	982
3.7.7.17.4.3.35	OS_TASK_PRIORITY_LOWEST	982
3.7.7.17.4.3.36	OS_NUM_OF_PRIORITIES	982
3.7.7.17.4.3.37	OS_TASK_UNUSED	982
3.7.7.17.4.3.38	OS_TASK_ACQUIRED	983
3.7.7.17.4.3.39	OS_TASK_SUSPEND	983
3.7.7.17.4.3.40	OS_TASK_PEND	983
3.7.7.17.4.3.41	OS_TASK_TIMEOUT	983
3.7.7.17.4.3.42	OS_TASK_DELAY	983
3.7.7.17.4.3.43	OS_TASK_READY	983
3.7.7.17.4.3.44	OS_TASK_RUNNING	983
3.7.7.17.4.3.45	OS_TASK_EXPIRED	983
3.7.7.17.4.4	Function Documentation	983
3.7.7.17.4.4.1	osTaskFind	983
3.7.7.17.4.4.2	osTaskCreate	984
3.7.7.17.4.4.3	osTaskActivate	984
3.7.7.17.4.4.4	osTaskSuspend	984
3.7.7.17.4.4.5	osTaskDelete	985
3.7.7.17.4.4.6	osTaskDelay	986
3.7.7.17.4.4.7	osTaskPrioritySet	986
3.7.7.17.4.4.8	osTaskSchedulerLock	987
3.7.7.17.4.4.9	osTaskSchedulerUnlock	987
3.7.7.17.4.4.10	osTaskSelf	987
3.7.7.17.4.4.11	osTaskStatusGet	987
3.7.7.17.4.4.12	osTaskPrivateDataGet	988

Section number	Title	Page
3.7.7.17.4.4.13	osTaskPrivateDataSet	988
3.7.7.17.4.4.14	osTaskPriorityGet	988
3.7.7.17.4.4.15	osTaskPriorityReadyCount	989
3.7.7.17.4.4.16	osTaskYield	989
3.7.7.17.4.4.17	osTaskNameGet	990
3.7.8	Hardware Timers Initialization API	990
3.7.8.1	Overview	990
3.7.8.2	Data Structure Documentation	991
3.7.8.2.1	struct hw_timers32_data_t	991
3.7.8.2.1.1	Field Documentation	991
3.7.8.2.1.1.1	timer32_core_id	991
3.7.8.2.2	struct timer32_trigger_params_t	991
3.7.8.2.2.1	Field Documentation	991
3.7.8.2.2.1.1	trigger_type	991
3.7.8.2.2.1.2	trigger_input	991
3.7.8.2.3	struct timer32_group_init_params_t	991
3.7.8.2.3.1	Field Documentation	992
3.7.8.2.3.1.1	trigger_params	992
3.7.8.2.3.1.2	clock_in	992
3.7.8.2.3.1.3	group_num	992
3.7.8.3	Enumeration Type Documentation	992
3.7.8.3.1	timer32_input_select_t	992
3.7.8.4	Function Documentation	992
3.7.8.4.1	hwTimer32Initialize	992
3.7.8.4.2	hwTimer32GroupInitialize	992

Chapter 1
Introduction

The SmartDSP Operating System (OS) is a Real Time Operating System (RTOS) that runs on the following StarCore DSP-based processors:

- B4860/B4420

The StarCore CodeWarrior (CW) Development Studio includes SmartDSP OS royalty-free source code. Further, SmartDSP OS has a high-level Application Programming Interface (API) that lets users develop integrated applications for StarCore processors.

This Reference Manual documents SmartDSP OS API. For more information on the features of SmartDSP OS, refer to SmartDSP OS User Guide.

Accompanying Documentation

The Documentation page describes the documentation included in this version of CodeWarrior Development Studio for StarCore DSP Architectures. You access the Documentation page by:

- a shortcut link on the Desktop that the CodeWarrior installer creates by default, or
- opening START_HERE.html in `CWInstallDir`.

Chapter 2
Kernel

2.1 Overview

Modules

- SmartDSP OS Initialization
- HAL-Hardware Abstraction Layers
- System Memory Management
- Interrupts
- Timers
- Utilities
- Queues API
- Frames API
- Clock API
- Multi Core and Multi Device Management
- Multitasking
- Architectures
- MPIC Module API

2.2 SmartDSP OS Initialization

2.2.1 Overview

```
These functions are called via osInitialize()
```

Modules

- B4860 Initialization API

Data Structures

- struct os_mem_heap_t

Functions

- os_status osInitializeKernel (uint32_t system_clock, uint8_t num_of_cores, uint8_t max_num_of↩ _cores, uint8_t id_of_master_core)
- os_status osActivate (os_background_task_function background_task)
- os_status osArchInitialize ()
- os_status osArchDevicesInitialize ()
- os_status osMemInitialize (os_mem_heap_t os_mem_heap_local[ ], int num_of_local_heaps, os_↩ mem_heap_t os_mem_heap_shared[ ], int num_of_shared_heaps)

- os_status osMemLocalInitialize (uint16_t total_mem_parts, uint8_t *local_mngmnt_space, uint32_t local_mngmnt_size)
- os_status osMemSharedInitialize (uint8_t *shared_mngmnt_space, uint32_t shared_mngmnt_size)
- os_status osHwiInitialize (void *int_vec_add)
- os_status osTickInitialize (uint32_t tick_parameter)
- os_status osTickStart (os_swi_priority priority)
- os_status osSwiInitialize (uint16_t total_swi)
- os_status osTimerInitialize (uint16_t total_timers)
- os_status osHwTimerInitialize ()
- os_status osMessageInitialize (uint16_t total_messages)
- os_status osMultiCoreSyncInitialize ()
- os_status sysInit ()
- os_status osQueueInitialize (uint16_t total_queues, uint16_t total_squeues)
- os_status osFrameInitialize ()
- os_status osCioInitialize (uint16_t total_cio_devices)
- os_status osBioInitialize (uint16_t total_bio_devices)
- os_status osSioInitialize (uint16_t total_sio_devices)
- os_status osCopInitialize (uint16_t total_cop_devices)
- os_status osDmaInitialize (arch_specific *dma_config)
- os_status osHwWatchdogsInit (void)
- os_status osEventsInitialize (int max_event_queue, int max_event_semaphore)
- os_status osTasksInitialize (int max_task)
- os_status osLogInitialize (uint8_t *base_addr, uint32_t stack_size, uint32_t num_cores)

2.2.2 Data Structure Documentation

2.2.2.1 struct os_mem_heap_t

Heap data structure.

2.2.3 Function Documentation

2.2.3.1 os_status osInitializeKernel (uint32_t *system_clock*, uint8_t *num_of_cores*, uint8_t *max_num_of_cores*, uint8_t *id_of_master_core*)

Initializes the OS kernel.

This function initializes various parameters of the OS, using the given configuration parameters. In addition, it aligns the stack as necessary.

Parameters

system_clock	- The system clock in MHz.

num_of_cores	- Number of active cores (multicore only).
max_num_of_cores	- Maximum number of cores (multicore only).
id_of_master_core	- ID of the master core (multicore only).

Returns

OS_SUCCESS

Warning

Must be called before osStart().

2.2.3.2 os_status osActivate (os_background_task_function *background_task*)

Starts the OS operation.

This function receives a pointer to a background task that is ready to run and switches into that task. If the background task runs infinitely, osActivate() never returns.

Parameters

background_task	- The background task to run.

Return values

OS_FAIL	- The background task has returned;

Warning

osInitialize() must be called before osActivate(); Don't use temporary variables here, since this function does not return, and its stack will not be cleared.

2.2.3.3 os_status osArchInitialize ( )

Architecture-specific general initialization.

This function should be implemented per architecture.

Returns

OS_SUCCESS on success; OS_FAIL otherwise.

2.2.3.4 os_status osArchDevicesInitialize ( )

Architecture-specific devices initialization.

This function should be implemented per architecture.

Returns

OS_SUCCESS on success; OS_FAIL otherwise.

2.2.3.5 os_status osMemInitialize (os_mem_heap_t *os_mem_heap_local[],* int *num_of_local_heaps,* os_mem_heap_t *os_mem_heap_shared[],* int *num_of_shared_heaps*)

Initializes the memory management module.

Parameters

os_mem_heap_local	- Pointer to list of local heaps.
num_of_local_heaps	- Number of local heaps.
os_mem_heap_shared	- Pointer to list of shared heaps (multicore only).
num_of_shared_heaps	- Number of shared heaps (multicore only).

Return values

OS_SUCCESS	- Memory management module was initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory; operation failed.

2.2.3.6 os_status osMemLocalInitialize (uint16_t *total_mem_parts,* uint8_t * *local_mngmnt_space,* uint32_t *local_mngmnt_size*)

Initializes the local memory management module.

Parameters

total_mem_↩ parts	- Number of memory partitions in the system.
local_↩ mngmnt_space	- Pointer to preallocated management memory space for local heaps.
local_↩ mngmnt_size	- Amount of memory space preallocated for local memory internal management (Bytes).

Return values

OS_SUCCESS	- Memory management module was initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory; operation failed.

2.2.3.7 os_status osMemSharedInitialize (uint8_t * *shared_mngmnt_space,* uint32_t *shared_mngmnt_size*)

Initializes the shared memory management module.

Parameters

shared_↩ mngmnt_space	- Pointer to preallocated management memory space for shared heaps.
shared_↩ mngmnt_size	- Amount of memory space preallocated for shared memory internal management (Bytes).

Return values

OS_SUCCESS	- Memory management module was initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory; operation failed.

2.2.3.8 os_status osHwiInitialize (void * *int_vec_add*)

Initializes the hardware interrupts module.

Parameters

int_vec_add	- Base address of interrupts vector.

Returns

OS_SUCCESS on success; OS_FAIL otherwise.

Warning

Must be called before calling osStart() and before creating a hardware interrupt.

2.2.3.9 os_status osTickInitialize (uint32_t *tick_parameter*)

Initializes the OS Tick module.

Parameters

tick_parameter	- Tick value (architecture-specific), must be greater than zero.

Returns

OS_SUCCESS

Warning

The tick value parameter MUST be greater than zero !

2.2.3.10 os_status osTickStart (os_swi_priority *priority*)

Starts the OS tick functionality.

Parameters

priority	- OS Tick priority

Returns

OS_SUCCESS on success of osTickSetup(); OS_FAIL otherwise.

2.2.3.11 os_status osSwiInitialize (uint16_t *total_swi*)

Initializes the software interrupts module.

Parameters

total_swi	- Number of software interrupts in the system.

Return values

OS_SUCCESS	- Software interrupts module was initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory; operation failed.

2.2.3.12 os_status osTimerInitialize (uint16_t *total_timers*)

Initializes the software timers module.

Parameters

total_timers	- Number of software timers in the system.

Return values

OS_SUCCESS	- Software timers module was initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory; operation failed.

2.2.3.13 os_status osHwTimerInitialize ( )

Initializes the hardware timers module.

Returns

OS_SUCCESS

2.2.3.14 os_status osMessageInitialize (uint16_t *total_messages*)

Initializes the intercore messages module.

Parameters

total_messages	- Number of intercore messages in the system.

Return values

OS_SUCCESS	- The intercore messages module was initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory; operation failed.
OS_ERR_MSG_NUM_LIMIT	- Requested number of messages is too high (unsupported); operation failed.

2.2.3.15 os_status osMultiCoreSyncInitialize ( )

Initializes the multicore synchronization module.

Return values

OS_SUCCESS	- The multicore synchronization module was initialized successfully.
OS_ERR_NO_MEMORY	- For master core only: not enough memory; operation failed. other cores will enter a busy wait state until the master core is successful.

2.2.3.16 os_status sysInit ( )

Initializes the multicore synchronization module.

Return values

OS_SUCCESS	- The multicore synchronization module was initialized successfully.
OS_ERR_NO_MEMORY	- For master core only: not enough memory; operation failed. other cores will enter a busy wait state until the master core is successful.

2.2.3.17 os_status osQueueInitialize (uint16_t *total_queues,* uint16_t *total_squeues*)

Initializes the OS Queues.

Parameters

total_queues	- Number of queues in the system.

total_squeues	- Number of shared queues in the system (multicore only).

Return values

OS_SUCCESS	- Queues were initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory; operation failed.

2.2.3.18 os_status osFrameInitialize ( )

Initializes the OS Frames.

Returns

OS_SUCCESS

2.2.3.19 os_status osCioInitialize (uint16_t *total_cio_devices*)

Initializes the CIO layer.

Parameters

total_cio_devices	- Number of CIO devices in the system.

Return values

OS_SUCCESS	- CIO was initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory; operation failed.

2.2.3.20 os_status osBioInitialize (uint16_t *total_bio_devices*)

Initializes the BIO layer.

Parameters

total_bio_devices	- Number of BIO devices in the system.

Return values

OS_SUCCESS	- BIO was initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory; operation failed.

2.2.3.21 os_status osSioInitialize (uint16_t *total_sio_devices*)

Initializes the SIO layer.

Parameters

total_sio_devices	- Number of SIO devices in the system.

Return values

OS_SUCCESS	- SIO was initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory; operation failed.

2.2.3.22 os_status osCopInitialize (uint16_t *total_cop_devices*)

Initializes the COP layer.

Parameters

total_cop_devices	- Number of COP devices in the system.

Return values

OS_SUCCESS	- COP was initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory; operation failed.

2.2.3.23 os_status osDmaInitialize (arch_specific * *dma_config*)

Initializes the DMA controller.

Parameters

dma_config	- Architecture-specific DMA configuration structure.

Returns

OS_SUCCESS on success; OS_FAIL otherwise.

2.2.3.24 os_status osHwWatchdogsInit (void)

Return values

OS_SUCCESS	- the watchdog SWI and timer were initialized successfully.
OS_FAIL	- the OS failed to initialize the watchdog SWI and timer.

2.2.3.25 os_status osEventsInitialize (int *max_event_queue,* int *max_event_semaphore*)

Stores max number of event types. Currently only queues and counting semaphores are supported.

Parameters

max_event_↩ queue	- Maximal number of event queues.
max_event_↩ semaphore	- Maximal number of event semaphores.

Returns

OS_SUCCESS - Events initializized.
OS_FAIL - Events not initializized.

2.2.3.26 os_status osTasksInitialize (int *max_task*)

Allocates a scheduler and then allocate tasks and events objects and insert them in it. Background function is defined in smartdsp_init.c, before osInitialize. It can be chosen to be wait command or idle count or user defined. In osStart, background task is created and activated.

Parameters

max_task	- Maximal number of tasks.

Returns

OS_SUCCESS - Tasks initializized.
OS_FAIL - Tasks not initializized.

2.2.3.27 os_status osLogInitialize (uint8_t * *base_addr,* uint32_t *stack_size,* uint32_t *num_cores*)

Overrides the default stack locaiton and size for kernel awareness. Assumes contiguous address space shared between OS_NUM_OF_CORES. If no multicore support, all stack size goes to single core. Will configure virtual and physical address to the same value (in platforms where applicable).

Parameters

base_addr	- Base address of shared stack size.
stack_size	- Size of shared stack size.
num_cores	- Number of cores (if have multicore support)

Returns

OS_SUCCESS - Kernel awareness stack set.
OS_FAIL - Kernel awareness stack not set.

2.2.4 B4860 Initialization API

2.2.4.1 Overview

```
API for initializing the B4860 kernel
```

Data Structures

- union platform_init_params_t

Functions

- void b4860GetCoreInformation ()
- os_status b4860Initialize (float clock_in, os_virt_ptr dsp_cluster_base, platform_init_params_t *platform_params, l2_init_params_t *l2_init_params, cpc_init_params_t *cpc_params)

- void b4860HetBspStubInitialize ()
- void b4860HetBootInitialize ()
- os_status b4860HetInitialize (os_virt_ptr soc_het_ctrl, uint32_t soc_het_size, os_virt_ptr pa_het_b, os_virt_ptr sc_het_b)
- os_status b4420HssiLanesPowerDown ()

2.2.4.2 Data Structure Documentation

2.2.4.2.1 union platform_init_params_t

Platform configuration parameters.

Data Fields

- uint32_t m_cr
- uint32_t __pad0__:20
- uint32_t ge:1
- uint32_t ice:1
- uint32_t dce:1
- uint32_t __pad1__:1
- uint32_t soee:1
- uint32_t vccc:1
- uint32_t vcee:1
- uint32_t __pad2__:1
- uint32_t mpe:1
- uint32_t ate:1
- uint32_t eccee:1
- uint32_t __pad3__:1

2.2.4.2.1.1 Field Documentation

2.2.4.2.1.1.1 uint32_t platform_init_params_t::__pad0__

Reserved by hardware.

2.2.4.2.1.1.2 uint32_t platform_init_params_t::ge

Gather Enable - TRUE/FALSE enables/disables SGB gathering function; turning it off has performance implications.

2.2.4.2.1.1.3 uint32_t platform_init_params_t::ice

Instruction Cache Enable - When TRUE, Instruction Cache is enabled.

Instruction accesses cache policy determined by relevant MMU descriptor. When FALSE, Instruction Cache is disabled overriding ICache and L2 cache policy to non-cacheable for all instruction accesses

2.2.4.2.1.1.4 uint32_t platform_init_params_t::dce

Data Cache Enable - When TRUE, Data Cache is enabled.

Data accesses cache policy determined by relevant MMU descriptor. When FALSE, Data Cache is disabled overriding DCache and L2 cache policy to non-cacheable for all data accesses

2.2.4.2.1.1.5 uint32_t platform_init_params_t::__pad1__

Reserved by hardware.

2.2.4.2.1.1.6 uint32_t platform_init_params_t::soee

Stack Overrun Error Enabled - TRUE/FALSE enables/disables precise exceptions as a result of stack related accesses that match non-stack descriptors.

2.2.4.2.1.1.7 uint32_t platform_init_params_t::vccc

Voluntary Cache Commands Cancel -.

- When TRUE, the MMU and CME cancel the following cache performance commands: DMALL↩ OC, DFETCHx
- When false, cache performance commands are enabled

2.2.4.2.1.1.8 uint32_t platform_init_params_t::vcee

Voluntary Cache Commands Error Enabled - When TRUE,.

- generates precise exception on error on granular DFETCHx commands, unless VCCC bit is enabled: DFLUSH, DSYNC, DINVALIDATE, DTUNLOCK, DMALLOC, DFETCHx When F↩ ALSE, the MMU closes the access inside the platform, but does not inform the core of the error

2.2.4.2.1.1.9 uint32_t platform_init_params_t::__pad2__

Reserved by hardware.

2.2.4.2.1.1.10 uint32_t platform_init_params_t::mpe

Memory Protection Enable - TRUE/FALSE enables/disables the protection-checking function in all enabled segment descriptors.

It also enables/disables the miss interrupt support on a miss access

2.2.4.2.1.1.11 uint32_t platform_init_params_t::ate

Address Translation Enable - TRUE/FALSE enables/disables the address translation mechanism.

2.2.4.2.1.1.12 uint32_t platform_init_params_t::eccee

Error Detection Code Exception Enable - TRUE/FALSE enables/disables the the ECC exception.

2.2.4.2.1.1.13 uint32_t platform_init_params_t::__pad3__

Reserved by hardware.

2.2.4.2.1.1.14 uint32_t platform_init_params_t::m_cr

all attributes above

2.2.4.3 Function Documentation

2.2.4.3.1 void b4860GetCoreInformation ( )

Initializes core information in global register. This functions should generally be called by osInitialize. For cases where core information is needed prior to osInitialize, application may use this function.

Returns

None

Warning

None

2.2.4.3.2 os_status b4860Initialize (float *clock_in*, os_virt_ptr *dsp_cluster_base*, platform_init_params_t * *platform_params*, l2_init_params_t * *l2_init_params*, cpc_init_params_t * *cpc_params*)

Initializes B4860 general architecture parts.

Parameters

in	*clock_in*	- B4860 CLKIN frequency (MHz).
in	*dsp_cluster_base*	- B4860 DSP cluster base address.
in	*platform_params*	- Platform initialization parameters
in	*l2_init_params*	- L2 cache initialization parameters
in	*cpc_params*	- L3 cache/SRAM initialization parameters

Returns

OS status

Warning

None

2.2.4.3.3 void b4860HetBspStubInitialize ( )

This function mimics what the PA BSP should do in a "real" system

It writes to shared control space values as defined at ctrl_params or sets some default values if ctrl_params == NULL

2.2.4.3.4 void b4860HetBootInitialize ( )

Finalizes heterogeneous boot using HW semaphores;

SC take semaphore and waits till PA releases it

Warning

None

2.2.4.3.5 os_status b4860HetInitialize (os_virt_ptr *soc_het_ctrl,* uint32_t *soc_het_size,* os_virt_ptr *pa_het_b,* os_virt_ptr *sc_het_b*)

Initialized PA and SC shared space;

Parameters

in	*soc_het_ctrl*	- virtual pointer to os_het_control_t shared space
in	*soc_het_size*	- size of shared control space
in	*pa_het_b*	- virtual base PA shared cacheable space
in	*sc_het_b*	- virtual base SC shared cacheable space

Returns

OS status: OS_SUCCESS or OS_FAIL

Warning

None

2.2.4.3.6 os_status b4420HssiLanesPowerDown ( )

Power off B4420 not supported HSSI SERDES lanes.

Returns

OS success

2.3 HAL-Hardware Abstraction Layers

2.3.1 Overview

Modules

- BIO-Buffered I/O Module API
- COP-Co-Processor Module API
- CIO-Character I/O Module API
- SIO-Synchronized I/O Module API

2.3.2 BIO-Buffered I/O Module API

2.3.2.1 Overview

```
Buffer I/O routines, defines, enumerations and API
```

Modules

- BIO Initialization API
- BIO Runtime API

Data Structures

- struct bio_dev_open_params_t
- struct bio_ch_open_params_t
- struct bio_channel_t
- struct bio_interface_stat_t

Typedefs

- typedef void * bio_dev_handle

BIO Channel Modes

When opening a channel, the application has to specify whether it is BIO_READ or BIO_WRITE.

In addition, a channel may be opened as BIO_ACTIVE or else be activated later.

- #define BIO_READ 0x1
- #define BIO_WRITE 0x2
- #define BIO_ACTIVE 0x4

BIO Control Commands

Used in osBioDeviceCtrl() and osBioChannelCtrl()

- #define BIO_LLD_COMMAND 0x01000000
- #define BIO_COMMAND_WRITE 0x00000000
- #define BIO_COMMAND_READ 0x10000000
- #define BIO_CHANNEL_TX_ENABLE (0x00000001 | BIO_LLD_COMMAND | BIO_COMM↩ AND_WRITE)
- #define BIO_CHANNEL_RX_ENABLE (0x00000002 | BIO_LLD_COMMAND | BIO_COMM↩ AND_READ)
- #define BIO_CHANNEL_TX_DISABLE (0x00000003 | BIO_LLD_COMMAND | BIO_COM↩ MAND_WRITE)
- #define BIO_CHANNEL_RX_DISABLE (0x00000004 | BIO_LLD_COMMAND | BIO_COM↩ MAND_READ)
- #define BIO_CHANNEL_STAT_GET (0x0000000A | BIO_LLD_COMMAND | BIO_COMMA↩ ND_READ)
- #define BIO_CHANNEL_RX_ADD_ID (0x0000000B | BIO_LLD_COMMAND | BIO_COMM↩ AND_READ)
- #define BIO_CHANNEL_RX_REMOVE_ID (0x0000000C | BIO_LLD_COMMAND | BIO_C↩ OMMAND_READ)
- #define BIO_CHANNEL_RX_PHYS_LLD_ADDR_ADD (0x0000000D | BIO_LLD_COMMA↩ ND | BIO_COMMAND_READ)
- #define BIO_CHANNEL_RX_PHYS_LLD_ADDR_REMOVE (0x0000000E | BIO_LLD_COM↩ MAND | BIO_COMMAND_READ)
- #define BIO_DEVICE_TX_ENABLE (0x00000001 | BIO_LLD_COMMAND)
- #define BIO_DEVICE_RX_ENABLE (0x00000002 | BIO_LLD_COMMAND)
- #define BIO_DEVICE_TX_DISABLE (0x00000003 | BIO_LLD_COMMAND)
- #define BIO_DEVICE_RX_DISABLE (0x00000004 | BIO_LLD_COMMAND)
- #define BIO_DEVICE_RESET (0x00000008 | BIO_LLD_COMMAND)
- #define BIO_DEVICE_SUPPORT_MULTIBUFFER (0x00000009 | BIO_LLD_COMMAND)
- #define BIO_DEVICE_STAT_GET (0x00000009)
- #define BIO_REMOVE_BUFFER 0xFFFF

2.3.2.2 Data Structure Documentation

2.3.2.2.1 struct bio_dev_open_params_t

BIO device configuration parameters.

This structure is passed to osBioDeviceOpen() and defines some of the device parameters that the BIO requires. The structure also contains a pointer to LLD-specific parameters. This pointer is passed to the LLD as it is.

Data Fields

- os_mem_part_t ∗ common_pool
- void ∗ lld_params

2.3.2.2.1.1 Field Documentation

2.3.2.2.1.1.1 os_mem_part_t∗ bio_dev_open_params_t::common_pool

Pointer to initialized common buffers pool.

2.3.2.2.1.1.2 void∗ bio_dev_open_params_t::lld_params

LLD-specific parameters.

2.3.2.2.2 struct bio_ch_open_params_t

BIO channel configuration parameters.

This structure is passed to osBioChannelOpen() and defines some of the channel parameters that the BIO requires. The structure also contains a pointer to LLD-specific parameters. This pointer is passed to the LLD as it is.

Data Fields

- uint16_t channel_num
- void(∗ callback)(void ∗param, uint32_t data, uint32_t error_status)
- void ∗ cb_parameter
- os_frames_pool_t ∗ frames_pool
- os_mem_part_t ∗ buffers_pool
- void ∗ lld_params

2.3.2.2.2.1 Field Documentation

2.3.2.2.2.1.1 uint16_t bio_ch_open_params_t::channel_num

Channel number.

2.3.2.2.2.1.2 void(∗ bio_ch_open_params_t::callback)(void ∗param, uint32_t data, uint32_t error_status)

Application's callback for this channel; for Tx, the data is the confirmed frame; for Rx, data is always 0.

2.3.2.2.2.1.3 void* bio_ch_open_params_t::cb_parameter

Application's parameter for the callback.

2.3.2.2.2.1.4 os_frames_pool_t* bio_ch_open_params_t::frames_pool

Frames pool for the channel.

2.3.2.2.2.1.5 os_mem_part_t* bio_ch_open_params_t::buffers_pool

Rx buffers pool.

2.3.2.2.2.1.6 void* bio_ch_open_params_t::lld_params

LLD-specific channel parameters.

2.3.2.2.3 struct bio_channel_t

BIO channel internal structure.

This structure is exposed to the application layer to let the application allocate memory for the channels in a convenient way. The application should not change any field in this structure.

Warning

> This structure is declared publicly to enable allocating enough memory for it by the application. Users are prohibited from directly accessing the structure.

Data Fields

- uint32_t status
- void(* callback)(void *param, uint32_t data, uint32_t error_status)
- void * cb_parameter
- void * lld_ch
- void * bio_interf
- uint8_t channel_type
- os_queue_handle frames_queue
- os_frames_pool_t * frames_pool
- os_mem_part_t * buffers_pool
- os_queue_handle buffers_queue
- void(* f_RxFrameBuild)(void *bio_ch_handle, os_frame_t *frame, uint32_t frame_length)

2.3.2.2.3.1 Field Documentation

2.3.2.2.3.1.1 uint32_t bio_channel_t::status

The channel's status.

2.3.2.2.3.1.2 void(* bio_channel_t::callback)(void *param, uint32_t data, uint32_t error_status)

Application's callback for this channel; for Tx, the data is the confirmed frame; for Rx, data is always 0.

2.3.2.2.3.1.3 void* bio_channel_t::cb_parameter

Application's parameter for the callback.

2.3.2.2.3.1.4 void* bio_channel_t::lld_ch

The matching LLD channel handle.

2.3.2.2.3.1.5 void* bio_channel_t::bio_interf

The channel's BIO device interface.

2.3.2.2.3.1.6 uint8_t bio_channel_t::channel_type

The channel's type (read or write).

2.3.2.2.3.1.7 os_queue_handle bio_channel_t::frames_queue

The channel's frames queue handle.

2.3.2.2.3.1.8 os_frames_pool_t* bio_channel_t::frames_pool

The channel's frames pool.

2.3.2.2.3.1.9 os_mem_part_t* bio_channel_t::buffers_pool

The channel's buffers pool (Rx channel only).

2.3.2.2.3.1.10 os_queue_handle bio_channel_t::buffers_queue

The channel's buffers queue handle (Rx channel only).

2.3.2.2.3.1.11 void(* bio_channel_t::f_RxFrameBuild)(void *bio_ch_handle, os_frame_t *frame, uint32_t frame_length)

Frame build function (Rx channels only, for single-buffer or multi-buffer frames).

2.3.2.2.4 struct bio_interface_stat_t

BIO interface statistics.

Data Fields

- uint32_t rx_frames
- uint32_t rx_bytes
- uint32_t tx_frames
- uint32_t tx_bytes
- uint32_t tx_errors
- uint32_t rx_errors
- uint32_t interface_speed

- bool is_promisc

2.3.2.2.4.1 Field Documentation

2.3.2.2.4.1.1 uint32_t bio_interface_stat_t::rx_frames

Num of RX frames.

2.3.2.2.4.1.2 uint32_t bio_interface_stat_t::rx_bytes

Num of RX bytes.

2.3.2.2.4.1.3 uint32_t bio_interface_stat_t::tx_frames

Num of TX frames.

2.3.2.2.4.1.4 uint32_t bio_interface_stat_t::tx_bytes

Num of TX bytes.

2.3.2.2.4.1.5 uint32_t bio_interface_stat_t::tx_errors

Num of TX errors.

2.3.2.2.4.1.6 uint32_t bio_interface_stat_t::rx_errors

Num of RX errors.

2.3.2.2.4.1.7 uint32_t bio_interface_stat_t::interface_speed

Speed.

2.3.2.2.4.1.8 bool bio_interface_stat_t::is_promisc

IF is promiscuous.

2.3.2.3 Macro Definition Documentation

2.3.2.3.1 #define BIO_READ 0x1

Channel is for input.

2.3.2.3.2 #define BIO_WRITE 0x2

Channel is for output.

2.3.2.3.3 #define BIO_ACTIVE 0x4

Channel is active.

2.3.2.3.4 #define BIO_LLD_COMMAND 0x01000000

Defines a command that will be handled by the Low Level Driver.

2.3.2.3.5 #define BIO_COMMAND_WRITE 0x00000000

Defines a command for an output channel.

2.3.2.3.6 #define BIO_COMMAND_READ 0x10000000

Defines a command for an input channel.

2.3.2.3.7 #define BIO_CHANNEL_TX_ENABLE (0x00000001 | BIO_LLD_COMMAND | BIO_COMMAND_WRITE)

Enable an output channel.

2.3.2.3.8 #define BIO_CHANNEL_RX_ENABLE (0x00000002 | BIO_LLD_COMMAND | BIO_COMMAND_READ)

Enable an input channel.

2.3.2.3.9 #define BIO_CHANNEL_TX_DISABLE (0x00000003 | BIO_LLD_COMMAND | BIO_COMMAND_WRITE)

Disable an output channel.

2.3.2.3.10 #define BIO_CHANNEL_RX_DISABLE (0x00000004 | BIO_LLD_COMMAND | BIO_COMMAND_READ)

Disable an input channel.

2.3.2.3.11 #define BIO_CHANNEL_STAT_GET (0x0000000A | BIO_LLD_COMMAND | BIO_COMMAND_READ)

Return the statistics on a channel.

2.3.2.3.12 #define BIO_CHANNEL_RX_ADD_ID (0x0000000B | BIO_LLD_COMMAND | BIO_COMMAND_READ)

Add a LLD ID to the channel.

2.3.2.3.13 #define BIO_CHANNEL_RX_REMOVE_ID (0x0000000C | BIO_LLD_COMMAND | BIO_COMMAND_READ)

Remove a LLD ID to the channel.

2.3.2.3.14 #define BIO_CHANNEL_RX_PHYS_LLD_ADDR_ADD (0x0000000D | BIO_LLD_COMMAND | BIO_COMMAND_READ)

Add a LLD physical address to the channel.

2.3.2.3.15 #define BIO_CHANNEL_RX_PHYS_LLD_ADDR_REMOVE (0x0000000E | BIO_LLD_COMMAND | BIO_COMMAND_READ)

Remove a LLD physical address to the channel.

2.3.2.3.16 #define BIO_DEVICE_TX_ENABLE (0x00000001 | BIO_LLD_COMMAND)

Enable an output device.

2.3.2.3.17 #define BIO_DEVICE_RX_ENABLE (0x00000002 | BIO_LLD_COMMAND)

Enable an input device.

2.3.2.3.18 #define BIO_DEVICE_TX_DISABLE (0x00000003 | BIO_LLD_COMMAND)

Disable an output device.

2.3.2.3.19 #define BIO_DEVICE_RX_DISABLE (0x00000004 | BIO_LLD_COMMAND)

Disable an input device.

2.3.2.3.20 #define BIO_DEVICE_RESET (0x00000008 | BIO_LLD_COMMAND)

Reset the device.

2.3.2.3.21 #define BIO_DEVICE_SUPPORT_MULTIBUFFER (0x00000009 | BIO_LLD_COMMAND)

Check if device supports multi-buffered-frames.

2.3.2.3.22 #define BIO_DEVICE_STAT_GET (0x00000009)

Return the statistics on a device.

2.3.2.3.23 #define BIO_REMOVE_BUFFER 0xFFFF

Free device's buffer.

2.3.2.4 Typedef Documentation

2.3.2.4.1 typedef void* bio_dev_handle

BIO device handle - passed as parameter to all device functions.

2.3.2.5 BIO Initialization API

2.3.2.5.1 Overview

```
BIO upper layer (serializer) API for the
           application initialization stage
```

Functions

- bio_dev_handle osBioDeviceOpen (char *device_name, bio_dev_open_params_t *dev_open_↩ params)
- os_status osBioChannelOpen (bio_dev_handle bio_handle, bio_channel_t *bio_ch, int mode, bio↩ _ch_open_params_t *ch_open_params)
- os_status osBioChannelClose (bio_channel_t *bio_ch)

2.3.2.5.2 Function Documentation

2.3.2.5.2.1 bio_dev_handle osBioDeviceOpen (char * *device_name,* bio_dev_open_params_t * *dev_open_params*)

Opens a BIO device for operation using the given parameters.

Parameters

in	*device_name*	- The name of the device.
in	*dev_open_↩ params*	- Device parameters for use by both BIO and LLD, or NULL to use the default device parameters.

Returns

A BIO device handle if the device was opened successfully
NULL if the device failed to open

2.3.2.5.2.2 os_status osBioChannelOpen (bio_dev_handle *bio_handle,* bio_channel_t * *bio_ch,* int *mode,* bio_ch_open_params_t * *ch_open_params*)

Opens a channel of a BIO device using the given parameters.

Parameters

in	*bio_handle*	- BIO device handle returned by osBioDeviceOpen()
out	*bio_ch*	- Pointer to a BIO channel structure.
in	*mode*	- Channel mode (BIO_READ, BIO_WRITE, BIO_ACTIVE).
in	*ch_open_↩ params*	- Channel parameters.

Return values

OS_SUCCESS	if channel was opened successfully
OS_ERR_PARAM_INV↩ ALID	if invalid parameters were provided to the function
OS_FAIL	if the underlying LLD channel failed to open properly

Returns

Error status, encoded in os_error.h, for other errors

2.3.2.5.2.3 os_status osBioChannelClose (bio_channel_t * *bio_ch*)

Closes an open channel of a BIO device.

Parameters

in	*bio_ch*	- Pointer to a valid BIO channel returned by osBioChannelOpen()

Return values

OS_SUCCESS	if channel closed properly

Returns

Error status, encoded in os_error.h, if the underlying LLD channel failed to close properly

2.3.2.6 BIO Runtime API

2.3.2.6.1 Overview

```
BIO upper layer (serializer) API for the
           application runtime stage
```

Functions

- os_status osBioChannelTx (bio_channel_t ∗bio_ch, void ∗frame)
- void ∗ osBioChannelRx (bio_channel_t ∗bio_ch)
- os_status osBioDeviceCtrl (bio_dev_handle bio_handle, uint32_t command, void ∗param)
- os_status osBioChannelCtrl (bio_channel_t ∗bio_ch, uint32_t command, void ∗param)

2.3.2.6.2 Function Documentation

2.3.2.6.2.1 os_status osBioChannelTx (bio_channel_t ∗ *bio_ch,* void ∗ *frame*)

Call this function to transmit a frame through a BIO channel that was previously opened for transmission.

Parameters

in	*bio_ch*	- Pointer to a valid BIO channel returned by osBioChannelOpen()
in	*frame*	- The frame to transmit (actually a pointer to an os_frame_t structure)

Return values

OS_SUCCESS	if frame was transmitted properly

Returns

Error status, encoded in os_error.h, if the underlying LLD channel failed to transmit the frame

2.3.2.6.2.2 void* osBioChannelRx (bio_channel_t * *bio_ch*)

This function returns a pointer to a received frame.

Parameters

in	*bio_ch*	- Pointer to a valid BIO channel returned by osBioChannelOpen()

Return values

Pointer	to a received frame (actually a pointer to an os_frame_t structure).
NULL	if no frames are available on the channel

2.3.2.6.2.3 os_status osBioDeviceCtrl (bio_dev_handle *bio_handle,* uint32_t *command,* void * *param*)

Performs control commands on a device.

Parameters

in	*bio_handle*	- BIO device handle returned by osBioDeviceOpen()
in	*command*	- Command code - see BIO Control Commands
in	*param*	- Command parameter, command specific

Return values

OS_SUCCESS	if command was executed properly
OS_ERR_COMMAND_UNSUPPORTED	if command is unsupported

Returns

Error status, encoded in os_error.h, if the underlying LLD device failed to execute the command

2.3.2.6.2.4 os_status osBioChannelCtrl (bio_channel_t * *bio_ch,* uint32_t *command,* void * *param*)

Performs control commands on a channel.

Parameters

in	*bio_ch*	- Pointer to a valid BIO channel returned by osBioChannelOpen()
in	*command*	- Command code - see BIO Control Commands
in	*param*	- Command parameter, command specific

Return values

OS_SUCCESS	if command was executed properly
OS_ERR_COMMAND_INVALID	if BIO_READ command was called on a BIO_WRITE channel, or vice versa

Returns

Error status, encoded in os_error.h, if the underlying LLD channel failed to execute the command

2.3.3 COP-Co-Processor Module API

2.3.3.1 Overview

```
Co-Processors routines, defines, enumerations and API
```

Modules

- COP Initialization API
- COP Runtime API

Data Structures

- struct cop_job_handle
- struct cop_dev_open_params_t
- struct cop_ch_open_params_t
- struct cop_interface_stat_t

Macros

- #define MAX_NUM_OF_CHANNELS 16

Typedefs

- typedef void * cop_dev_handle
- typedef void * cop_job_id
- typedef void(* os_cop_callback)(void *job_handle, void *param)
- typedef void(* os_cop_error)(void *param)
- typedef void * cop_channel_t

COP Channel Modes

```
When opening a channel, the application has to
specify whether it is #COP_CH_REAP or
#COP_CH_DISPATCH or both.
```

Channels start as automatically being enabled

- #define COP_CH_REAP 0x1
- #define COP_CH_DISPATCH 0x2
- #define COP_CH_SHARED_DELETED 0x4

COP Control Commands

```
Used in osCopDeviceCtrl() and osCopChannelCtrl()
```

- #define COP_LLD_COMMAND 0x01000000
- #define COP_NON_BLOCKING_COMMAND 0x02000000
- #define COP_CHANNEL_DISABLE (0x00000001 | COP_LLD_COMMAND)
- #define COP_CHANNEL_ENABLE (0x00000002 | COP_LLD_COMMAND)
- #define COP_CHANNEL_STAT_GET (0x0000000F | COP_LLD_COMMAND)
- #define COP_CHANNEL_POLL (0x0000000E | COP_LLD_COMMAND)
- #define COP_DEVICE_RESET (0x00000001 | COP_LLD_COMMAND)
- #define COP_DEVICE_STAT_GET (0x00000002)
- #define COP_DEVICE_DISPACHER_SET (0x00000003)
- #define COP_DEVICE_ISR_SET (0x00000004 | COP_LLD_COMMAND)
- #define COP_CHANNEL_CALLBACK_SET (0x00000005)
- #define COP_CHANNEL_CALLBACK_REMOVE (0x00000006)

2.3.3.2 Data Structure Documentation

2.3.3.2.1 struct cop_job_handle

COP job descriptor.

The application creates a NULL terminated linked list of such descriptors and calls osCopChannel↩ Dispatch() to send them to be executed.

Warning

> The linked list of jobs must be created on the heap. Placing in on the stack may cause memory corruption

Data Fields

- cop_job_id job_id
- void * device_specific
- struct cop_job_handle_s * next

2.3.3.2.1.1 Field Documentation

2.3.3.2.1.1.1 cop_job_id cop_job_handle::job_id

Used by the application to identify finished jobs.

2.3.3.2.1.1.2 void* cop_job_handle::device_specific

LLD specific job parameters.

2.3.3.2.1.1.3 struct cop_job_handle_s* cop_job_handle::next

Pointer to the next job or NULL to indicate the last job.

2.3.3.2.2 struct cop_dev_open_params_t

COP device configuration parameters.

This structure is passed to osCopDeviceOpen() and defines some of the device parameters that the COP requires. The structure also contains a pointer to LLD-specific parameters. This pointer is passed to the LLD as it is.

Data Fields

- os_cop_callback dispatch_callback
- os_cop_error error_callback
- void * lld_params
- uint8_t max_num_of_channels

2.3.3.2.2.1 Field Documentation

2.3.3.2.2.1.1 os_cop_callback cop_dev_open_params_t::dispatch_callback

Application's dispatch callback function by the COP module.

2.3.3.2.2.1.2 os_cop_error cop_dev_open_params_t::error_callback

Application's callback for errors.

2.3.3.2.2.1.3 void* cop_dev_open_params_t::lld_params

LLD parameters for device open.

2.3.3.2.2.1.4 uint8_t cop_dev_open_params_t::max_num_of_channels

Max number of channels allowed on device.

2.3.3.2.3 struct cop_ch_open_params_t

COP channel configuration parameters.

This structure is passed to osCopChannelOpen() and defines some of the channel parameters that the COP requires. The structure also contains a pointer to LLD-specific parameters. This pointer is passed to the LLD as it is.

Data Fields

- uint16_t channel_num
- uint16_t num_jobs
- void * callback_parameter
- void * error_callback_parameter
- void * lld_params
- os_mem_type heap

2.3.3.2.3.1 Field Documentation

2.3.3.2.3.1.1 uint16_t cop_ch_open_params_t::channel_num

Channel number.

2.3.3.2.3.1.2 uint16_t cop_ch_open_params_t::num_jobs

Number of jobs channel can handle.

2.3.3.2.3.1.3 void* cop_ch_open_params_t::callback_parameter

Application's parameter for the data callbacks from this channel.

2.3.3.2.3.1.4 void* cop_ch_open_params_t::error_callback_parameter

Application's parameter for the error callbacks from this channel.

2.3.3.2.3.1.5 void* cop_ch_open_params_t::lld_params

LLD-specific channel parameters.

2.3.3.2.3.1.6 os_mem_type cop_ch_open_params_t::heap

Heap from which to allocate the channel structure.

2.3.3.2.4 struct cop_interface_stat_t

COP interface statistics.

Data Fields

- uint32_t dispatched_jobs
- uint32_t reaped_jobs
- uint32_t failed_jobs

2.3.3.2.4.1 Field Documentation

2.3.3.2.4.1.1 uint32_t cop_interface_stat_t::dispatched_jobs

Number of dispatched jobs.

2.3.3.2.4.1.2 uint32_t cop_interface_stat_t::reaped_jobs

Number of reaped jobs.

2.3.3.2.4.1.3 uint32_t cop_interface_stat_t::failed_jobs

Number of failed jobs.

2.3.3.3 Macro Definition Documentation

2.3.3.3.1 #define MAX_NUM_OF_CHANNELS 16

Defines a maximum number of channel allowed to be opened on a COP device.

2.3.3.3.2 #define COP_CH_REAP 0x1

The channel can reap jobs after the co-processor finishes executing them.

2.3.3.3.3 #define COP_CH_DISPATCH 0x2

The channel can send jobs for the co-processor to execute.

2.3.3.3.4 #define COP_CH_SHARED_DELETED 0x4

The channel can send jobs for the co-processor to execute.

2.3.3.3.5 #define COP_LLD_COMMAND 0x01000000

Defines a command that will be handled by the Low Level Driver.

2.3.3.3.6 #define COP_NON_BLOCKING_COMMAND 0x02000000

Defines a command that will be non-blocking.

2.3.3.3.7 #define COP_CHANNEL_DISABLE (0x00000001 | COP_LLD_COMMAND)

Disable a channel from processing more jobs.

2.3.3.3.8 #define COP_CHANNEL_ENABLE (0x00000002 | COP_LLD_COMMAND)

Enable a channel to process jobs.

2.3.3.3.9 #define COP_CHANNEL_STAT_GET (0x0000000F | COP_LLD_COMMAND)

Read the LLD statistics for the channel.

2.3.3.3.10 #define COP_CHANNEL_POLL (0x0000000E | COP_LLD_COMMAND)

Poll the LLD channel for jobs that are ready for reaping.

2.3.3.3.11 #define COP_DEVICE_RESET (0x00000001 | COP_LLD_COMMAND)

Reset the LLD device.

2.3.3.3.12 #define COP_DEVICE_STAT_GET (0x00000002)

Read the statistics for the COP device.

2.3.3.3.13 #define COP_DEVICE_DISPACHER_SET (0x00000003)

Set the function to be called by osCopChannelDispatch() in order to send jobs to the LLD.

This function may only be set once, generally by the OS

2.3.3.3.14 #define COP_DEVICE_ISR_SET (0x00000004 | COP_LLD_COMMAND)

Set the function to be called by the LLD when the co-processor indicates that a job is finished executing.

This function may only be set once, generally by the OS

2.3.3.3.15 #define COP_CHANNEL_CALLBACK_SET (0x00000005)

Set a channel specific callback fucntion, thus overriding the use of the device callback on the specified channel.

2.3.3.3.16 #define COP_CHANNEL_CALLBACK_REMOVE (0x00000006)

Remove a channel specific callback fucntion, thus restoring the use of the device callback on the specified channel.

2.3.3.4 Typedef Documentation

2.3.3.4.1 typedef void* cop_dev_handle

COP device handle - passed as parameter to all device functions.

2.3.3.4.2 typedef void* cop_job_id

COP job ID - a parameter to the job descriptor.

2.3.3.4.3 typedef void(* os_cop_callback)(void *job_handle, void *param)

COP Channel callback function typedef.

This function is called by the COP Module when the LLD indicates that a job(s) is ready for the application - i.e. the co-processor finished execution

2.3.3.4.4 typedef void(* os_cop_error)(void *param)

COP error callback function typedef.

2.3.3.4.5 typedef void* cop_channel_t

COP Channel handle - passed as parameter to all Channel's functions.

2.3.3.5 COP Initialization API

2.3.3.5.1 Overview

COP upper layer API for the application initialization stage

Functions

- cop_dev_handle osCopDeviceOpen (char *device_name, cop_dev_open_params_t *dev_open_↩ params)
- os_status osCopDeviceClose (cop_dev_handle deviceHandle)
- os_status osCopChannelOpen (cop_dev_handle cop_handle, cop_channel_t *cop_ch, cop_ch_↩ open_params_t *ch_open_params)
- os_status osCopSharedChannelOpen (cop_dev_handle cop_handle, cop_channel_t *cop_ch, cop↩ _ch_open_params_t *ch_open_params, int mode, os_queue_handle queue)
- os_status osCopChannelClose (cop_channel_t cop_ch)
- os_status osCopSharedChannelClose (cop_channel_t cop_ch)

2.3.3.5.2 Function Documentation

2.3.3.5.2.1 cop_dev_handle osCopDeviceOpen (char * *device_name,* cop_dev_open_params_t * *dev_open_params*)

Opens a COP device for operation using the given parameters.

Parameters

in	*device_name*	- The name of the device.
in	*dev_open_params*	- Device parameters for use by both COP and LLD, or NULL to use the default device parameters.

Returns

A COP device handle if the device was opened successfully
NULL if the device failed to open

2.3.3.5.2.2 os_status osCopDeviceClose (cop_dev_handle *deviceHandle*)

Closes an opened COP device.

Parameters

in	*deviceHandle*	- A COP device handle to close

Return values

OS_SUCCESS	if channel was opened successfully
OS_FAIL	if the underlying LLD channel failed to open properly

Returns

Error status, encoded in os_error.h, for other errors

2.3.3.5.2.3 os_status osCopChannelOpen (cop_dev_handle *cop_handle,* cop_channel_t * *cop_ch,* cop_ch_open_params_t * *ch_open_params*)

Opens a channel of a COP device using the given parameters.

Parameters

in	*cop_handle*	- COP device handle, osCopDeviceOpen() returns this handle.
out	*cop_ch*	- Pointer to a COP channel structure.
in	*ch_open_↩ params*	- Channel parameters.

Return values

OS_SUCCESS	if channel was opened successfully
OS_FAIL	if the underlying LLD channel failed to open properly

Returns

Error status, encoded in os_error.h, for other errors

2.3.3.5.2.4 os_status osCopSharedChannelOpen (cop_dev_handle *cop_handle*, cop_channel_t * *cop_ch*, cop_ch_open_params_t * *ch_open_params*, int *mode*, os_queue_handle *queue*)

Opens a shared channel of a COP device using the given parameters

A COP shared channel is one that can be dispatched to from one core and reaped by another (or the same one).

Parameters

in	*cop_handle*	- COP device handle, osCopDeviceOpen() returns this handle.
out	*cop_ch*	- Pointer to a COP channel structure.
in	*ch_open_↩ params*	- Channel parameters.
in	*mode*	- Channel mode (COP_CH_REAP or COP_CH_DISPATCH)
out	*queue*	- Pointer to a created queue handle; if mode isn't (COP_CH_REAP | COP_CH_DISPATCH) - must be a shared queue

Return values

OS_SUCCESS	if channel was opened successfully
OS_FAIL	if the underlying LLD channel failed to open properly

Returns

Error status, encoded in os_error.h, for other errors

2.3.3.5.2.5 os_status osCopChannelClose (cop_channel_t *cop_ch*)

Closes an open channel of a COP device.

Parameters

in	*cop_ch*	- Pointer to a valid COP channel, osCopChannelOpen() returns this pointer.

Return values

OS_SUCCESS	if channel was closed successfully

Returns

Error status, encoded in os_error.h, for other errors

2.3.3.5.2.6 os_status osCopSharedChannelClose (cop_channel_t *cop_ch*)

Closes an open channel of a COP device.

Parameters

in	*cop_ch*	- Pointer to a valid COP channel, osCopChannelOpen() returns this pointer.

Return values

OS_SUCCESS	if channel was closed successfully

Returns

Error status, encoded in os_error.h, for other errors

2.3.3.6 COP Runtime API

2.3.3.6.1 Overview

```
COP upper layer API for the  application runtime stage
```

Functions

- os_status osCopDeviceCtrl (cop_dev_handle cop_handle, uint32_t command, void *param)
- os_status osCopChannelCtrl (cop_channel_t *cop_ch, uint32_t command, void *param)
- os_status osCopChannelDispatch (cop_channel_t *cop_ch, cop_job_handle *jobs, int *num_jobs)

2.3.3.6.2 Function Documentation

2.3.3.6.2.1 os_status osCopDeviceCtrl (cop_dev_handle *cop_handle,* uint32_t *command,* void * *param*)

Performs control commands on a device.

Parameters

in	*cop_handle*	- COP device handle returned by osCopDeviceOpen()
in	*command*	- Command code - see COP Control Commands
in	*param*	- Command parameter, command specific

Return values

OS_SUCCESS	if command was executed properly
OS_ERR_COMMAND_UNSUPPORTED	if command is unsupported
OS_ERR_ALREADY_CREATED	if calling with COP_DEVICE_DISPACHER_SET and there already is a dispatcher for the device

Returns

Error status, encoded in os_error.h, if the underlying LLD device failed to execute the command

2.3.3.6.2.2 os_status osCopChannelCtrl (cop_channel_t * *cop_ch*, uint32_t *command*, void * *param*)

Performs control commands on a channel.

Parameters

in	*cop_ch*	- COP device handle returned by osCopDeviceOpen()
in	*command*	- Command code - see COP Control Commands
in	*param*	- Command parameter, command specific

Return values

OS_SUCCESS	if command was executed properly
OS_ERR_COMMAND_UNSUPPORTED	if command is unsupported

Returns

Error status, encoded in os_error.h, if the underlying LLD device failed to execute the command

2.3.3.6.2.3 os_status osCopChannelDispatch (cop_channel_t * *cop_ch*, cop_job_handle * *jobs*, int * *num_jobs*)

Dispatches a job(s) to a channel.

This channel takes a NULL terminated link list of jobs (cop_job_handle) and sends it the channel for execution

Parameters

in	*cop_ch*	- Pointer to a valid COP channel, osCopChannelOpen() returns this pointer.
in	*jobs*	- Pointer to the head of a NULL terminated link list of jobs
in	*num_jobs*	- Number of jobs in the linked list

Return values

OS_SUCCESS	if the jobs were dispatched successfully

Returns

Error status, encoded in os_error.h, if there was an error in dispatching the jobs

2.3.4 CIO-Character I/O Module API

2.3.4.1 Overview

```
Character I/O routines, defines, enumerations and API
```

Modules

- CIO Initialization API
- CIO Runtime API

Data Structures

- struct cio_ch_open_params_t
- struct cio_channel_t

Typedefs

- typedef void * cio_dev_handle

CIO Channel Modes

```
When opening a channel, the application has to
specify whether it is #CIO_READ or #CIO_WRITE.
```

In addition, a channel may be opened as CIO_ACTIVE or else be activated later.

- #define CIO_READ 0x1
- #define CIO_WRITE 0x2
- #define CIO_ACTIVE 0x4

CIO Control Commands

Used in osCioDeviceCtrl() and osCioChannelCtrl()

- #define CIO_LLD_COMMAND 0x40000000
- #define CIO_COMMAND_WRITE 0x20000000
- #define CIO_COMMAND_READ 0x10000000
- #define CIO_CHANNEL_TX_ENABLE (0x00000001 | CIO_LLD_COMMAND | CIO_COMM↩ AND_WRITE)
- #define CIO_CHANNEL_RX_ENABLE (0x00000002 | CIO_LLD_COMMAND | CIO_COMM↩ AND_READ)
- #define CIO_CHANNEL_TX_DISABLE (0x00000003 | CIO_LLD_COMMAND | CIO_COM↩ MAND_WRITE)
- #define CIO_CHANNEL_RX_DISABLE (0x00000004 | CIO_LLD_COMMAND | CIO_COM↩ MAND_READ)
- #define CIO_DEVICE_TX_ENABLE (0x00000001 | CIO_LLD_COMMAND | CIO_COMMA↩ ND_WRITE)
- #define CIO_DEVICE_RX_ENABLE (0x00000002 | CIO_LLD_COMMAND | CIO_COMMA↩ ND_READ)
- #define CIO_DEVICE_TX_DISABLE (0x00000003 | CIO_LLD_COMMAND | CIO_COMMA↩ ND_WRITE)
- #define CIO_DEVICE_RX_DISABLE (0x00000004 | CIO_LLD_COMMAND | CIO_COMMA↩ ND_READ)

2.3.4.2 Data Structure Documentation

2.3.4.2.1 struct cio_ch_open_params_t

CIO channel configuration parameters.

This structure is passed to osCioChannelOpen() and defines some of the channel parameters that the CIO requires. The structure also contains a pointer to LLD-specific parameters. This pointer is passed to the LLD as it is.

Data Fields

- uint16_t channel_num
- uint8_t * data_base
- uint16_t data_size
- void * lld_params
- void(* cb_tx_rx)(void *ch, uint16_t size)
- void * cb_parameter

2.3.4.2.1.1 Field Documentation

2.3.4.2.1.1.1 uint16_t cio_ch_open_params_t::channel_num

Channel number.

2.3.4.2.1.1.2 uint8_t* cio_ch_open_params_t::data_base

Character serializing buffer base address.

2.3.4.2.1.1.3 uint16_t cio_ch_open_params_t::data_size

Character serializing buffer size (in bytes).

2.3.4.2.1.1.4 void* cio_ch_open_params_t::lld_params

LLD-specific channel parameters.

2.3.4.2.1.1.5 void(* cio_ch_open_params_t::cb_tx_rx)(void *ch, uint16_t size)

Application callback for transmit or receive.

2.3.4.2.1.1.6 void* cio_ch_open_params_t::cb_parameter

Application callback parameter.

2.3.4.2.2 struct cio_channel_t

CIO channel internal structure.

This structure is exposed to the application layer to let the application allocate memory for the channels in a convenient way. The application should not change any field in this structure.

Warning

> This structure is declared publicly to enable allocating enough memory for it by the application. Users are prohibited from directly accessing the structure.

Data Fields

- uint32_t status
- uint8_t * data_base
- uint8_t * data_end
- uint8_t * data_current
- uint8_t * data_current_free
- uint8_t * data_next
- void * cb_parameter
- uint16_t borrow_right
- uint16_t available_size
- uint16_t max_size
- void(* cb_tx_rx)(void *ch, uint16_t size)
- void * lld_ch
- void * cio_interf
- uint8_t channel_type

2.3.4.2.2.1 Field Documentation

2.3.4.2.2.1.1 uint32_t cio_channel_t::status

The channel's status.

2.3.4.2.2.1.2 uint8_t* cio_channel_t::data_base

Data base pointer.

2.3.4.2.2.1.3 uint8_t* cio_channel_t::data_end

Data end pointer.

2.3.4.2.2.1.4 uint8_t* cio_channel_t::data_current

Pointer to data currently transmitted/received.

2.3.4.2.2.1.5 uint8_t* cio_channel_t::data_current_free

Pointer to unused (free) data.

2.3.4.2.2.1.6 uint8_t* cio_channel_t::data_next

Pointer to next byte for allocation by application (Tx) or device (Rx).

2.3.4.2.2.1.7 void* cio_channel_t::cb_parameter

Application's parameter for the callback.

2.3.4.2.2.1.8 uint16_t cio_channel_t::borrow_right

Current reserved size at buffer end.

2.3.4.2.2.1.9 uint16_t cio_channel_t::available_size

Current available data size.

2.3.4.2.2.1.10 uint16_t cio_channel_t::max_size

Maximum data size.

2.3.4.2.2.1.11 void(* cio_channel_t::cb_tx_rx)(void *ch, uint16_t size)

Application's callback for this channel.

2.3.4.2.2.1.12 void* cio_channel_t::lld_ch

The matching LLD channel handle.

2.3.4.2.2.1.13 void* cio_channel_t::cio_interf

The channel's CIO device interface.

2.3.4.2.2.1.14 uint8_t cio_channel_t::channel_type

The channel's type (read or write).

2.3.4.3 Macro Definition Documentation

2.3.4.3.1 #define CIO_READ 0x1

Channel is for input.

2.3.4.3.2 #define CIO_WRITE 0x2

Channel is for output.

2.3.4.3.3 #define CIO_ACTIVE 0x4

Channel is active.

2.3.4.3.4 #define CIO_LLD_COMMAND 0x40000000

Defines a command that will be handled by the Low Level Driver.

2.3.4.3.5 #define CIO_COMMAND_WRITE 0x20000000

Defines a command for an output channel.

2.3.4.3.6 #define CIO_COMMAND_READ 0x10000000

Defines a command for an input channel.

2.3.4.3.7 #define CIO_CHANNEL_TX_ENABLE (0x00000001 | CIO_LLD_COMMAND | CIO_COMMAND_WRITE)

Enable an output channel.

2.3.4.3.8 #define CIO_CHANNEL_RX_ENABLE (0x00000002 | CIO_LLD_COMMAND | CIO_COMMAND_READ)

Enable an input channel.

2.3.4.3.9 #define CIO_CHANNEL_TX_DISABLE (0x00000003 | CIO_LLD_COMMAND | CIO_COMMAND_WRITE)

Disable an output channel.

2.3.4.3.10 #define CIO_CHANNEL_RX_DISABLE (0x00000004 | CIO_LLD_COMMAND | CIO_COMMAND_READ)

Disable an input channel.

2.3.4.3.11 #define CIO_DEVICE_TX_ENABLE (0x00000001 | CIO_LLD_COMMAND | CIO_COMMAND_WRITE)

Enable an output device.

2.3.4.3.12 #define CIO_DEVICE_RX_ENABLE (0x00000002 | CIO_LLD_COMMAND | CIO_COMMAND_READ)

Enable an input device.

2.3.4.3.13 #define CIO_DEVICE_TX_DISABLE (0x00000003 | CIO_LLD_COMMAND | CIO_COMMAND_WRITE)

Disable an output device.

2.3.4.3.14 #define CIO_DEVICE_RX_DISABLE (0x00000004 | CIO_LLD_COMMAND | CIO_COMMAND_READ)

Disable an input device.

2.3.4.4 Typedef Documentation

2.3.4.4.1 typedef void* cio_dev_handle

CIO device handle - passed as parameter to all device functions.

2.3.4.5 CIO Initialization API

2.3.4.5.1 Overview

CIO upper layer API for the application initialization stage

Functions

- cio_dev_handle osCioDeviceOpen (char *device_name, void *lld_params)
- os_status osCioChannelOpen (cio_dev_handle cio_handle, cio_channel_t *cio_ch, int mode, cio_ch_open_params_t *ch_open_params)
- os_status osCioChannelClose (cio_channel_t *cio_ch)

2.3.4.5.2 Function Documentation

2.3.4.5.2.1 cio_dev_handle osCioDeviceOpen (char * *device_name*, void * *lld_params*)

Opens a CIO device for operation using the given parameters.

Parameters

in	*device_name*	- The name of the device.
in	*lld_params*	- Device-specific parameters or NULL to use the default device parameters.

Returns

A CIO device handle if the device was opened successfully
NULL if the device failed to open

2.3.4.5.2.2 os_status osCioChannelOpen (cio_dev_handle *cio_handle*, cio_channel_t * *cio_ch*, int *mode*, cio_ch_open_params_t * *ch_open_params*)

Opens a channel of a CIO device using the given parameters.

Parameters

in	*cio_handle*	- CIO device handle returned by osCioDeviceOpen()
out	*cio_ch*	- Pointer to a CIO channel structure.

in	*mode*	- Channel mode (CIO_READ, CIO_WRITE, CIO_ACTIVE).
in	*ch_open_↩ params*	- Channel parameters.

Return values

OS_SUCCESS	if channel was opened successfully
OS_FAIL	if channel failed to open

2.3.4.5.2.3 os_status osCioChannelClose (cio_channel_t * *cio_ch*)

Closes an open channel of a CIO device.

Parameters

in	*cio_ch*	- Pointer to a valid CIO channel returned by osCioChannelOpen()

Return values

OS_SUCCESS	if channel closed properly

Returns

Error status, encoded in os_error.h, if the underlying LLD channel failed to close properly

2.3.4.6 CIO Runtime API

2.3.4.6.1 Overview

CIO upper layer API for the application runtime stage

Functions

- uint8_t * osCioChannelBufferGet (cio_channel_t *cio_ch, uint16_t size)
- uint16_t osCioChannelTxBufferPut (cio_channel_t *cio_ch)
- uint8_t * osCioChannelRxBufferGet (cio_channel_t *cio_ch, uint16_t *size)
- void osCioChannelRxBufferFree (cio_channel_t *cio_ch, uint16_t size)
- os_status osCioDeviceCtrl (cio_dev_handle cio_handle, uint32_t command, void *param)
- os_status osCioChannelCtrl (cio_channel_t *cio_ch, uint32_t command, void *param)

2.3.4.6.2 Function Documentation

2.3.4.6.2.1 uint8_t* osCioChannelBufferGet (cio_channel_t * *cio_ch,* uint16_t *size*)

Call this function to obtain free buffer space.

You must call this function before transmitting or receiving (from the LLD).

Parameters

in	*cio_ch*	- Pointer to a valid CIO channel. osCioChannelOpen() returns this pointer.
in	*size*	- Requested size.

Returns

Pointer to the allocated buffer.

Return values

Returns	NULL if memory is not available.

2.3.4.6.2.2 uint16_t osCioChannelTxBufferPut (cio_channel_t * *cio_ch*)

Call this function to initiate transmission of the contents of a previously allocated buffer.

Parameters

in	*cio_ch*	- Pointer to a valid CIO channel. osCioChannelOpen() returns this pointer.

Returns

Number of characters which were accepted for transmit by LLD.

2.3.4.6.2.3 uint8_t* osCioChannelRxBufferGet (cio_channel_t * *cio_ch*, uint16_t * *size*)

This function returns a pointer to the received data

Parameters

in	*cio_ch*	- Pointer to a valid CIO channel. osCioChannelOpen() returns this pointer.
out	*size*	- Pointer to number of received bytes

Returns

Pointer to the received data

Return values

NULL	if memory is not available.

2.3.4.6.2.4 void osCioChannelRxBufferFree (cio_channel_t * *cio_ch*, uint16_t *size*)

Call this function when the application is done processing the received data.

Parameters

in	*cio_ch*	- Pointer to a valid CIO channel. osCioChannelOpen() returns this pointer.
in	*size*	- Number of bytes to free.

2.3.4.6.2.5 os_status osCioDeviceCtrl (cio_dev_handle *cio_handle*, uint32_t *command*, void * *param*)

Performs control commands on a device.

Parameters

in	*cio_handle*	- CIO device handle.
in	*command*	- Command code - see CIO Control Commands
in	*param*	- Command parameter.

Return values

OS_SUCCESS	if command was executed properly

Returns

Error status, encoded in os_error.h, if the underlying LLD device failed to execute the command

2.3.4.6.2.6 os_status osCioChannelCtrl (cio_channel_t * *cio_ch*, uint32_t *command*, void * *param*)

Performs control commands on a channel.

Parameters

in	*cio_ch*	- Pointer to a valid CIO channel. osCioChannelOpen() returns this pointer.
in	*command*	- Command code - see CIO Control Commands
in	*param*	- Command parameter.

Return values

OS_SUCCESS	if command was executed properly
OS_ERR_COMMAND_INVALID	if CIO_READ command was called on a CIO_WRITE channel, or vice versa

Returns

Error status, encoded in os_error.h, if the underlying LLD channel failed to execute the command

2.3.5 SIO-Synchronized I/O Module API

2.3.5.1 Overview

```
Synchronized I/O routines, defines, enumerations and API
```

Modules

- SIO Initialization API
- SIO Runtime API

Data Structures

- struct sio_dev_open_params_t
- struct sio_ch_open_params_t
- struct sio_channel_t

Typedefs

- typedef void * sio_dev_handle

SIO Channel Modes

```
When opening a channel, the application has to
specify whether it is #SIO_READ or #SIO_WRITE.
```

In addition, a channel may be opened as SIO_ACTIVE or else be activated later.

- #define SIO_READ 0x1
- #define SIO_WRITE 0x2
- #define SIO_ACTIVE 0x4

SIO Control Commands

Used in osSioDeviceCtrl() and osSioChannelCtrl()

- #define SIO_LLD_COMMAND 0
- #define SIO_COMMAND_WRITE 0x10000000
- #define SIO_COMMAND_READ 0x20000000
- #define SIO_CHANNEL_TX_ENABLE (0x00000001 | SIO_LLD_COMMAND | SIO_COMMAND_WRITE)
- #define SIO_CHANNEL_RX_ENABLE (0x00000002 | SIO_LLD_COMMAND | SIO_COMMAND_READ)
- #define SIO_CHANNEL_TX_DISABLE (0x00000003 | SIO_LLD_COMMAND | SIO_COMMAND_WRITE)
- #define SIO_CHANNEL_RX_DISABLE (0x00000004 | SIO_LLD_COMMAND | SIO_COMMAND_READ)
- #define SIO_DEVICE_TX_ENABLE (0x00000001 | SIO_LLD_COMMAND | SIO_COMMAND_WRITE)
- #define SIO_DEVICE_RX_ENABLE (0x00000002 | SIO_LLD_COMMAND | SIO_COMMAND_READ)
- #define SIO_DEVICE_RX_TX_ENABLE (SIO_DEVICE_TX_ENABLE | SIO_COMMAND_READ)
- #define SIO_DEVICE_TX_DISABLE (0x00000003 | SIO_LLD_COMMAND | SIO_COMMAND_WRITE)
- #define SIO_DEVICE_RX_DISABLE (0x00000004 | SIO_LLD_COMMAND | SIO_COMMAND_READ)
- #define SIO_DEVICE_RX_INTERRUPT_STATUS (0x00000005 | SIO_LLD_COMMAND | SIO_COMMAND_READ)

2.3.5.2 Data Structure Documentation

2.3.5.2.1 struct sio_dev_open_params_t

SIO device configuration parameters.

This structure is passed to osSioDeviceOpen() and defines some of the device parameters that the SIO requires. The structure also contains a pointer to LLD-specific parameters. This pointer is passed to the LLD as it is.

Data Fields

- void(* rx_callback)(void *param)
- void * rx_callback_parameter
- void(* tx_callback)(void *param)
- void * tx_callback_parameter
- void(* error_callback)(void *param)
- void * lld_params

2.3.5.2.1.1 Field Documentation

2.3.5.2.1.1.1 void(∗ sio_dev_open_params_t::rx_callback)(void ∗param)

Application's RX callback.

2.3.5.2.1.1.2 void∗ sio_dev_open_params_t::rx_callback_parameter

Application's RX callback parameter.

2.3.5.2.1.1.3 void(∗ sio_dev_open_params_t::tx_callback)(void ∗param)

Application's TX callback.

2.3.5.2.1.1.4 void∗ sio_dev_open_params_t::tx_callback_parameter

Application's TX callback parameter.

2.3.5.2.1.1.5 void(∗ sio_dev_open_params_t::error_callback)(void ∗param)

Currently unused.

2.3.5.2.1.1.6 void∗ sio_dev_open_params_t::lld_params

LLD parameters for device open.

2.3.5.2.2 struct sio_ch_open_params_t

SIO channel configuration parameters.

This structure is passed to osSioChannelOpen() and defines some of the channel parameters that the SIO requires. The structure also contains a pointer to LLD-specific parameters. This pointer is passed to the LLD as it is.

Data Fields

- uint16_t channel_num
- void ∗ callback_parameter
- uint16_t num_of_buffers
- uint32_t buffer_size
- uint32_t buffer_data_size
- uint16_t num_of_channel_buffers
- uint8_t ∗ channel_buffers_base
- void ∗ lld_params

2.3.5.2.2.1 Field Documentation

2.3.5.2.2.1.1 uint16_t sio_ch_open_params_t::channel_num

Channel number.

2.3.5.2.2.1.2 void* sio_ch_open_params_t::callback_parameter

Application's parameter for the data and error callbacks; If this field is NULL, the callback shall not be called for this channel.

2.3.5.2.2.1.3 uint16_t sio_ch_open_params_t::num_of_buffers

How many buffers are going to be used by the channel.

2.3.5.2.2.1.4 uint32_t sio_ch_open_params_t::buffer_size

The size of the complete channel buffer.

2.3.5.2.2.1.5 uint32_t sio_ch_open_params_t::buffer_data_size

The size of the filled channel buffer.

2.3.5.2.2.1.6 uint16_t sio_ch_open_params_t::num_of_channel_buffers

Number of buffers for the channel use at every given point in time.

2.3.5.2.2.1.7 uint8_t* sio_ch_open_params_t::channel_buffers_base

Where to place the channel data; should be of size (buffer_size * num_of_buffers).

2.3.5.2.2.1.8 void* sio_ch_open_params_t::lld_params

LLD-specific channel parameters.

2.3.5.2.3 struct sio_channel_t

SIO channel internal structure.

This structure is exposed to the application layer to let the application allocate memory for the channels in a convenient way. The application should not change any field in this structure.

Warning

> This structure is declared publicly to enable allocating enough memory for it by the application. Users are prohibited from directly accessing the structure.

Data Fields

- void * sio_interf
- void * callback_parameter
- uint8_t status
- uint8_t mode
- uint16_t num_of_buffers
- uint32_t buffer_size
- uint32_t buffer_data_size

- uint8_t * channel_buffers_base
- volatile uint16_t * first_driver_buffer
- volatile uint16_t * last_driver_buffer
- void * lld_channel
- struct sio_channel_t * next
- uint16_t application_buffer

2.3.5.2.3.1 Field Documentation

2.3.5.2.3.1.1 void* sio_channel_t::sio_interf

Application's callback for data.

2.3.5.2.3.1.2 void* sio_channel_t::callback_parameter

Application's parameter for the data and error callbacks; If this field is NULL, the callback shall not be called for this channel.

2.3.5.2.3.1.3 uint8_t sio_channel_t::status

SIO channel status.

2.3.5.2.3.1.4 uint8_t sio_channel_t::mode

Is this a TX or RX channel.

2.3.5.2.3.1.5 uint16_t sio_channel_t::num_of_buffers

How many buffers are going to be used by the chanel.

2.3.5.2.3.1.6 uint32_t sio_channel_t::buffer_size

The size of the complete channel buffer; This field is used only when common_db is NULL.

2.3.5.2.3.1.7 uint32_t sio_channel_t::buffer_data_size

The size of the filled channel buffer; This field is used only when common_db is NULL.

2.3.5.2.3.1.8 uint8_t* sio_channel_t::channel_buffers_base

Where to place the channel data; Should be of size (buffer_size * num_of_buffers).

2.3.5.2.3.1.9 volatile uint16_t* sio_channel_t::first_driver_buffer

First buffer held by LLD; if common_driver_buffer_index != NULL it is used, and this field is redundant.

2.3.5.2.3.1.10 volatile uint16_t* sio_channel_t::last_driver_buffer

Num of LLD buffers, used to calculate under run.

2.3.5.2.3.1.11 void* sio_channel_t::lld_channel

Used in close channel lld function.

2.3.5.2.3.1.12 struct sio_channel_t* sio_channel_t::next

The next channel in the list.

2.3.5.2.3.1.13 uint16_t sio_channel_t::application_buffer

The next buffer for the user.

2.3.5.3 Macro Definition Documentation

2.3.5.3.1 #define SIO_READ 0x1

Channel is for input.

2.3.5.3.2 #define SIO_WRITE 0x2

Channel is for output.

2.3.5.3.3 #define SIO_ACTIVE 0x4

Channel is active.

2.3.5.3.4 #define SIO_LLD_COMMAND 0

Defines a command that will be handled by the Low Level Driver.

2.3.5.3.5 #define SIO_COMMAND_WRITE 0x10000000

Defines a command for an output channel.

2.3.5.3.6 #define SIO_COMMAND_READ 0x20000000

Defines a command for an input channel.

2.3.5.3.7 #define SIO_CHANNEL_TX_ENABLE (0x00000001 | SIO_LLD_COMMAND | SIO_COMMAND_WRITE)

Enable an output channel.

2.3.5.3.8 #define SIO_CHANNEL_RX_ENABLE (0x00000002 | SIO_LLD_COMMAND | SIO_COMMAND_READ)

Enable an input channel.

2.3.5.3.9 #define SIO_CHANNEL_TX_DISABLE (0x00000003 | SIO_LLD_COMMAND | SIO_COMMAND_WRITE)

Disable an output channel.

2.3.5.3.10 #define SIO_CHANNEL_RX_DISABLE (0x00000004 | SIO_LLD_COMMAND | SIO_COMMAND_READ)

Disable an input channel.

2.3.5.3.11 #define SIO_DEVICE_TX_ENABLE (0x00000001 | SIO_LLD_COMMAND | SIO_COMMAND_WRITE)

Enable an output device.

2.3.5.3.12 #define SIO_DEVICE_RX_ENABLE (0x00000002 | SIO_LLD_COMMAND | SIO_COMMAND_READ)

Enable an input device.

2.3.5.3.13 #define SIO_DEVICE_RX_TX_ENABLE (SIO_DEVICE_TX_ENABLE | SIO_COMMAND_READ)

Enable an input/output device.

2.3.5.3.14 #define SIO_DEVICE_TX_DISABLE (0x00000003 | SIO_LLD_COMMAND | SIO_COMMAND_WRITE)

Disable an output device.

2.3.5.3.15 #define SIO_DEVICE_RX_DISABLE (0x00000004 | SIO_LLD_COMMAND | SIO_COMMAND_READ)

Disable an input device.

2.3.5.3.16 #define SIO_DEVICE_RX_INTERRUPT_STATUS (0x00000005 | SIO_LLD_COMMAND | SIO_COMMAND_READ)

RX interrupt status.

2.3.5.4 Typedef Documentation

2.3.5.4.1 typedef void* sio_dev_handle

SIO device handle - passed as parameter to all device functions.

2.3.5.5 SIO Initialization API

2.3.5.5.1 Overview

```
SIO upper layer API for the application initialization stage
```

Functions

- sio_dev_handle osSioDeviceOpen (char *device_name, sio_dev_open_params_t *dev_open_↩ params)
- os_status osSioChannelOpen (sio_dev_handle sio_handle, sio_channel_t *sio_ch, int mode, sio_↩ ch_open_params_t *ch_open_params)
- os_status osSioChannelClose (sio_channel_t *sio_ch)

2.3.5.5.2 Function Documentation

2.3.5.5.2.1 sio_dev_handle osSioDeviceOpen (char * *device_name*, sio_dev_open_params_t * *dev_open_params*)

Opens a SIO device for operation using the given parameters.

Parameters

in	*device_name*	- The name of the device.
in	*dev_open_↩ params*	- Device parameters for use by both SIO and LLD, or NULL to use the default device parameters.

Returns

A SIO device handle if the device was opened successfully
NULL if the device failed to open

2.3.5.5.2.2 os_status osSioChannelOpen (sio_dev_handle *sio_handle,* sio_channel_t * *sio_ch,* int *mode,* sio_ch_open_params_t * *ch_open_params*)

Opens a channel of a SIO device using the given parameters.

Parameters

in	*sio_handle*	- SIO device handle returned by osSioDeviceOpen()
out	*sio_ch*	- Pointer to a SIO channel structure.
in	*mode*	- Channel mode (SIO_READ, SIO_WRITE, SIO_ACTIVE).
in	*ch_open_↩ params*	- Channel parameters.

Return values

OS_SUCCESS	if channel was opened successfully
OS_FAIL	if the underlying LLD channel failed to open properly

2.3.5.5.2.3 os_status osSioChannelClose (sio_channel_t * *sio_ch*)

Closes an open channel of a SIO device.

Parameters

in	*sio_ch*	- Pointer to a valid SIO channel returned by osSioChannelOpen()

Return values

OS_SUCCESS	if channel closed properly

Returns

Error status, encoded in os_error.h, if the underlying LLD channel failed to close properly

2.3.5.6 SIO Runtime API

2.3.5.6.1 Overview

SIO upper layer API for the application runtime stage

Functions

- os_status osSioDeviceCtrl (sio_dev_handle sio_handle, uint32_t command, void *param)
- os_status osSioChannelCtrl (sio_channel_t *sio_ch, uint32_t command, void *param)
- uint8_t * osSioBufferGet (sio_channel_t *sio_ch, uint16_t *length)
- os_status osSioBufferPut (sio_channel_t *sio_ch)

2.3.5.6.2 Function Documentation

2.3.5.6.2.1 os_status osSioDeviceCtrl (sio_dev_handle *sio_handle,* uint32_t *command,* void * *param*)

Performs control commands on a device.

Parameters

in	*sio_handle*	- SIO device handle returned by osSioDeviceOpen()
in	*command*	- Command code - see SIO Control Commands
in	*param*	- Command parameter, command specific

Return values

OS_SUCCESS	if command was executed properly
OS_ERR_COMMAND_UNSUPPORTED	if command is unsupported

Returns

Error status, encoded in os_error.h, if the underlying LLD device failed to execute the command

2.3.5.6.2.2 os_status osSioChannelCtrl (sio_channel_t * *sio_ch,* uint32_t *command,* void * *param*)

Performs control commands on a channel.

Parameters

in	*sio_ch*	- Pointer to a valid SIO channel returned by osSioChannelOpen()
in	*command*	- Command code - see SIO Control Commands
in	*param*	- Command parameter, command specific

Return values

OS_SUCCESS	if command was executed properly
OS_ERR_COMMAND_INVALID	if SIO_READ command was called on a SIO_WRITE channel, or vice versa

Returns

Error status, encoded in os_error.h, if the underlying LLD channel failed to execute the command

2.3.5.6.2.3 uint8_t* osSioBufferGet (sio_channel_t * *sio_ch,* uint16_t * *length*)

Gets a valid buffer from SIO if exists, or NULL

Parameters

in	*sio_ch*	- Pointer to a valid SIO channel, osSioChannelOpen() returns this pointer.
out	*length*	- buffer_data_size.

Return values

pointer	to a valid buffer
NULL	if no more valid buffers exist for the channel

2.3.5.6.2.4 os_status osSioBufferPut (sio_channel_t * *sio_ch*)

Puts a buffer previously to the SIO channel

This function puts a buffer, obtained by calling osSioBufferGet(), to the SIO channel. The SIO module will put the buffers in the same order as they were given to the application

Parameters

in	*sio_ch*	- Pointer to a valid SIO channel, osSioChannelOpen() returns this pointer.

Return values

OS_SUCCESS	Allways.

2.4 System Memory Management

2.4.1 Overview

Memory allocation, alignment and buffers management.

Modules

- MMU - Memory Management Unit API
- Memory Allocation API
- Buffers Pool Management API
- Caches Control API

Data Structures

- struct os_mem_part_t

Macros

- #define ALIGN_SIZE(SIZE, ALIGNMENT) ((((uint32_t)(SIZE)) + ((uint32_t)(ALIGNMENT)) - 1) & (~(((uint32_t)(ALIGNMENT)) - 1)))
- #define ALIGN_DOWN_SIZE(SIZE, ALIGNMENT) ((uint32_t)(SIZE) & (~(((uint32_t)(ALIGNMENT)) - 1)))
- #define ALIGNABLE_SIZE(SIZE, ALIGNMENT) (((uint32_t)(SIZE)) + ((uint32_t)(ALIGNMENT)) - 1)
- #define ALIGN_ADDRESS(ADDRESS, ALIGNMENT) ((((uint32_t)(ADDRESS)) + ((uint32_t)(ALIGNMENT)) - 1) & (~(((uint32_t)(ALIGNMENT)) - 1)))
- #define ALIGN_DOWN_ADDRESS(ADDRESS, ALIGNMENT) ((uint32_t)(ADDRESS) & (~(((uint32_t)(ALIGNMENT)) - 1)))
- #define ALIGN_DOWN_PHYS_ADDRESS(ADDRESS, ALIGNMENT) ((os_phys_ptr)(ADDRESS) & (~(((os_phys_ptr)(ALIGNMENT)) - 1)))
- #define IS_ALIGNED(ADDRESS, ALIGNMENT) (!(((uint32_t)(ADDRESS)) & (((uint32_t)(ALIGNMENT)) - 1)))
- #define IS_COMPILER_ALIGNED(ADDRESS, ALIGNMENT) (((uint32_t)(ADDRESS)) % ((uint32_t)(ALIGNMENT)) == 0)
- #define MEM_PART_SIZE(NUM_OF_BUFFERS) ALIGNABLE_SIZE((sizeof(os_mem_part_t) + ((NUM_OF_BUFFERS) ∗ sizeof(os_virt_ptr))), ALIGNED_4_BYTES)
- #define MEM_PART_DATA_SIZE(NUM, BUF_SIZE, ALIGNMENT) ALIGNABLE_SIZE(((NUM)∗(ALIGN_SIZE(BUF_SIZE, ALIGNMENT))), ALIGNMENT)
- #define OS_MEM_PRAM OS_MEM_QE_PRAM
- #define OS_MEM_DDR_LOCAL OS_MEM_DDR0_LOCAL_CACHEABLE
- #define OS_MEM_DDR_SHARED OS_MEM_DDR0_SHARED_NONCACHEABLE
- #define OS_MEM_DDR_SHARED_CACHEABLE OS_MEM_DDR0_SHARED_CACHEABLE
- #define OS_MEM_DDR_LOCAL_NONCACHEABLE OS_MEM_DDR0_LOCAL_NONCACHEABLE

Enumerations

- enum os_mem_type {
  OS_MEM_LOCAL_CACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_CACHEABLE_TYPE), OS_MEM_LOCAL_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP), OS_MEM_SHARED_CACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_SHARED_TYPE | OS_MEM_CACHEABLE_TYPE), OS_MEM_SHARED_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_SHARED_TYPE), OS_MEM_QE_PRAM_SHARED_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_SHARED_TYPE | OS_MEM_QE_PRAM_TYPE), OS_MEM_MAPLE_PRAM_SHARED_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_SHARED_TYPE | OS_MEM_MAPLE_PRAM_TYPE), OS_MEM_M2_SHARED_CACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_CACHEABLE_TYPE | OS_MEM_SHARED_TYPE | OS_MEM_M2_TYPE), OS_MEM_M3_LOCAL_CACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_CACHEABLE_TYPE | OS_MEM_M3_TYPE), OS_MEM_M3_LOCAL_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_M3_TYPE), OS_MEM_M3_SHARED_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEA↩

P | OS_MEM_SHARED_TYPE | OS_MEM_M3_TYPE), OS_MEM_M3_SHARED_CACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_CACHEABLE_TYPE | OS_MEM_SHARED_TYPE | OS_MEM_M3_TYPE), OS_MEM_DDR0_LOCAL_CACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_CACHEABLE_TYPE | OS_MEM_DDR0_TYPE),
OS_MEM_DDR0_LOCAL_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_DDR0_TYPE), OS_MEM_DDR0_SHARED_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_SHARED_TYPE | OS_MEM_DDR0_TYPE), OS_MEM_DDR0_SHARED_CACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_CACHEABLE_TYPE | OS_MEM_SHARED_TYPE | OS_MEM_DDR0_TYPE),
OS_MEM_DDR1_LOCAL_CACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_CACHEABLE_TYPE | OS_MEM_DDR1_TYPE), OS_MEM_DDR1_LOCAL_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_DDR1_TYPE), OS_MEM_DDR1_SHARED_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_SHARED_TYPE | OS_MEM_DDR1_TYPE),
OS_MEM_DDR1_SHARED_CACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_CACHEABLE_TYPE | OS_MEM_SHARED_TYPE | OS_MEM_DDR1_TYPE), OS_MEM_HET_DDR1_SHARED_CACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_CACHEABLE_TYPE | OS_MEM_SHARED_TYPE | OS_MEM_DDR1_TYPE | OS_MEM_HET_TYPE), OS_MEM_HET_DDR0_SHARED_CACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_CACHEABLE_TYPE | OS_MEM_SHARED_TYPE | OS_MEM_DDR0_TYPE | OS_MEM_HET_TYPE),
OS_MEM_MAPLE0_BD_RING_SHARED_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_SHARED_TYPE | OS_MEM_MAPLE0_BD_RING_TYPE), OS_MEM_MAPLE1_BD_RING_SHARED_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_SHARED_TYPE | OS_MEM_MAPLE1_BD_RING_TYPE), OS_MEM_MAPLE2_BD_RING_SHARED_NONCACHEABLE = (OS_VALID_NUM | OS_SMARTDSP_HEAP | OS_MEM_SHARED_TYPE | OS_MEM_MAPLE2_BD_RING_TYPE) }

Memory offsets

- #define OFFSET_0_BYTES 0x0
- #define OFFSET_2_BYTES 0x2
- #define OFFSET_4_BYTES 0x4
- #define OFFSET_8_BYTES 0x8
- #define OFFSET_16_BYTES 0x10
- #define OFFSET_32_BYTES 0x20
- #define OFFSET_64_BYTES 0x40
- #define OFFSET_128_BYTES 0x80
- #define OFFSET_256_BYTES 0x100
- #define OFFSET_512_BYTES 0x200
- #define OFFSET_1024_BYTES 0x400
- #define OFFSET_2048_BYTES 0x800
- #define OFFSET_4096_BYTES 0x1000
- #define ALIGNED_1_BYTES 0x1
- #define ALIGNED_2_BYTES 0x2
- #define ALIGNED_4_BYTES 0x4

- #define ALIGNED_8_BYTES 0x8
- #define ALIGNED_16_BYTES 0x10
- #define ALIGNED_32_BYTES 0x20
- #define ALIGNED_64_BYTES 0x40
- #define ALIGNED_128_BYTES 0x80
- #define ALIGNED_256_BYTES 0x100
- #define ALIGNED_512_BYTES 0x200
- #define ALIGNED_1024_BYTES 0x400
- #define ALIGNED_2048_BYTES 0x800
- #define ALIGNED_4096_BYTES 0x1000

Memory Type Enumeration definitions

- #define OS_VALID_NUM_MASK 0xFFF00000
- #define OS_MEM_FLAGS_MASK 0x000FF000
- #define OS_MEM_TYPE_MASK 0x0000001F
- #define OS_VALID_NUM 0xEC900000
- #define OS_SMARTDSP_HEAP 0x00080000
- #define OS_MEM_CACHEABLE_TYPE 0x00040000
- #define OS_MEM_SHARED_TYPE 0x00020000
- #define OS_MEM_HET_TYPE 0x00010000
- #define OS_MEM_M1_TYPE 0x00000001
- #define OS_MEM_M2_TYPE 0x00000002
- #define OS_MEM_M3_TYPE 0x00000003
- #define OS_MEM_DDR0_TYPE 0x00000004
- #define OS_MEM_DDR1_TYPE 0x00000005
- #define OS_MEM_QE_PRAM_TYPE 0x00000006
- #define OS_MEM_MAPLE_PRAM_TYPE 0x00000007
- #define OS_MEM_MAPLE0_BD_RING_TYPE (OS_MEM_MAPLE_PRAM_TYPE | 0x00000010)
- #define OS_MEM_MAPLE1_BD_RING_TYPE (OS_MEM_MAPLE_PRAM_TYPE | 0x00000020)
- #define OS_MEM_MAPLE2_BD_RING_TYPE (OS_MEM_MAPLE_PRAM_TYPE | 0x00000030)

2.4.2 Data Structure Documentation

2.4.2.1 struct os_mem_part_t

Memory Management Allocation Structure.

Data Fields

- os_spinlock_handle guard
- volatile uint8_t ** array_addr
- volatile int32_t curr_index
- volatile uint32_t block_size
- volatile uint16_t shared
- volatile uint16_t buffer_offset

2.4.2.1.1 Field Documentation

2.4.2.1.1.1 os_spinlock_handle os_mem_part_t::guard

Multicore synchronization element.

2.4.2.1.1.2 volatile uint8_t** os_mem_part_t::array_addr

Memory blocks array.

2.4.2.1.1.3 volatile int32_t os_mem_part_t::curr_index

Memory block index to get or free.

2.4.2.1.1.4 volatile uint32_t os_mem_part_t::block_size

Size of each block (in bytes).

2.4.2.1.1.5 volatile uint16_t os_mem_part_t::shared

Shared between cores.

2.4.2.1.1.6 volatile uint16_t os_mem_part_t::buffer_offset

buffer space reserved for LLD

2.4.3 Macro Definition Documentation

2.4.3.1 #define OFFSET_0_BYTES 0x0

0-bytes offset

2.4.3.2 #define OFFSET_2_BYTES 0x2

2-bytes offset

2.4.3.3 #define OFFSET_4_BYTES 0x4

4-bytes offset

2.4.3.4 #define OFFSET_8_BYTES 0x8

8-bytes offset

2.4.3.5 #define OFFSET_16_BYTES 0x10

16-bytes offset

2.4.3.6 #define OFFSET_32_BYTES 0x20

32-bytes offset

2.4.3.7 #define OFFSET_64_BYTES 0x40

64-bytes offset

2.4.3.8 #define OFFSET_128_BYTES 0x80

128-bytes offset

2.4.3.9 #define OFFSET_256_BYTES 0x100

256-bytes offset

2.4.3.10 #define OFFSET_512_BYTES 0x200

512-bytes offset

2.4.3.11 #define OFFSET_1024_BYTES 0x400

1024-bytes offset

2.4.3.12 #define OFFSET_2048_BYTES 0x800

2048-bytes offset

2.4.3.13 #define OFFSET_4096_BYTES 0x1000

4096-bytes offset

2.4.3.14 #define ALIGNED_1_BYTES 0x1

1-bytes alignment

2.4.3.15 #define ALIGNED_2_BYTES 0x2

2-bytes alignment

2.4.3.16 #define ALIGNED_4_BYTES 0x4

4-bytes alignment

2.4.3.17 #define ALIGNED_8_BYTES 0x8

8-bytes alignment

2.4.3.18 #define ALIGNED_16_BYTES 0x10

16-bytes alignment

2.4.3.19 #define ALIGNED_32_BYTES 0x20

32-bytes alignment

2.4.3.20 #define ALIGNED_64_BYTES 0x40

64-bytes alignment

2.4.3.21 #define ALIGNED_128_BYTES 0x80

128-bytes alignment

2.4.3.22 #define ALIGNED_256_BYTES 0x100

256-bytes alignment

2.4.3.23 #define ALIGNED_512_BYTES 0x200

512-bytes alignment

2.4.3.24 #define ALIGNED_1024_BYTES 0x400

1024-bytes alignment

2.4.3.25 #define ALIGNED_2048_BYTES 0x800

2048-bytes alignment

2.4.3.26 #define ALIGNED_4096_BYTES 0x1000

4096-bytes alignment

2.4.3.27 #define ALIGN_SIZE(*SIZE, ALIGNMENT*) ((((uint32_t)(SIZE)) + ((uint32_t)(ALIGNMENT)) - 1) & (~(((uint32_t)(ALIGNMENT)) - 1)))

Align a given size - equivalent to ceil(SIZE,ALIGNMENT)

2.4.3.28 #define ALIGN_DOWN_SIZE(*SIZE, ALIGNMENT*) ((uint32_t)(SIZE) & (~(((uint32_t)(ALIGNMENT)) - 1)))

Align a given size to a lower aligned size - equivalent to floor(SIZE,ALIGNMENT)

2.4.3.29 #define ALIGNABLE_SIZE(*SIZE, ALIGNMENT*) (((uint32_t)(SIZE)) + ((uint32_t)(ALIGNMENT)) - 1)

Extend a given size to make it alignable.

2.4.3.30 #define ALIGN_ADDRESS(*ADDRESS, ALIGNMENT*) ((((uint32_t)(ADDRESS)) + ((uint32_t)(ALIGNMENT)) - 1) & (~(((uint32_t)(ALIGNMENT)) - 1)))

Align a given address - equivalent to ceil(ADDRESS,ALIGNMENT)

2.4.3.31 #define ALIGN_DOWN_ADDRESS(*ADDRESS, ALIGNMENT*) ((uint32_t)(ADDRESS) & (~(((uint32_t)(ALIGNMENT)) - 1)))

Align a given address to a lower aligned address - equivalent to floor(ADDRESS,ALIGNMENT)

2.4.3.32 #define ALIGN_DOWN_PHYS_ADDRESS(*ADDRESS, ALIGNMENT*) ((os_phys_ptr)(ADDRESS) & (~(((os_phys_ptr)(ALIGNMENT)) - 1)))

Align a given address to a lower aligned address - equivalent to floor(ADDRESS,ALIGNMENT)

2.4.3.33 #define IS_ALIGNED(*ADDRESS, ALIGNMENT*) (!(((uint32_t)(ADDRESS)) & (((uint32_t)(ALIGNMENT)) - 1)))

Check if a given address is aligned.

2.4.3.34 #define IS_COMPILER_ALIGNED(*ADDRESS, ALIGNMENT*) (((uint32_t)(ADDRESS)) % ((uint32_t)(ALIGNMENT)) == 0)

Check if a given address is aligned; Use it for cw_assert()

2.4.3.35 #define MEM_PART_SIZE(*NUM_OF_BUFFERS*) ALIGNABLE_SIZE((sizeof(os_mem_part_t) + ((NUM_OF_BUFFERS) ∗ sizeof(os_virt_ptr))), ALIGNED_4_BYTES)

Determines the needed memory size for a memory manager.

Use this macro to set the size of the memory space dedicated for the memory manager. This memory space should be passed as the *mem_part* parameter to osMemPartCreate(). If you use a smaller size the stack will be corrupted.

2.4.3.36 #define MEM_PART_DATA_SIZE(*NUM, BUF_SIZE, ALIGNMENT*) ALIGNABLE_SIZE(((NUM)∗(ALIGN_SIZE(BUF_SIZE, ALIGNMENT))), ALIGNMENT)

Determines the needed memory size for all memory blocks.

Use this macro to set the size of the memory space dedicated for the allocated memory blocks. This memory space should be passed as the *data_address* parameter to osMemPartCreate(). If you use a smaller size the stack will be corrupted.

2.4.3.37 #define OS_VALID_NUM_MASK 0xFFF00000

SmartDSP validity check number is encoded here.

2.4.3.38 #define OS_MEM_FLAGS_MASK 0x000FF000

Memory flags (see below) should be encoded in these bits.

2.4.3.39 #define OS_MEM_TYPE_MASK 0x0000001F

Memory type should be encoded in these bits.

2.4.3.40 #define OS_VALID_NUM 0xEC900000

Used by SmartDSP code to ensure that this is a valid heap.

2.4.3.41 #define OS_SMARTDSP_HEAP 0x00080000

Bit signifying that the heap is defined by SmartDSP OS.

2.4.3.42 #define OS_MEM_CACHEABLE_TYPE 0x00040000

Bit signifying that the heap is cacheable.

2.4.3.43 #define OS_MEM_SHARED_TYPE 0x00020000

Bit signifying that the heap is shared.

2.4.3.44 #define OS_MEM_HET_TYPE 0x00010000

Heap in PA DDR where SC owns memory allocator - for heterogeneous SoC only.

2.4.3.45 #define OS_MEM_M1_TYPE 0x00000001

Heap in M1 memory.

2.4.3.46 #define OS_MEM_M2_TYPE 0x00000002

Heap in M2 memory.

2.4.3.47 #define OS_MEM_M3_TYPE 0x00000003

Heap in M3 memory.

2.4.3.48 #define OS_MEM_DDR0_TYPE 0x00000004

Heap in DDR0 memory.

2.4.3.49 #define OS_MEM_DDR1_TYPE 0x00000005

Heap in DDR1 memory.

2.4.3.50 #define OS_MEM_QE_PRAM_TYPE 0x00000006

Heap in QE PRAM memory.

2.4.3.51 #define OS_MEM_MAPLE_PRAM_TYPE 0x00000007

Heap in MAPLE PRAM memory.

2.4.3.52 #define OS_MEM_MAPLE0_BD_RING_TYPE (OS_MEM_MAPLE_PRAM_TYPE | 0x00000010)

Heap in MAPLE0 BD ring memory.

2.4.3.53 #define OS_MEM_MAPLE1_BD_RING_TYPE (OS_MEM_MAPLE_PRAM_TYPE | 0x00000020)

Heap in MAPLE1 BD ring memory.

2.4.3.54 #define OS_MEM_MAPLE2_BD_RING_TYPE (OS_MEM_MAPLE_PRAM_TYPE | 0x00000030)

Heap in MAPLE2 BD ring memory.

2.4.3.55 #define OS_MEM_PRAM OS_MEM_QE_PRAM

QE parameter RAM memory.

2.4.3.56 #define OS_MEM_DDR_LOCAL OS_MEM_DDR0_LOCAL_CACHEABLE

DDR0 local memory.

2.4.3.57 #define OS_MEM_DDR_SHARED OS_MEM_DDR0_SHARED_NONCACHEABLE

DDR0 shared memory.

2.4.3.58 #define OS_MEM_DDR_SHARED_CACHEABLE OS_MEM_DDR0_SHARED_CACHEABLE

DDR0 shared cacheable memory.

2.4.3.59 #define OS_MEM_DDR_LOCAL_NONCACHEABLE OS_MEM_DDR0_LOCAL_↩ NONCACHEABLE

DDR0 local non-cacheable.

2.4.4 Enumeration Type Documentation

2.4.4.1 enum os_mem_type

Memory Type Enumeration.

Each heap managed by SmartDSP MUST have a unique identifier which will allows SmartDSP to differentiate it from other heaps. The *os_mem_type* enumeration provides identifiers for all SmartDSP defined heaps. The SmartDSP heap enumeration is split into 3 parts - "magic number", flags and memory type. Each such part has a bit mask as shown below: OS_VALID_NUM_MASK, OS_MEM_FLAGS_MASK and OS_MEM_TYPE_MASK. The "magic number" is used in order to verify the validity of a heap. The flags specify the characteristics of the heap. The memory type specifies the type of memory the heap resides in.

Warning

All heaps provided by SmartDSP will have OS_SMARTDSP_HEAP defined in their enumeration. In order to prevent a user defined heap's identifier from clashing with a SmartDSP identifier; the user MUST NOT define OS_SMARTDSP_HEAP as part of the heap identifier. All other flags are assumed to be valid in any user defined heap identifier. As such SmartDSP MAY perform checks on the heap type (cacheable, shared etc.) before performing any runtime code using the heap.

Not all heaps are supported on all architectures

Enumerator

OS_MEM_LOCAL_CACHEABLE Local cacheable memory - This heap is required in all Smart↩ DSP OS applications; the physical memory location is chosen by the OS based on the the architecture.

OS_MEM_LOCAL_NONCACHEABLE Local non-cacheable memory - This heap is required in most SmartDSP OS applications; the physical memory location is chosen by the OS based on the the architecture.

OS_MEM_SHARED_CACHEABLE Shared cacheable memory - This heap is required in all multicore SmartDSP OS applications; the physical memory location is chosen by the OS based on the the architecture.

OS_MEM_SHARED_NONCACHEABLE Shared non-cacheable memory - This heap is required in all multicore SmartDSP OS applications; the physical memory location is chosen by the OS based on the the architecture.

OS_MEM_QE_PRAM_SHARED_NONCACHEABLE QUICC Engine PRAM heap.

OS_MEM_MAPLE_PRAM_SHARED_NONCACHEABLE MAPLE heap.

OS_MEM_M2_SHARED_CACHEABLE Shared cacheable heap in M2 memory.
OS_MEM_M3_LOCAL_CACHEABLE Local cacheable heap in M3 memory.
OS_MEM_M3_LOCAL_NONCACHEABLE Local non-cacheable heap in M3 memory.
OS_MEM_M3_SHARED_NONCACHEABLE Shared non-cacheable heap in M3 memory.
OS_MEM_M3_SHARED_CACHEABLE Shared cacheable heap in M3 memory.
OS_MEM_DDR0_LOCAL_CACHEABLE Local cacheable heap in DDR0 memory.
OS_MEM_DDR0_LOCAL_NONCACHEABLE Local non-cacheable heap in DDR0 memory.
OS_MEM_DDR0_SHARED_NONCACHEABLE Shared non-cacheable heap in DDR0 memory.
OS_MEM_DDR0_SHARED_CACHEABLE Shared cacheable heap in DDR0 memory.
OS_MEM_DDR1_LOCAL_CACHEABLE Local cacheable heap in DDR1 memory.
OS_MEM_DDR1_LOCAL_NONCACHEABLE Local non-cacheable heap in DDR1 memory.
OS_MEM_DDR1_SHARED_NONCACHEABLE Shared non-cacheable heap in DDR1 memory.
OS_MEM_DDR1_SHARED_CACHEABLE Shared cacheable heap in DDR1 memory.
OS_MEM_HET_DDR1_SHARED_CACHEABLE Shared cacheable heap in PA DDR where SC owns memory allocator. This heap is shared for multicore devices and local for single core devices
OS_MEM_HET_DDR0_SHARED_CACHEABLE Shared cacheable heap in PA DDR where SC owns memory allocator. This heap is shared for multicore devices and local for single core devices
OS_MEM_MAPLE0_BD_RING_SHARED_NONCACHEABLE MAPLE0 BD ring heap.
OS_MEM_MAPLE1_BD_RING_SHARED_NONCACHEABLE MAPLE1 BD ring heap.
OS_MEM_MAPLE2_BD_RING_SHARED_NONCACHEABLE MAPLE2 BD ring heap.

2.4.5 MMU - Memory Management Unit API

2.4.5.1 Overview

`Memory Managment Unit, defines, enumerations and API`

The SmartDSP OS MMU module make a destinct separation between program and data aspects of the MMU

Modules

- Data MMU API
- Program MMU API
- SC3900 MMU Initialization API
- SC3900 MMU API

Data Structures

- struct os_mmu_error

Macros

- #define OS_MMU_SYSTEM_CONTEXT 1

Typedefs

- typedef uint32_t os_mmu_segment_handle

Enumerations

- enum mmu_memory_t { OS_MMU_DATA = 0, OS_MMU_PROG }

Functions

- os_status osMmuDataSegmentFind (os_mmu_segment_handle *descriptor)
- os_status osMmuDataSegmentCreate (os_mmu_segment_handle descriptor, os_const_virt_ptr virt_addr, os_const_phys_ptr phys_addr, uint32_t size, os_mmu_attr attr, void *arch)
- os_status osMmuDataSegmentEnable (os_mmu_segment_handle descriptor, bool enable)
- os_status osMmuDataSegmentSizeGet (os_mmu_segment_handle descriptor, uint32_t *size)
- os_status osMmuDataSegmentVirtBaseGet (os_mmu_segment_handle descriptor, os_virt_ptr *virt_base)
- os_status osMmuProgSegmentSizeGet (os_mmu_segment_handle descriptor, uint32_t *size)
- os_status osMmuProgSegmentVirtBaseGet (os_mmu_segment_handle descriptor, os_virt_ptr *virt_base)
- os_status osMmuDataVirtToPhys (os_const_virt_ptr virt_addr, os_phys_ptr *phys_addr)
- os_status osMmuDataVirtProbe (os_const_virt_ptr virt_addr)
- os_status osMmuDataSegmentProbe (os_const_virt_ptr virt_addr, os_mmu_segment_handle *segment_num)
- int osMmuDataNumOfSegmentsGet ()
- int osMmuDataNumOfUsedSegmentsGet ()
- int osMmuDataMaxIdGet ()
- os_status osMmuDataErrorDetect (struct os_mmu_error *err)
- void osMmuDataErrorClear ()

2.4.5.2 Data Structure Documentation

2.4.5.2.1 struct os_mmu_error

MMU error - populated when reporting MMU exceptions.

Data Fields

- uint32_t error_address
- uint32_t error_pc
- uint32_t rw_access
- uint32_t priviledge_level
- uint32_t access_width

2.4.5.2.1.1 Field Documentation

2.4.5.2.1.1.1 uint32_t os_mmu_error::error_address

violation address

2.4.5.2.1.1.2 uint32_t os_mmu_error::error_pc

program counter that caused the exception

2.4.5.2.1.1.3 uint32_t os_mmu_error::rw_access

read is 0, write is 1

2.4.5.2.1.1.4 uint32_t os_mmu_error::priviledge_level

user is 0, supervisor is 1

2.4.5.2.1.1.5 uint32_t os_mmu_error::access_width

Access Violation Width.

2.4.5.3 Macro Definition Documentation

2.4.5.3.1 #define OS_MMU_SYSTEM_CONTEXT 1

This is reserved and default ID for system context - MAY NOT BE CHANGED.

2.4.5.4 Typedef Documentation

2.4.5.4.1 typedef uint32_t os_mmu_segment_handle

MMU segment handle - passed to MMU segment API.

2.4.5.5 Enumeration Type Documentation

2.4.5.5.1 enum mmu_memory_t

MMU memory type - describes the type of memory managed.

Enumerator

OS_MMU_DATA Data MMU.
OS_MMU_PROG Program MMU.

2.4.5.6 Function Documentation

2.4.5.6.1 os_status osMmuDataSegmentFind (os_mmu_segment_handle * *descriptor*)

Find a free data segment in MMU

Parameters

out	*descriptor*	- descriptor's pointer

Return values

OS_SUCCESS	if successful
OS_FAIL	if not available

Warning

This function returns a free segment in MMU. There is no function that deletes the segment.

2.4.5.6.2 os_status osMmuDataSegmentCreate (os_mmu_segment_handle *descriptor,* os_const_virt_ptr *virt_addr,* os_const_phys_ptr *phys_addr,* uint32_t *size,* os_mmu_attr *attr,* void * *arch*)

Create data segment in MMU

This function creates new MMU data segment based on the user's configuration.

Parameters

in	*descriptor*	- Descriptor handle returned by osMmuDataSegmentFind()
in	*virt_addr*	- Virtual base address of this segment.
in	*phys_addr*	- Physical base address of this segment.
in	*size*	- Size of this segment in bytes.
in	*attr*	- Attributes of the segment (arch dependant).
in	*arch*	- Arch specific parameters

Return values

OS_SUCCESS	if successful

OS_ERR_MMU_WRONG_DESC	if descriptor is not legal
OS_ERR_MMU_WRONG_ALIGNMENT	if virt_addr or phys_addr alignment is illegal
OS_ERR_MMU_WRONG_SIZE	if size is illegal
OS_ERR_MMU_WRONG_BOUNDARY	if there are boundary crossing issues with the combination of size and virt_addr or phys_addr
OS_ERR_MMU_DSPE_ERR	if the MMU hardware rejected the configuration

2.4.5.6.3 os_status osMmuDataSegmentEnable (os_mmu_segment_handle *descriptor,* bool *enable*)

Enable a data segment in MMU

Parameters

in	*descriptor*	- decriptor's handle returned by osMmuDataSegmentFind()
in	*enable*	- TRUE - enable, FALSE - disable

Return values

OS_SUCCESS	if successful

2.4.5.6.4 os_status osMmuDataSegmentSizeGet (os_mmu_segment_handle *descriptor,* uint32_t * *size*)

Retrieve the segment size of data segment

Parameters

in	*descriptor*	- Descriptor handle returned by osMmuDataSegmentFind() or osMmuDataSegmentProbe()
in	*size*	- Size of this segment in bytes; used as return value.

Return values

OS_SUCCESS	if successful
OS_FAIL	if size calculation failed

2.4.5.6.5 os_status osMmuDataSegmentVirtBaseGet (os_mmu_segment_handle *descriptor,* os_virt_ptr * *virt_base*)

Retrieve the segment virtual base of data segment

Parameters

in	*descriptor*	- Descriptor handle returned by osMmuDataSegmentFind() or osMmuDataSegmentProbe()
in	*virt_base*	- Virtual base of segment; used as return value.

Return values

OS_SUCCESS	if successful
OS_FAIL	if size calculation failed

2.4.5.6.6 os_status osMmuProgSegmentSizeGet (os_mmu_segment_handle *descriptor,* uint32_t * *size*)

Retrieve the segment size of Program segment

Parameters

in	*descriptor*	- Descriptor handle returned by osMmuProgSegmentFind() or osMmuProgSegmentProbe()
in	*size*	- Size of this segment in bytes; used as return value.

Return values

OS_SUCCESS	if successful
OS_FAIL	if size calculation failed

2.4.5.6.7 os_status osMmuProgSegmentVirtBaseGet (os_mmu_segment_handle *descriptor,* os_virt_ptr * *virt_base*)

Retrieve the segment virtual base of Program segment

Parameters

in	*descriptor*	- Descriptor handle returned by osMmuProgSegmentFind() or osMmuProgSegmentProbe()
in	*virt_base*	- Virtual base of segment; used as return value.

Return values

OS_SUCCESS	if successful
OS_FAIL	if size calculation failed

2.4.5.6.8 os_status osMmuDataVirtToPhys (os_const_virt_ptr *virt_addr,* os_phys_ptr * *phys_addr*)

Translate a data virtual address to the corresponding physical one

Parameters

in	*virt_addr*	- virtual address pointer
out	*phys_addr*	- pointer to a physical address pointer

Returns

OS_SUCCESS if translation succedded
Error status, encoded in os_error.h, for other errors

2.4.5.6.9 os_status osMmuDataVirtProbe (os_const_virt_ptr *virt_addr*)

Check if a data virtual address is valid

Parameters

in	*virt_addr*	- virtual address pointer

Returns

OS_SUCCESS if address is legal within existing context
Error status, encoded in os_error.h, for other errors

2.4.5.6.10 os_status osMmuDataSegmentProbe (os_const_virt_ptr *virt_addr,* os_mmu_segment_handle * *segment_num*)

Check which MMU segment maps a given address

Parameters

in	*virt_addr*	- virtual address pointer
out	*segment_num*	- segment number that maps the address

Returns

OS_SUCCESS if address is legal within existing context
Error status, encoded in os_error.h, for other errors

2.4.5.6.11 int osMmuDataNumOfSegmentsGet ( )

Get number total number data segments

Returns

number of used data segments

2.4.5.6.12 int osMmuDataNumOfUsedSegmentsGet ( )

Get number of used data segments

Returns

number of used data segments

2.4.5.6.13 int osMmuDataMaxIdGet ( )

Get the number of maximum data id

Returns

The highest data id supported by MMU

2.4.5.6.14 os_status osMmuDataErrorDetect (struct os_mmu_error * *err*)

Read the MMU and populate the os_mmu_error structre

Parameters

out	*err*	- error structre

Returns

Error status, encoded in os_error.h,

2.4.5.6.15 void osMmuDataErrorClear ( )

Clear the data MMU status

2.4.5.7 Data MMU API

2.4.5.8 Program MMU API

2.4.5.8.1 Overview

Program Memory Managment Unit, defines, enumerations and API

Functions

- os_status osMmuProgSegmentFind (os_mmu_segment_handle *descriptor)
- os_status osMmuProgSegmentCreate (os_mmu_segment_handle descriptor, os_const_virt_ptr virt_addr, os_const_phys_ptr phys_addr, uint32_t size, os_mmu_attr attr, void *arch)
- os_status osMmuProgSegmentEnable (os_mmu_segment_handle descriptor, bool enable)
- uint32_t osMmuProgCurrentIDSet (uint32_t id)
- uint32_t osMmuProgCurrentIDGet ()
- os_status osMmuProgVirtToPhys (os_const_virt_ptr virt_addr, os_phys_ptr *phys_addr)
- os_status osMmuProgVirtProbe (os_const_virt_ptr virt_addr)
- os_status osMmuProgSegmentProbe (os_const_virt_ptr virt_addr, os_mmu_segment_handle *segment_num)
- os_status osMmuDataGetAttr (os_const_virt_ptr virt_addr, os_mmu_attr *attr)
- os_status osMmuProgGetAttr (os_const_virt_ptr virt_addr, os_mmu_attr *attr)
- int osMmuProgNumOfSegmentsGet ()
- int osMmuProgNumOfUsedSegmentsGet ()
- int osMmuProgMaxIdGet ()
- os_status osMmuProgErrorDetect (struct os_mmu_error *err)
- void osMmuProgErrorClear ()
- os_status osMmuDataPhysToVirtManual (os_const_phys_ptr phys_addr, os_virt_ptr *virt_addr)

2.4.5.8.2 Function Documentation

2.4.5.8.2.1 os_status osMmuProgSegmentFind (os_mmu_segment_handle * *descriptor*)

Find a free prog segment in MMU

Parameters

out	*descriptor*	- descriptor's pointer

Return values

OS_SUCCESS	if successful
OS_FAIL	if not available

Warning

This function returns a free segment in MMU. There is no function that deletes the segment.

2.4.5.8.2.2 os_status osMmuProgSegmentCreate (os_mmu_segment_handle *descriptor*, os_const_virt_ptr *virt_addr*, os_const_phys_ptr *phys_addr*, uint32_t *size*, os_mmu_attr *attr*, void * *arch*)

Create prog segment in MMU

This function creates new MMU prog segment based on the user's configuration.

Parameters

in	*descriptor*	- Descriptor handle returned by osMmuProgSegmentFind()
in	*virt_addr*	- Virtual base address of this segment.
in	*phys_addr*	- Physical base address of this segment.
in	*size*	- Size of this segment.
in	*attr*	- Attributes of the segment (arch dependant).
in	*arch*	- Arch specific parameters

Return values

OS_SUCCESS	if successful
OS_ERR_MMU_WRONG_DESC	if descriptor is not legal
OS_ERR_MMU_WRONG_ALIGNMENT	if virt_addr or phys_addr alignment is illegal
OS_ERR_MMU_WRONG_SIZE	if size is illegal
OS_ERR_MMU_WRONG_BOUNDARY	if there are boundary crossing issues with the combination of size and virt_addr or phys_addr

OS_ERR_MMU_DSPE↩ _ERR	if the MMU hardware rejected the configuration

2.4.5.8.2.3 os_status osMmuProgSegmentEnable (os_mmu_segment_handle *descriptor,* bool *enable*)

Enable a prog segment in MMU

Parameters

in	*descriptor*	- decriptor's handle returned by osMmuProgSegmentFind()
in	*enable*	- TRUE - enable, FALSE - disable

Return values

OS_SUCCESS	if successful

2.4.5.8.2.4 uint32_t osMmuProgCurrentIDSet (uint32_t *id*)

Set current prog ID

Set current prog ID. Optinal use-case is when two virtual segments reside on the same addresses

Parameters

in	*id*	- ID to set

Returns

previous prog ID

2.4.5.8.2.5 uint32_t osMmuProgCurrentIDGet ( )

Get current Prog ID

Returns

prog ID

2.4.5.8.2.6 os_status osMmuProgVirtToPhys (os_const_virt_ptr *virt_addr,* os_phys_ptr * *phys_addr*)

Translate a prog virtual address to the corresponding physical one

Parameters

in	*virt_addr*	- virtual address pointer
out	*phys_addr*	- pointer to a physical address pointer

Returns

OS_SUCCESS if translation succedded
Error status, encoded in os_error.h, for other errors

2.4.5.8.2.7 os_status osMmuProgVirtProbe (os_const_virt_ptr *virt_addr*)

Check if a prog virtual address is valid

Parameters

in	*virt_addr*	- virtual address pointer

Returns

OS_SUCCESS if address is legal within existing context
Error status, encoded in os_error.h, for other errors

2.4.5.8.2.8 os_status osMmuProgSegmentProbe (os_const_virt_ptr *virt_addr,* os_mmu_segment_handle * *segment_num*)

Check which MMU segment maps a given address

Parameters

in	*virt_addr*	- virtual address pointer
out	*segment_num*	- segment number that maps the address

Returns

OS_SUCCESS if address is legal within existing context
Error status, encoded in os_error.h, for other errors

2.4.5.8.2.9 os_status osMmuDataGetAttr (os_const_virt_ptr *virt_addr,* os_mmu_attr * *attr*)

Returns the attribute of MMU data segement according to virtual address

Example: if the function returns OS_SUCCESS, use the attr as follows: (attr & MMU_DATA_CACHE↩ ABLE_WRITETHROUGH) to check if the segment is cacheable (sc3x00)

Parameters

in	*virt_addr*	- virtual address pointer
out	*attr*	- attributes of the segement in a form of masks. masks defined in specific arch h file should be used

Returns

OS_SUCCESS if address is legal within existing context
Error status, encoded in os_error.h, for other errors

2.4.5.8.2.10 os_status osMmuProgGetAttr (os_const_virt_ptr *virt_addr,* os_mmu_attr * *attr*)

Returns the attribute of MMU program segement according to virtual address

Example: if the function returns OS_SUCCESS, use the attr as follows: (attr & MMU_PROG_L2_CA↩ CHEABLE) to check if the segment is cacheable (sc3x00)

Parameters

in	*virt_addr*	- virtual address pointer
out	*attr*	- attributes of the segement in a form of masks. masks defined in specific arch h file should be used

Returns

OS_SUCCESS if address is legal within existing context
Error status, encoded in os_error.h, for other errors

2.4.5.8.2.11 int osMmuProgNumOfSegmentsGet ( )

Get number total number prog segments

Returns

number of used prog segments

2.4.5.8.2.12 int osMmuProgNumOfUsedSegmentsGet ( )

Get number of used prog segments

Returns

number of used prog segments

2.4.5.8.2.13 int osMmuProgMaxIdGet ( )

Get the number of maximum prog id

Returns

The highest prog id supported by MMU

2.4.5.8.2.14 os_status osMmuProgErrorDetect (struct os_mmu_error * *err*)

Read the MMU and populate the os_mmu_error structre

Parameters

out	*err*	- error structre

Returns

Error status, encoded in os_error.h,

2.4.5.8.2.15 void osMmuProgErrorClear ( )

Clear the data MMU status

2.4.5.8.2.16 os_status osMmuDataPhysToVirtManual (os_const_phys_ptr *phys_addr,* os_virt_ptr * *virt_addr*)

Translate data physical address to virtual by manual inspection

Parameters

in	*phys_addr*	- physical address
out	*virt_addr*	- virtual address

Warning

This is an extremely slow implementation and shouldn't be used in runtime

Returns

OS_SUCCESS if segment exists

2.4.5.9 SC3900 MMU Initialization API

2.4.5.9.1 Overview

Memory Management Unit Initialization API

Data Structures

- union sc39xx_mmu_init_params_t

2.4.5.9.2 Data Structure Documentation

2.4.5.9.2.1 union sc39xx_mmu_init_params_t

Platform configuration parameters.

Data Fields

- uint32_t m_cr
- uint32_t gr:1
- uint32_t gp:1
- uint32_t ge:1
- uint32_t ice:1
- uint32_t dce:1
- uint32_t cvae:1
- uint32_t soee:1
- uint32_t vccc:1
- uint32_t ncee:1
- uint32_t mpe:1
- uint32_t ate:1
- uint32_t eccee:1

2.4.5.9.2.1.1 Field Documentation

2.4.5.9.2.1.2 uint32_t sc39xx_mmu_init_params_t::gr

Guarded Rewind - TRUE/FALSE enables/disables the MMU assertion to rewind to the core, when a speculative guarded read occurred.

2.4.5.9.2.1.3 uint32_t sc39xx_mmu_init_params_t::gp

Guarded Priority in SGB -.

- When TRUE Guarded access priority is High,
- When FALSE, Guarded access priority is Low

2.4.5.9.2.1.4 uint32_t sc39xx_mmu_init_params_t::ge

Gather Enable - TRUE/FALSE enables/disables SGB gathering function.

2.4.5.9.2.1.5 uint32_t sc39xx_mmu_init_params_t::ice

Instruction Cache Enable - When TRUE, Instruction Cache is enabled.

Instruction accesses cache policy determined by relevant MMU descriptor. When FALSE, Instruction Cache is disabled overriding ICache and L2 cache policy to non-cacheable for all instruction accesses

2.4.5.9.2.1.6 uint32_t sc39xx_mmu_init_params_t::dce

Data Cache Enable - When TRUE, Data Cache is enabled.

Data accesses cache policy determined by relevant MMU descriptor. When FALSE, Data Cache is disabled overriding DCache and L2 cache policy to non-cacheable for all data accesses

2.4.5.9.2.1.7 uint32_t sc39xx_mmu_init_params_t::cvae

Capture Violation Address Enable - TRUE/FALSE enables/disables the capture mechanism of the violated access address.

If disabled, the MMU registers M_PVA and M_DVA are not updated if there is an error (power saving mode)

2.4.5.9.2.1.8 uint32_t sc39xx_mmu_init_params_t::soee

Stack Overrun Error Enabled - TRUE/FALSE enables/disables precise exceptions as a result of stack related accesses that match non-stack descriptors.

2.4.5.9.2.1.9 uint32_t sc39xx_mmu_init_params_t::vccc

Voluntary Cache Commands Cancel -.

- When TRUE, the MMU and CME cancel the following cache performance commands: DMALL↩ OC, DFETCHx
- When false, cache performance commands are enabled

2.4.5.9.2.1.10 uint32_t sc39xx_mmu_init_params_t::ncee

Noncacheable Cache Commands Error Enabled - When TRUE,.

- generates precise exception on granular data cache commands to non-cacheable memory: DFLUSH, DSYNC, DINVALIDATE, DTUNLOCK, DMALLOC, DFETCHx
- generates non-precise exception on granular program cache commands to non-cacheable memory: PINVALIDATE, PUNLOCK, PFETCHx When FALSE, no exceptions are generated due to cache commands to non-cacheable memory

2.4.5.9.2.1.11 uint32_t sc39xx_mmu_init_params_t::mpe

Memory Protection Enable - TRUE/FALSE enables/disables the protection-checking function in all enabled segment descriptors.

It also enables/disables the miss interrupt support on a miss access

2.4.5.9.2.1.12 uint32_t sc39xx_mmu_init_params_t::ate

Address Translation Enable - TRUE/FALSE enables/disables the address translation mechanism.

2.4.5.9.2.1.13 uint32_t sc39xx_mmu_init_params_t::eccee

Error Detection Code Exception Enable - TRUE/FALSE enables/disables the the ECC exception.

2.4.5.9.2.1.14 uint32_t sc39xx_mmu_init_params_t::m_cr

all attributes above

2.4.5.10 SC3900 MMU API

2.4.5.10.1 Overview

SC3900 specific Program and Data MMU API

Macros

- #define MMU_FIELD_SET(val, shift) (os_mmu_attr)((val) << (shift))
- #define MMU_ATTR_FIELD_SET(reg_shift, shift, val) (os_mmu_attr)(MMU_FIELD_SET((val), (shift)) << (reg_shift))
- #define MMU_ATTR_FIELD_GET(reg, reg_shift, shift, mask) (os_mmu_attr)(((uint64_t)(reg) & MMU_ATTR_FIELD_SET((reg_shift), (shift), (mask))) >> (reg_shift))
- #define MMU_BIT (1)
- #define MMU_BIT_SET(bit_num) MMU_FIELD_SET(MMU_BIT, (bit_num))
- #define MMU_ATTR_BIT_SET(reg_shift, bit_num) (MMU_BIT_SET(bit_num) << (reg_shift))
- #define MMU_ATTR_BIT_GET(reg, reg_shift, bit_num) (((uint64_t)(reg) & MMU_ATTR_FIE↩ LD_SET((reg_shift), (bit_num), MMU_BIT)) >> (reg_shift))
- #define MMU_REGA_SHIFT (0)
- #define MMU_REGC_SHIFT (32)
- #define MMU_MAX_TID 0xFF
- #define MMU_ATTR_GET_PSDA(attr) (((uint64_t)(attr) & (MMU_ATTR_PSDA_MASK)) >> (MMU_REGA_SHIFT))
- #define MMU_ATTR_GET_PSDC(attr) (((uint64_t)(attr) & (MMU_ATTR_PSDC_MASK)) >> (MMU_REGC_SHIFT))
- #define MMU_ATTR_GET_DSDA(attr) (((uint64_t)(attr) & (MMU_ATTR_DSDA_MASK)) >> (MMU_REGA_SHIFT))
- #define MMU_ATTR_GET_DSDC(attr) (((uint64_t)(attr) & (MMU_ATTR_DSDC_MASK)) >> (MMU_REGC_SHIFT))
- #define MMU_ATTR_GET_MATC(attr) ((((uint64_t)(attr)<<(MMU_REGC_SHIFT)) & (MM↩ U_ATTR_DSDC_MASK)) >> (MMU_REGC_SHIFT))
- #define MMU_ATTR_GET_MATC(attr) ((((uint64_t)(attr)<<(MMU_REGC_SHIFT)) & (MM↩ U_ATTR_DSDC_MASK)) >> (MMU_REGC_SHIFT))
- #define MMU_PHYS_ADDR_LOW(addr) ((uint64_t)(addr) & 0xFFFFFFFFULL)
- #define MMU_PHYS_ADDR_HIGH(addr) (((uint64_t)(addr) >> 32) & 0x0000000F)

MMU Program Register Bits

```
These bits are used for setting the attributes of
Program MMU segments
```

- #define MMU_PROG_MATT_ENABLE MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 0)
- #define MMU_PROG_DEF_XPERM_SUPER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 2)
- #define MMU_PROG_DEF_XPERM_USER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 4)
- #define MMU_PROG_CACHEABLE_REGION MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 5)
- #define MMU_PROG_NO_PREFETCH MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 0)
- #define MMU_PROG_PREFETCH_ON_MISS MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 1)
- #define MMU_PROG_PREFETCH_ANY MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 2)
- #define MMU_PROG_FLEX_SEGMENT_MODEL MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 10)
- #define MMU_PROG_PID(id) MMU_ATTR_FIELD_SET(MMU_REGC_SHIFT, 8, id)
- #define MMU_PROG_L2_PARTITION_ID(id) MMU_ATTR_FIELD_SET(MMU_REGC_SHIFT, 20, id)
- #define MMU_PROG_COHERENT MMU_ATTR_BIT_SET(MMU_REGC_SHIFT, 16)

MMU Data Register Bits

```
These bits are used for setting the attributes of
Data MMU segments
```

- #define MMU_DATA_MATT_ENABLE MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 0)
- #define MMU_DATA_DEF_WPERM_SUPER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 1)
- #define MMU_DATA_DEF_RPERM_SUPER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 2)
- #define MMU_DATA_DEF_WPERM_USER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 3)
- #define MMU_DATA_DEF_RPERM_USER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 4)
- #define MMU_DATA_CACHEABLE_REGION MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 5)
- #define MMU_DATA_WRITETHROUGH_REGION MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 6)
- #define MMU_DATA_NO_PREFETCH MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 0)
- #define MMU_DATA_PREFETCH_ON_MISS MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 1)
- #define MMU_DATA_PREFETCH_ANY MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 2)
- #define MMU_DATA_FLEX_SEGMENT_MODEL MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 10)
- #define MMU_DATA_DID(id) MMU_ATTR_FIELD_SET(MMU_REGC_SHIFT, 8, id)
- #define MMU_DATA_COHERENT MMU_ATTR_BIT_SET(MMU_REGC_SHIFT, 16)
- #define MMU_DATA_PERIPHERAL MMU_ATTR_BIT_SET(MMU_REGC_SHIFT, 17)
- #define MMU_DATA_GUARDED MMU_ATTR_BIT_SET(MMU_REGC_SHIFT, 18)
- #define MMU_DATA_STACK_ENABLE MMU_ATTR_BIT_SET(MMU_REGC_SHIFT, 19)
- #define MMU_DATA_L2_PARTITION_ID(id) MMU_ATTR_FIELD_SET(MMU_REGC_SHIFT, 20, id)

- #define M_CR_GE 0x00000800
- #define M_CR_ICE 0x00000400
- #define M_CR_DCE 0x00000200
- #define M_CR_SOEE 0x00000080
- #define M_CR_VCCC 0x00000040
- #define M_CR_VCEE 0x00000020
- #define M_CR_MPE 0x00000008
- #define M_CR_ATE 0x00000004
- #define M_CR_ECCEE 0x00000002
- #define M_CR_CMIR 0x00000001
- #define M_DSR_DCCV 0x00200000
- #define M_DSR_DAVW 0x00180000
- #define M_DSR_DAVD 0x00020000
- #define M_DSR_DMSD 0x00000000
- #define M_DSR_DPV 0x00000002
- #define M_DSR_DNAE 0x00000008
- #define M_DSR_DSM 0x00000001
- #define M_DSR_DECC 0x0000000f
- #define M_DSR_DRE 0x0000000e
- #define M_DSR_DNC 0x00000009
- #define M_DSR_DNCC 0x00000007
- #define M_DSR_DSOV 0x00000006
- #define M_DSR_DSAS 0x00000005
- #define M_DSR_DPAS 0x00000003
- #define M_PSR_PMSD 0x00000000
- #define M_PSR_PNAE 0x00000008
- #define M_PSR_PSM 0x00000001
- #define M_PSR_PECC 0x0000000f
- #define M_PSR_PPV 0x00000002
- #define M_PSR_PFE 0x0000000e
- #define M_PSR_PNC 0x00000009

MMU Segment Register Masks

- #define MMU_ATTR_PSDA_MASK
- #define MMU_ATTR_PSDC_MASK (MMU_PROG_PID(MMU_MAX_TID) | MMU_PROG_↩ L2_PARTITION_ID(3) | MMU_PROG_COHERENT)
- #define MMU_ATTR_DSDA_MASK
- #define MMU_ATTR_DSDC_MASK

SC39XX MMU defines

- #define MMU_SC39XX_MIN_SIZE 0x1000
- #define MMU_SC39XX_MAX_FLEX_SIZE 0xFC0000
- #define MMU_MATT_ADDRESS_MASK 0xfffff000
- #define MMU_MATT_ADDRESS_EXT_MASK 0x0000000f

SC39XX MMU reserved MATT

- #define MMU_DSP_CLUSTER_MATT 0
- #define MMU_SYS_STACK_MATT 2
- #define MMU_TASK_STACK_MATT 3

2.4.5.10.2 Macro Definition Documentation

2.4.5.10.2.1 #define MMU_FIELD_SET(*val, shift*) (os_mmu_attr)((val) << (shift))

Set a multi-bit field in a register.

2.4.5.10.2.2 #define MMU_ATTR_FIELD_SET(*reg_shift, shift, val*) (os_mmu_attr)(MMU_FIELD_SET((val), (shift)) << (reg_shift))

Set a multi-bit MMU attribute field.

2.4.5.10.2.3 #define MMU_ATTR_FIELD_GET(*reg, reg_shift, shift, mask*) (os_mmu_attr)(((uint64_t)(reg) & MMU_ATTR_FIELD_SET((reg_shift), (shift), (mask))) >> (reg_shift))

Get a multi-bit attribute field.

2.4.5.10.2.4 #define MMU_BIT (1)

Single bit.

2.4.5.10.2.5 #define MMU_BIT_SET(*bit_num*) MMU_FIELD_SET(MMU_BIT, (bit_num))

Set a single bit.

2.4.5.10.2.6 #define MMU_ATTR_BIT_SET(*reg_shift, bit_num*) (MMU_BIT_SET(bit_num) << (reg_shift))

Set a single bit MMU attribute field.

2.4.5.10.2.7 #define MMU_ATTR_BIT_GET(*reg, reg_shift, bit_num*) (((uint64_t)(reg) & MMU_ATTR_FIELD_SET((reg_shift), (bit_num), MMU_BIT)) >> (reg_shift))

Get a single bit MMU attribute field.

2.4.5.10.2.8 #define MMU_REGA_SHIFT (0)

Bit shift of MMU REGA in the os_mmu_attr.

2.4.5.10.2.9 #define MMU_REGC_SHIFT (32)

Bit shift of MMU REGC in the os_mmu_attr.

2.4.5.10.2.10 #define MMU_MAX_TID 0xFF

Maximum supported D/PID in MMU.

2.4.5.10.2.11 #define MMU_PROG_MATT_ENABLE MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 0)

Segment Descriptor Enable.

2.4.5.10.2.12 #define MMU_PROG_DEF_XPERM_SUPER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 2)

Fetch Permission For Program Accesses in Supervisor Level.

2.4.5.10.2.13 #define MMU_PROG_DEF_XPERM_USER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 4)

Fetch Permission For Program Accesses in User Level.

2.4.5.10.2.14 #define MMU_PROG_CACHEABLE_REGION MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 5)

Segment is Cacheable in the ICache and L2 Cache.

2.4.5.10.2.15 #define MMU_PROG_NO_PREFETCH MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 0)

No prefetch on segment.

2.4.5.10.2.16 #define MMU_PROG_PREFETCH_ON_MISS MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 1)

Prefetch after a cache miss on segment.

2.4.5.10.2.17 #define MMU_PROG_PREFETCH_ANY MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 2)

Prefetch after a cache miss/hit on segment.

2.4.5.10.2.18 #define MMU_PROG_FLEX_SEGMENT_MODEL MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 10)

Indicates that the segment uses Flexible Segment Model.

2.4.5.10.2.19 #define MMU_PROG_PID(*id*) MMU_ATTR_FIELD_SET(MMU_REGC_SHIFT, 8, id)

Segment Program task ID; PID = 0 used to define shared memory and matches any PID generated by the core.

2.4.5.10.2.20 #define MMU_PROG_L2_PARTITION_ID(*id*) MMU_ATTR_FIELD_SET(MMU_REGC_SHIFT, 20, id)

L2 Partitioning ID.

2.4.5.10.2.21 #define MMU_PROG_COHERENT MMU_ATTR_BIT_SET(MMU_REGC_SHIFT, 16)

Program Coherent Memory Segment.

2.4.5.10.2.22 #define MMU_DATA_MATT_ENABLE MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 0)

Segment Descriptor Enable.

2.4.5.10.2.23 #define MMU_DATA_DEF_WPERM_SUPER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 1)

Write Permission For Data Accesses in Supervisor Level.

2.4.5.10.2.24 #define MMU_DATA_DEF_RPERM_SUPER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 2)

Read Permission For Data Accesses in Supervisor Level.

2.4.5.10.2.25 #define MMU_DATA_DEF_WPERM_USER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 3)

Write Permission For Data Accesses in User Level.

2.4.5.10.2.26 #define MMU_DATA_DEF_RPERM_USER MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 4)

Read Permission For Data Accesses in User Level.

2.4.5.10.2.27 #define MMU_DATA_CACHEABLE_REGION MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 5)

Segment is Cacheable in the DCache and L2 Cache.

2.4.5.10.2.28 #define MMU_DATA_WRITETHROUGH_REGION MMU_ATTR_BIT_SET(MMU_REGA_SHIFT, 6)

Segment is write-through.

2.4.5.10.2.29 #define MMU_DATA_NO_PREFETCH MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 0)

No prefetch on segment.

2.4.5.10.2.30 #define MMU_DATA_PREFETCH_ON_MISS MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 1)

Prefetch after a cache miss on segment.

2.4.5.10.2.31 #define MMU_DATA_PREFETCH_ANY MMU_ATTR_FIELD_SET(MMU_REGA_↩ SHIFT, 7, 2)

Prefetch after a cache miss/hit on segment.

2.4.5.10.2.32 #define MMU_DATA_FLEX_SEGMENT_MODEL MMU_ATTR_BIT_SET(MMU_R↩ EGA_SHIFT, 10)

Indicates that the segment uses Flexible Segment Model.

2.4.5.10.2.33 #define MMU_DATA_DID(*id*) MMU_ATTR_FIELD_SET(MMU_REGC_SHIFT, 8, id)

Segment Program task ID; DID = 0 used to define shared memory and matches any DID generated by the core.

2.4.5.10.2.34 #define MMU_DATA_COHERENT MMU_ATTR_BIT_SET(MMU_REGC_SHIFT, 16)

Data Coherent Memory Segment.

2.4.5.10.2.35 #define MMU_DATA_PERIPHERAL MMU_ATTR_BIT_SET(MMU_REGC_SHIFT, 17)

Data Peripheral Space.

2.4.5.10.2.36 #define MMU_DATA_GUARDED MMU_ATTR_BIT_SET(MMU_REGC_SHIFT, 18)

Data Guarded Segment.

2.4.5.10.2.37 #define MMU_DATA_STACK_ENABLE MMU_ATTR_BIT_SET(MMU_REGC_SH↩ IFT, 19)

Stack Descriptor - Stack related accesses are permitted.

2.4.5.10.2.38 #define MMU_DATA_L2_PARTITION_ID(*id*) MMU_ATTR_FIELD_SET(MMU_↩ REGC_SHIFT, 20, id)

L2 Partitioning ID.

2.4.5.10.2.39 #define M_CR_GE 0x00000800

Gather Enable.

2.4.5.10.2.40 #define M_CR_ICE 0x00000400

Instruction Cache Enable.

2.4.5.10.2.41 #define M_CR_DCE 0x00000200

Instruction Data Enable.

2.4.5.10.2.42 #define M_CR_SOEE 0x00000080

Stack Overrun Error Enabled.

2.4.5.10.2.43 #define M_CR_VCCC 0x00000040

Fetch Cache Commands Cancel.

2.4.5.10.2.44 #define M_CR_VCEE 0x00000020

Voulantary Cache Commands Error Enabled.

2.4.5.10.2.45 #define M_CR_MPE 0x00000008

Memory Protection Enable.

2.4.5.10.2.46 #define M_CR_ATE 0x00000004

Address Translation Enable.

2.4.5.10.2.47 #define M_CR_ECCEE 0x00000002

Error Detection Code Exception Enable.

2.4.5.10.2.48 #define M_CR_CMIR 0x00000001

Clear MMU Interrupt Request.

2.4.5.10.2.49 #define M_DSR_DCCV 0x00200000

Data Cache Command Violation.

2.4.5.10.2.50 #define M_DSR_DAVW 0x00180000

Data Access Violation Width.

2.4.5.10.2.51 #define M_DSR_DAVD 0x00020000

Data Access Violation Direction.

2.4.5.10.2.52 #define M_DSR_DMSD 0x00000000

Data Multiple Segment Descriptor Hit.

2.4.5.10.2.53 #define M_DSR_DPV 0x00000002

Data Permission Violation.

2.4.5.10.2.54 #define M_DSR_DNAE 0x00000008

Data Non-aligned Access Error.

2.4.5.10.2.55 #define M_DSR_DSM 0x00000001

Data Segment Miss.

2.4.5.10.2.56 #define M_DSR_DECC 0x0000000f

Data ECC Error.

2.4.5.10.2.57 #define M_DSR_DRE 0x0000000e

Data Read Error.

2.4.5.10.2.58 #define M_DSR_DNC 0x00000009

NC hit.

2.4.5.10.2.59 #define M_DSR_DNCC 0x00000007

Noncacheable Cache Command.

2.4.5.10.2.60 #define M_DSR_DSOV 0x00000006

Stack Overrun Violation.

2.4.5.10.2.61 #define M_DSR_DSAS 0x00000005

Semaphore Access Size Error.

2.4.5.10.2.62 #define M_DSR_DPAS 0x00000003

Peripheral Access Size Error.

2.4.5.10.2.63 #define M_PSR_PMSD 0x00000000

Program Multiple Segment Descriptor Hit.

2.4.5.10.2.64 #define M_PSR_PNAE 0x00000008

Program Non-aligned Access Error.

2.4.5.10.2.65 #define M_PSR_PSM 0x00000001

Program Segment Miss.

2.4.5.10.2.66 #define M_PSR_PECC 0x0000000f

Program ECC.

2.4.5.10.2.67 #define M_PSR_PPV 0x00000002

Program Permission Violation.

2.4.5.10.2.68 #define M_PSR_PFE 0x0000000e

Program Fetch Error.

2.4.5.10.2.69 #define M_PSR_PNC 0x00000009

NC hit.

2.4.5.10.2.70 #define MMU_ATTR_PSDA_MASK

Value:

```
( MMU_PROG_MATT_ENABLE | MMU_PROG_DEF_XPERM_SUPER |
      MMU_PROG_DEF_XPERM_USER |           \
      MMU_PROG_CACHEABLE_REGION |
      MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 3) |
      \
      MMU_PROG_FLEX_SEGMENT_MODEL )
```

Program Segment Descriptor Registers A valid bit mask.

2.4.5.10.2.71 #define MMU_ATTR_PSDC_MASK (MMU_PROG_PID(MMU_MAX_TID) | MMU_PROG_L2_PARTITION_ID(3) | MMU_PROG_COHERENT)

Program Segment Descriptor Registers C valid bit mask.

2.4.5.10.2.72 #define MMU_ATTR_DSDA_MASK

Value:

```
( MMU_DATA_MATT_ENABLE | MMU_DATA_DEF_WPERM_SUPER |
      MMU_DATA_DEF_RPERM_SUPER    |       \
      MMU_DATA_DEF_WPERM_USER |
      MMU_DATA_DEF_RPERM_USER | MMU_DATA_CACHEABLE_REGION |
       \
      MMU_DATA_WRITETHROUGH_REGION |
      MMU_ATTR_FIELD_SET(MMU_REGA_SHIFT, 7, 3)       |
      \
      MMU_DATA_FLEX_SEGMENT_MODEL )
```

Data Segment Descriptor Registers A valid bit mask.

2.4.5.10.2.73 #define MMU_ATTR_DSDC_MASK

Value:

```
( MMU_DATA_DID(MMU_MAX_TID) | MMU_DATA_COHERENT |
      MMU_DATA_PERIPHERAL |                 \
      MMU_DATA_GUARDED | MMU_DATA_STACK_ENABLE |
      MMU_DATA_L2_PARTITION_ID(3) )
```

Data Segment Descriptor Registers C valid bit mask.

2.4.5.10.2.74 #define MMU_ATTR_GET_PSDA(*attr*) (((uint64_t)(attr) & (MMU_ATTR_PSDA_MASK)) >> (MMU_REGA_SHIFT))

Program Segment Descriptor Registers A.

2.4.5.10.2.75 #define MMU_ATTR_GET_PSDC(*attr*) (((uint64_t)(attr) & (MMU_ATTR_PSDC_MASK)) >> (MMU_REGC_SHIFT))

Program Segment Descriptor Registers C.

2.4.5.10.2.76 #define MMU_ATTR_GET_DSDA(*attr*) (((uint64_t)(attr) & (MMU_ATTR_DSDA_MASK)) >> (MMU_REGA_SHIFT))

Data Segment Descriptor Registers A.

2.4.5.10.2.77 #define MMU_ATTR_GET_DSDC(*attr*) (((uint64_t)(attr) & (MMU_ATTR_DSDC_MASK)) >> (MMU_REGC_SHIFT))

Data Segment Descriptor Registers C.

2.4.5.10.2.78 #define MMU_ATTR_GET_MATC(*attr*) ((((uint64_t)(attr)<<(MMU_RE↩ GC_SHIFT)) & (MMU_ATTR_DSDC_MASK)) >> (MMU_REGC_SHIFT))

attribute from Registers C

2.4.5.10.2.79 #define MMU_ATTR_GET_MATC(*attr*) ((((uint64_t)(attr)<<(MMU_RE↩ GC_SHIFT)) & (MMU_ATTR_DSDC_MASK)) >> (MMU_REGC_SHIFT))

attribute from Registers C

2.4.5.10.2.80 #define MMU_PHYS_ADDR_LOW(*addr*) ((uint64_t)(addr) & 0xFFFFFFFFULL)

Mask of relevant bits for low physical address.

2.4.5.10.2.81 #define MMU_PHYS_ADDR_HIGH(*addr*) (((uint64_t)(addr) >> 32) & 0x0000000F)

Mask of relevant bits for high physical address.

2.4.5.10.2.82 #define MMU_SC39XX_MIN_SIZE 0x1000

4K Minimum Size

2.4.5.10.2.83 #define MMU_SC39XX_MAX_FLEX_SIZE 0xFC0000

16MB - 256KB Maximum Size in flexible model

2.4.5.10.2.84 #define MMU_MATT_ADDRESS_MASK 0xfffff000

MMU matt low adress masking.

2.4.5.10.2.85 #define MMU_MATT_ADDRESS_EXT_MASK 0x0000000f

MMU matt extantion adress masking.

2.4.5.10.2.86 #define MMU_DSP_CLUSTER_MATT 0

MMU numeric number of DSP cluster matt.

2.4.5.10.2.87 #define MMU_SYS_STACK_MATT 2

MMU numeric number of system stack matt.

2.4.5.10.2.88 #define MMU_TASK_STACK_MATT 3

MMU numeric number of task stack matt.

2.4.6 Memory Allocation API

2.4.6.1 Overview

Memory allocation API

Functions

- void ∗ osMalloc (uint32_t size, os_mem_type type)
- void ∗ osFastMalloc (uint32_t size, os_mem_type type)
- void ∗ osAlignedMalloc (uint32_t size, os_mem_type type, uint32_t alignment)
- void ∗ osFastAlignedMalloc (uint32_t size, os_mem_type type, uint32_t alignment)
- os_status osFree (void ∗addr)
- INLINE bool osMemTypeIsValid (os_mem_type type)
- void osDumpMemory (os_mem_type type)
- uint32_t osGetFreeMemSize (os_mem_type type)
- uint32_t osGetMaxBlockSize (os_mem_type type, uint32_t alignment)

2.4.6.2 Function Documentation

2.4.6.2.1 void∗ osMalloc (uint32_t *size*, os_mem_type *type*)

Simple memory allocation, from the system heap, with zeroing.

Parameters

in	*size*	- The memory size to allocate (in bytes).
in	*type*	- The memory type to allocate.

Returns

Pointer to the allocated memory, NULL if not available

2.4.6.2.2 void* osFastMalloc (uint32_t *size*, os_mem_type *type*)

Simple memory allocation, from the system heap, without initialization.

Parameters

in	*size*	- The memory size to allocate (in bytes).
in	*type*	- The memory type to allocate.

Returns

Pointer to the allocated memory, NULL if not available

2.4.6.2.3 void* osAlignedMalloc (uint32_t *size*, os_mem_type *type*, uint32_t *alignment*)

Aligned memory allocation, from the system heap.

Parameters

in	*size*	- The memory size to allocate (in bytes).
in	*type*	- The memory type to allocate.
in	*alignment*	- address alignment required.

Returns

Pointer to the allocated memory, NULL if not available

2.4.6.2.4 void* osFastAlignedMalloc (uint32_t *size*, os_mem_type *type*, uint32_t *alignment*)

Aligned memory allocation, from the system heap, without initialization.

Parameters

in	*size*	- The memory size to allocate (in bytes).
in	*type*	- The memory type to allocate.
in	*alignment*	- address alignment required.

Returns

Pointer to the allocated memory, NULL if not available

2.4.6.2.5 os_status osFree (void * *addr*)

Free memory allocation.

Parameters

in	*addr*	- Pointer to the allocated memory (as returned by all memory allocation functions - osMalloc, osFastMalloc, osAlignedMalloc and osFastAlignedMalloc).

Returns

OS_SUCCESS if operation succeeded or an error code otherwise

2.4.6.2.6 INLINE bool osMemTypeIsValid (os_mem_type *type*)

Verifies that a heap indicator is valid.

Parameters

in	*type*	- The heap indicator.

Return values

TRUE	- If the heap is valid
FALSE	- If the heap is invalid

2.4.6.2.7 void osDumpMemory (os_mem_type *type*)

Print statistics and memory map of given memory type.

Parameters

in	*type*	- The heap indicator.

Return values

None	

2.4.6.2.8 uint32_t osGetFreeMemSize (os_mem_type *type*)

free heap size.

Parameters

in	*type*	- The heap indicator.

Return values

Free	memory amount (in Bytes)

2.4.6.2.9 uint32_t osGetMaxBlockSize (os_mem_type *type*, uint32_t *alignment*)

maximum free consecutive memory.

Parameters

in	*type*	- The heap indicator.
in	*alignment*	- address alignment required.

Return values

Maximum	free consecutive memory size aligned to given parameter (in Bytes)

2.4.7 Buffers Pool Management API

2.4.7.1 Overview

Memory allocation API

Macros

- #define osMemBlockSize(mem_part) ((mem_part)->block_size)
- #define osMemBlockOffset(mem_part) ((mem_part)->buffer_offset)

- #define osMemBlockUnsafeGet(mem_part) (((mem_part)->curr_index >= 0) ? ((void *)(mem_part)->array_addr[((mem_part)->curr_index)--]) : NULL)
- #define osMemBlockUnsafeFree(mem_part, addr) { (mem_part)->array_addr[++((mem_part)->curr_index)] = addr; }
- #define osMemBlockCountGet(mem_part) ((mem_part)->curr_index + 1)

Functions

- os_mem_part_t * osMemPartCreate (uint32_t block_size, uint32_t num_of_blocks, uint8_t *data_address, uint32_t alignment, uint16_t buffer_offset, os_mem_part_t *mem_part, bool shared)
- INLINE void * osMemBlockGet (os_mem_part_t *mem_part)
- INLINE void osMemBlockFree (os_mem_part_t *mem_part, void *addr)
- os_status osMemLocalHeapSet (os_virt_ptr mem_ptr, uint32_t mem_size, os_mem_type type)
- INLINE void * osMemBlockSyncGet (os_mem_part_t *mem_part)
- INLINE void osMemBlockSyncFree (os_mem_part_t *mem_part, void *addr)

2.4.7.2 Macro Definition Documentation

2.4.7.2.1 #define osMemBlockSize(*mem_part*) ((mem_part)->block_size)

Gets the size of the blocks in the memory structure.

Parameters

in	*mem_part*	- The memory structure to get the block size from.

Returns

The block size.

2.4.7.2.2 #define osMemBlockOffset(*mem_part*) ((mem_part)->buffer_offset)

Gets the lld offset of the blocks in the memory structure.

Parameters

in	*mem_part*	- The memory structure to get the block offset from.

Returns

The block offset.

2.4.7.2.3 #define osMemBlockUnsafeGet(*mem_part*) (((mem_part)->curr_index >= 0) ? ((void *)(mem_part)->array_addr[((mem_part)->curr_index)--]) : NULL)

Gets a block from the memory structure (unsafe version).

Use this function when you can be sure that the memory structure cannot be accessed from any other source.

Parameters

in	*mem_part*	- The memory structure to get the block from.

Returns

The requested memory block.

2.4.7.2.4 #define osMemBlockUnsafeFree(*mem_part, addr*) { (mem_part)->array_addr[++((mem_part)->curr_index)] = addr; }

Releases a block back to the memory structure (unsafe version).

Use this function when you can be sure that the memory structure cannot be accessed from any other source.

Parameters

in	*mem_part*	- The memory structure that owns the block.
in	*addr*	- The block address.

2.4.7.2.5 #define osMemBlockCountGet(*mem_part*) ((mem_part)->curr_index + 1)

Get number of free buffers in memory pool

Parameters

in	*mem_part*	- The memory structure of buffer pool

Returns

number of free buffers

2.4.7.3 Function Documentation

2.4.7.3.1 os_mem_part_t* osMemPartCreate (uint32_t *block_size*, uint32_t *num_of_blocks*, uint8_t * *data_address*, uint32_t *alignment*, uint16_t *buffer_offset*, os_mem_part_t * *mem_part*, bool *shared*)

Initializes a memory structure for aligned, fixed-size blocks.

Parameters

in	*block_size*	- Size of allocatable block.
in	*num_of_blocks*	- Maximum number of allocatable blocks.
in	*data_address*	- Address at which to begin allocation. Use the MEM_PART_DATA_SIZE() macro to determine the required memory size.
in	*alignment*	- Alignment of the blocks (in bytes).
in	*buffer_offset*	- Buffer header allocated to LLD only
in	*mem_part*	- Address of the memory manager space to initialize. Use the MEM_PART_SIZE() macro to determine the required memory size.
in	*shared*	- Flag indicating whether this partition is shared between cores.

Returns

Pointer to the initialized memory structure.

Warning

The allocation size must be larger than (*alignment* - 1)

2.4.7.3.2 INLINE void* osMemBlockGet (os_mem_part_t * *mem_part*)

Gets a block from the memory structure.

Parameters

in	*mem_part*	- The memory structure to get the block from.

Returns

The requested memory block.

2.4.7.3.3 INLINE void osMemBlockFree (os_mem_part_t * *mem_part,* void * *addr*)

Releases a block back to the memory structure.

Parameters

in	*mem_part*	- The memory structure that owns the block.
in	*addr*	- The block address.

2.4.7.3.4 os_status osMemLocalHeapSet (os_virt_ptr *mem_ptr,* uint32_t *mem_size,* os_mem_type *type*)

Updates the entry of the local heap with new values

Function to be used inside appMemHeapsSet which is defined by application and activated inside osInitialize().

Parameters

in	*mem_ptr*	- Virtual start pointer of the memory allocated for this heap
in	*mem_size*	- The size of the memory allocated for this heap
in	*type*	- The name of the heap

Returns

OS_FAIL or OS_SUCCESS

2.4.7.3.5 INLINE void* osMemBlockSyncGet (os_mem_part_t * *mem_part*)

Gets a block from the memory structure (multicore safe).

Parameters

in	*mem_part*	- The memory structure to get the block from.

Returns

The requested memory block.

2.4.7.3.6 INLINE void osMemBlockSyncFree (os_mem_part_t * *mem_part,* void * *addr*)

Releases a block back to the memory structure (multicore safe).

Parameters

in	*mem_part*	- The memory structure that owns the block.
in	*addr*	- The block address.

2.4.8 Caches Control API

2.4.8.1 Overview

Modules

- Data SC3900 Cache API

2.4.8.2 Data SC3900 Cache API

2.4.8.2.1 Overview

SC3900 Cache operations defines, enumerations and API

Data Structures

- struct l2_cache_pf_params_t
- struct cpc_init_params_t
- struct l2_error_info_t
- struct l2_init_params_t
- struct os_cme_maintenance_instruction_params_t
- struct l2_query_params_t
- struct l1_query_params_t
- struct mem_query_params_t
- struct cme_external_query_result_t

Macros

- #define OS_LOCK_L2 0
- #define OS_UNLOCK_L2 1

Typedefs

- typedef void(∗ os_l2_cache_err_isr)(l2_error_info_t ∗)

Enumerations

- enum l2_err_trans_t { L2_TRANS_SNOOP =0, L2_TRANS_WRITE =1, L2_TRANS_READ =2 }

- enum l2_err_trans_source_t {
  L2_ERR_TRANS_SNOOP_EXT = (0), L2_ERR_TRANS_INST_INT = (0b10000), L2_ERR_T↩
  RANS_DATA_INT = (0b110001),
  L2_ERR_TRANS_UNKNOWN }
- enum l2_error_type_t {
  L2_TAG_MULTI_WAY_HIT = (1<<7), L2_TAG_MULTIPLE_BIT_ECC_ERROR = (1<<6),
  L2_TAG_ECC_ERROR = (1<<5),
  L2_DATA_MULTIPLE_BIT_ECC_ERROR = (1<<3), L2_DATA_ECC_ERROR = (1<<2),
  L2_CONFIGURATION_ERROR = (1<<0) }
- enum os_cme_external_query_type {
  CME_DQUERY_L1 = 0x10, CME_DQUERY_L12 = 0x11, CME_DQUERY_MEM = 0x12,
  CME_PQUERY_L1 = 0x30, CME_PQUERY_L12 = 0x31, CME_PQUERY_MEM = 0x32,
  CME_PQSYNC_MEM = 0x3C, CME_PBAR_L1SYNC = 0x3D, CME_PBAR_HWSYNC = 0x3F
  }
- enum os_cme_maintenance_instruction_type {
  CME_DFETCHB_L12 = 0x00, CME_DFETCHB_L2 = 0x01, CME_DFETCHB_LCK_L2 = 0x02,
  CME_DFETCHB_L12_ITW = 0x03, CME_DFETCHB_L2_ITW = 0x04, CME_DFETCHB_LC↩
  K_L2_ITW = 0x05,
  CME_DUNLOCKB_L2 = 0x07, CME_DCMB_SYNC_L1 = 0x08, CME_DCMB_FLUSH_L12 =
  0x0B,
  CME_DCMB_INVAL_L1 = 0x0C, CME_DCMB_INVAL_L12 = 0x0D, CME_PFETCHB_L12 =
  0x20,
  CME_PFETCHB_L2 = 0x21, CME_PFETCHB_LCK_L2 = 0x22, CME_PUNLOCKB_L2 = 0x27,
  CME_PCMB_INVAL_L1 = 0x2C, CME_PCMB_INVAL_L12 = 0x2D }
- enum os_l2_cache_bank_id {
  L2_CACHE_BANK_0 = 0, L2_CACHE_BANK_1 = 1, L2_CACHE_BANK_2 = 2,
  L2_CACHE_BANK_3 = 3 }
- enum os_l2_cache_way_id {
  L2_CACHE_WAY_0 = 0, L2_CACHE_WAY_1 = 1, L2_CACHE_WAY_2 = 2,
  L2_CACHE_WAY_3 = 3, L2_CACHE_WAY_4 = 4, L2_CACHE_WAY_5 = 5,
  L2_CACHE_WAY_6 = 6, L2_CACHE_WAY_7 = 7, L2_CACHE_WAY_8 = 8,
  L2_CACHE_WAY_9 = 9, L2_CACHE_WAY_10 = 10, L2_CACHE_WAY_11 = 11,
  L2_CACHE_WAY_12 = 12, L2_CACHE_WAY_13 = 13, L2_CACHE_WAY_14 = 14,
  L2_CACHE_WAY_15 = 15 }
- enum os_l2_cache_lock_status { L2_CACHE_ADDRESS_UNLOCKED = 0, L2_CACHE_AD↩
  DRESS_LOCKED = 1 }
- enum os_l2_cache_coherency_status {
  L2_CACHE_ADDRESS_INVALID = 0, L2_CACHE_ADDRESS_SHARED = 1, L2_CACHE_↩
  ADDRESS_EXCLUSIVE = 2,
  L2_CACHE_ADDRESS_MODIFIED = 3 }
- enum os_l1_cache_way_id {
  L1_CACHE_WAY_0 = 0, L1_CACHE_WAY_1 = 1, L1_CACHE_WAY_2 = 2,
  L1_CACHE_WAY_3 = 3, L1_CACHE_WAY_4 = 4, L1_CACHE_WAY_5 = 5,
  L1_CACHE_WAY_6 = 6, L1_CACHE_WAY_7 = 7 }
- enum os_l1_cache_hit_indication { L1_CACHE_ADDRESS_MISS = 0, L1_CACHE_ADDRES↩
  S_HIT = 1 }
- enum os_sgb_hit_indication { SGB_ADDRESS_MISS = 0, SGB_ADDRESS_HIT = 1 }

- enum os_l1_cache_edc_error_status { L1_CACHE_NO_EDC_ERROR = 0, L1_CACHE_EDC_↩ ERROR_WAS_DETECTED = 1 }
- enum os_mem_L1_ext_err { L1_NO_EXTER_MEM_ERROR = 0, L1_EXTER_MEM_ERROR↩ _WAS_DETECTED = 1 }
- enum os_mem_cacheable_descriptor_hit { CACHEABLE_DESCRIPTOR_HIT = 0, NO_CACH↩ EABLE_DESCRIPTOR_HIT = 1 }
- enum os_mem_bank_0_indication { EXTERNAL_ACCESS = 0, BANK_0_ACCESS = 1 }
- enum os_mem_mmu_error_status { MMUE_MULTIPLE_SEGMENT_HIT = 0, MMUE_SEGMENT_MISS_ERROR = 1, MMUE_↩ PERIPHERAL_ERROR = 3, MMUE_SEGMENT_MISS_WHEN_PROTECTION_DISABLED = 4, QUERY_HAS_NO_MM↩ U_ERROR_REPORTED = 7 }

Functions

- os_status osCmeExtQuery (uint8_t task_id, os_const_virt_ptr virt_addr, os_cme_external_query_↩ type cme_external_query, cme_external_query_result_t *cme_external_query_result)
- os_status osCmeMaintenanceInstruction (os_cme_maintenance_instruction_params_t cme_↩ maintenance_instruction_params, uint8_t *channel)
- uint8_t osCmeIsDataMaintenanceInstructionInProgress (uint8_t channel_mask)
- uint8_t osCmeIsProgMaintenanceInstructionInProgress (uint8_t channel_mask)
- os_status osCacheStoreBarrier (uint32_t cache_policy)
- os_status osCacheLoadBarrier (uint32_t cache_policy)
- os_status osCacheL2DataLock (l2_cache_pf_params_t params, bool operation)
- os_status osCacheL2ProgLock (l2_cache_pf_params_t params, bool operation)
- os_status osCacheL2SetPartitioning (uint8_t Partition_Id, uint32_t allocation_mask, uint32_t way↩ _mask)
- uint32_t osCacheL2GetDefaultPartitionWayMask ()
- os_status osCacheL2Query (os_const_virt_ptr virt_addr, l2_query_params_t *l2_query_params)
- os_status osCacheL2GlobalLockClear ()
- void osCacheL2ClearPartition ()
- os_status osCacheL2FastGlobalFlush ()
- os_status osSpecificCacheL2GlobalFlush (uint8_t cluster_num)
- os_status osCacheL2GlobalFlush ()

Cache Policy Options

- #define NOT_CACHED 0x00000000
- #define L1_CACHED 0x00000001
- #define L2_CACHED 0x00000002
- #define L1_L2_CACHED (L1_CACHED | L2_CACHED)

L2 Cache partition Options

- #define OS_L2PAR_STALLOC 0x00000001
- #define OS_L2PAR_DRDALLOC 0x00000040
- #define OS_L2PAR_IRDALLOC 0x00000080
- #define OS_L2PAR_DSTALLOC 0x00000400
- #define OS_L2PAR_DATA_DEFAULT (OS_L2PAR_STALLOC | OS_L2PAR_DRDALLOC | OS_L2PAR_DSTALLOC)

- #define OS_L2PAR_PROG_DEFAULT (OS_L2PAR_STALLOC | OS_L2PAR_IRDALLOC)

2.4.8.2.2 Data Structure Documentation

2.4.8.2.2.1 struct l2_cache_pf_params_t

L2 Cache prefetch configuration parameters.

Data Fields

- uint32_t start_addr
- uint32_t buffer_size

2.4.8.2.2.1.1 Field Documentation

2.4.8.2.2.1.2 uint32_t l2_cache_pf_params_t::start_addr

Start address of prefetch 128-byte resolution.

2.4.8.2.2.1.3 uint32_t l2_cache_pf_params_t::buffer_size

Buffer size 128-Byte resolution.

2.4.8.2.2.2 struct cpc_init_params_t

CPC configuration parameters.

Data Fields

- uint32_t os_l3_enable:1
- uint32_t os_m3_enable:1
- uint32_t __pad0__:30

2.4.8.2.2.2.1 Field Documentation

2.4.8.2.2.2.2 uint32_t cpc_init_params_t::os_l3_enable

Enables/disables L3 cache.

2.4.8.2.2.2.3 uint32_t cpc_init_params_t::os_m3_enable

Enables/disables SRAM mode.

2.4.8.2.2.2.4 uint32_t cpc_init_params_t::__pad0__

Reserved for future development.

2.4.8.2.2.3 struct l2_error_info_t

L2 cache error inforamtion data structure.

Data Fields

- l2_error_type_t l2_error_type
- bool multi_l2_errors
- os_phys_ptr err_capture_addr
- uint64_t err_capture_data
- uint32_t ecc_syndrome
- uint8_t core_id
- l2_err_trans_source_t trans_source
- l2_err_trans_t trans_type
- uint8_t double_word_num
- uint8_t tag_ecc_error_count
- uint8_t data_ecc_error_count

2.4.8.2.2.3.1 Field Documentation

2.4.8.2.2.3.2 l2_error_type_t l2_error_info_t::l2_error_type

Error detected type.

2.4.8.2.2.3.3 bool l2_error_info_t::multi_l2_errors

Multiple L2 errors of the same type were detected.

2.4.8.2.2.3.4 os_phys_ptr l2_error_info_t::err_capture_addr

Provides the real address of a captured error detected in the L2 cache of the processor.

2.4.8.2.2.3.5 uint64_t l2_error_info_t::err_capture_data

Provides the array data of a captured error detected in the L2 cache of the processor.

2.4.8.2.2.3.6 uint32_t l2_error_info_t::ecc_syndrome

provides both the calculated and stored ECC syndrome of a captured error detected in the L2 cache of the processor.

2.4.8.2.2.3.7 uint8_t l2_error_info_t::core_id

Core ID that issued TRANSTYPE.

If the transaction was from a snoop, this field is undefined.

2.4.8.2.2.3.8 l2_err_trans_source_t l2_error_info_t::trans_source

Transaction source for detected error.

2.4.8.2.2.3.9 l2_err_trans_t l2_error_info_t::trans_type

Transaction type for detected error.

2.4.8.2.2.3.10 uint8_t l2_error_info_t::double_word_num

For data ECC errors, contains the double-word number of the detected error.

For tag/status ECC errors, contains which way of the tag/status encountered the error.

2.4.8.2.2.3.11 uint8_t l2_error_info_t::tag_ecc_error_count

Number of ECC single-bit errors in the L2 tags detected.

2.4.8.2.2.3.12 uint8_t l2_error_info_t::data_ecc_error_count

Number of ECC single-bit errors in the L2 data detected.

2.4.8.2.2.4 struct l2_init_params_t

L2 cache initialization data structure.

Data Fields

- uint32_t os_l2_enable:1
- uint32_t os_l2_partition_enable:1
- uint32_t os_l2_lock_enable:1
- uint32_t os_l2_err_report_enable:1
- uint32_t os_l2_err_threshold:8
- uint32_t tag_multi_way_hit_int_en:1
- uint32_t tag_multiple_bit_ecc_err_int_en:1
- uint32_t tag_ecc_error_int_en:1
- uint32_t data_multiple_bit_ecc_err_int_en:1
- uint32_t data_ecc_err_int_en:1
- uint32_t l2_configuration_err_int_en:1
- uint32_t __pad0__:14
- os_l2_cache_err_isr l2_err_isr

2.4.8.2.2.4.1 Field Documentation

2.4.8.2.2.4.2 uint32_t l2_init_params_t::os_l2_enable

Enables/disables L2 cache.

2.4.8.2.2.4.3 uint32_t l2_init_params_t::os_l2_partition_enable

DEPRECATED.

2.4.8.2.2.4.4 uint32_t l2_init_params_t::os_l2_lock_enable

DEPRECATED.

2.4.8.2.2.4.5 uint32_t l2_init_params_t::os_l2_err_report_enable

Enables/disables L2 report.

2.4.8.2.2.4.6 uint32_t l2_init_params_t::os_l2_err_threshold

L2 cache threshold value for the number of ECC single-bit errors that are detected before reporting an error condition.

2.4.8.2.2.4.7 uint32_t l2_init_params_t::tag_multi_way_hit_int_en

Enables/disables Tag multi-way hit interrupt reporting.

2.4.8.2.2.4.8 uint32_t l2_init_params_t::tag_multiple_bit_ecc_err_int_en

Enables/disables Tag Multiple-bit ECC error interrupt reporting.

2.4.8.2.2.4.9 uint32_t l2_init_params_t::tag_ecc_error_int_en

Enables/disables Tag ECC interrupt reporting.

2.4.8.2.2.4.10 uint32_t l2_init_params_t::data_multiple_bit_ecc_err_int_en

Enables/disables Data Multiple-bit ECC error interrupt reporting.

2.4.8.2.2.4.11 uint32_t l2_init_params_t::data_ecc_err_int_en

Enables/disables Data ECC error interrupt reporting.

2.4.8.2.2.4.12 uint32_t l2_init_params_t::l2_configuration_err_int_en

Enables/disables L2 configuration error interrupt reporting.

2.4.8.2.2.4.13 uint32_t l2_init_params_t::__pad0__

Reserved for future development.

2.4.8.2.2.4.14 os_l2_cache_err_isr l2_init_params_t::l2_err_isr

L2 cache error ISR.

2.4.8.2.2.5 struct os_cme_maintenance_instruction_params_t

cache maintenance instruction parameters

Data Fields

- uint8_t task_id
- os_const_virt_ptr virt_addr
- uint32_t block_size
- os_cme_maintenance_instruction_type maintenance_instruction_type

2.4.8.2.2.5.1 Field Documentation

2.4.8.2.2.5.2 uint8_t os_cme_maintenance_instruction_params_t::task_id

Task ID of the cache instruction.

2.4.8.2.2.5.3 os_const_virt_ptr os_cme_maintenance_instruction_params_t::virt_addr

Maintenance instruction start address.

2.4.8.2.2.5.4 uint32_t os_cme_maintenance_instruction_params_t::block_size

Block size 128-Byte resolution.

2.4.8.2.2.5.5 os_cme_maintenance_instruction_type os_cme_maintenance_instruction_params_t::maintenance_instruction_type

The instruction type to perform.

2.4.8.2.2.6 struct l2_query_params_t

L2 cache Query command returned parameters.

Data Fields

- os_l2_cache_bank_id l2_cache_bank_id
- os_l2_cache_way_id l2_cache_way_id
- os_l2_cache_lock_status l2_cache_lock_status
- os_l2_cache_coherency_status l2_cache_coherency_status

2.4.8.2.2.6.1 Field Documentation

2.4.8.2.2.6.2 os_l2_cache_bank_id l2_query_params_t::l2_cache_bank_id

Reflects the L2 cache bank for the related query address.

2.4.8.2.2.6.3 os_l2_cache_way_id l2_query_params_t::l2_cache_way_id

Reflects the L2 cache way for the related query address.

2.4.8.2.2.6.4 os_l2_cache_lock_status l2_query_params_t::l2_cache_lock_status

Reflects the L2 cache lock status for the related query address.

2.4.8.2.2.6.5 os_l2_cache_coherency_status l2_query_params_t::l2_cache_coherency_status

Reflects the L2 cache hit and coherency status for the related query address.

2.4.8.2.2.7 struct l1_query_params_t

L1 cache Query command returned parameters.

Data Fields

- os_l1_cache_way_id l1_cache_way_id
- os_l1_cache_hit_indication l1_cache_hit_indication
- os_sgb_hit_indication sgb_hit_indication
- os_l1_cache_edc_error_status l1_cache_edc_error_status

2.4.8.2.2.7.1 Field Documentation

2.4.8.2.2.7.2 os_l1_cache_way_id l1_query_params_t::l1_cache_way_id

Reflects the L1 cache way for the related query address.

2.4.8.2.2.7.3 os_l1_cache_hit_indication l1_query_params_t::l1_cache_hit_indication

indication for L1 Hit

2.4.8.2.2.7.4 os_sgb_hit_indication l1_query_params_t::sgb_hit_indication

indication for SGB Hit

2.4.8.2.2.7.5 os_l1_cache_edc_error_status l1_query_params_t::l1_cache_edc_error_status

L1 EDC error status.

2.4.8.2.2.8 struct mem_query_params_t

MEM cache Query command returned parameters.

Data Fields

- os_mem_L1_ext_err L1_ext_err
- os_mem_cacheable_descriptor_hit cacheable_descriptor_hit
- os_mem_bank_0_indication bank_0_indication
- os_mem_mmu_error_status mmue_status
- uint8_t Segment_descriptor_index

2.4.8.2.2.8.1 Field Documentation

2.4.8.2.2.8.2 os_mem_L1_ext_err mem_query_params_t::L1_ext_err

Reflects the MEM cache way for the related query address.

2.4.8.2.2.8.3 os_mem_cacheable_descriptor_hit mem_query_params_t::cacheable_descriptor_hit

indication for MEM Hit

2.4.8.2.2.8.4 os_mem_bank_0_indication mem_query_params_t::bank_0_indication

MEM EDC error status.

2.4.8.2.2.8.5 os_mem_mmu_error_status mem_query_params_t::mmue_status

mmu error status

2.4.8.2.2.8.6 uint8_t mem_query_params_t::Segment_descriptor_index

Reflects the segment descriptor hit index for the related query addresss.

2.4.8.2.2.9 struct cme_external_query_result_t

CME cache Query command returned parameters.

Data Fields

- os_phys_ptr query_physical_address
- l2_query_params_t l2_query_params
- l1_query_params_t l1_query_params
- mem_query_params_t mem_query_params

2.4.8.2.2.9.1 Field Documentation

2.4.8.2.2.9.2 os_phys_ptr cme_external_query_result_t::query_physical_address

Query physical address.

In case of memory query holds 32 bit of data read from the query address

2.4.8.2.2.9.3 l2_query_params_t cme_external_query_result_t::l2_query_params

Reflects the L2 cache query result.

2.4.8.2.2.9.4 l1_query_params_t cme_external_query_result_t::l1_query_params

Reflects the L1 cache query result.

2.4.8.2.2.9.5 mem_query_params_t cme_external_query_result_t::mem_query_params

Reflects the memory query result.

2.4.8.2.3 Macro Definition Documentation

2.4.8.2.3.1 #define OS_LOCK_L2 0

L2 lock operation.

2.4.8.2.3.2 #define OS_UNLOCK_L2 1

L2 unlock operation.

2.4.8.2.3.3 #define NOT_CACHED 0x00000000

Not cacheable in any level of cache.

2.4.8.2.3.4 #define L1_CACHED 0x00000001

Cacheable in L1 cache only.

2.4.8.2.3.5 #define L2_CACHED 0x00000002

Cacheable in L2 cache only.

2.4.8.2.3.6 #define L1_L2_CACHED (L1_CACHED | L2_CACHED)

Cacheable in L1 and L2 cache.

2.4.8.2.3.7 #define OS_L2PAR_STALLOC 0x00000001

Stashing allocation control.

2.4.8.2.3.8 #define OS_L2PAR_DRDALLOC 0x00000040

Data read allocation control.

2.4.8.2.3.9 #define OS_L2PAR_IRDALLOC 0x00000080

Instruction read and fetch allocation control.

2.4.8.2.3.10 #define OS_L2PAR_DSTALLOC 0x00000400

Data store allocation control.

2.4.8.2.3.11 #define OS_L2PAR_DATA_DEFAULT (OS_L2PAR_STALLOC | OS_L2PAR_DRDALLOC | OS_L2PAR_DSTALLOC)

Data store & read allocation with Stashing allocation.

2.4.8.2.3.12 #define OS_L2PAR_PROG_DEFAULT (OS_L2PAR_STALLOC | OS_L2PAR_IRDALLOC)

program read & Stashing allocation

2.4.8.2.4 Typedef Documentation

2.4.8.2.4.1 typedef void(* os_l2_cache_err_isr)(l2_error_info_t *)

L2 cache error ISR type.

2.4.8.2.5 Enumeration Type Documentation

2.4.8.2.5.1 enum l2_err_trans_t

L2 cache error transaction type.

Enumerator

L2_TRANS_SNOOP snoop transaction type
L2_TRANS_WRITE write transaction type
L2_TRANS_READ read transaction type

2.4.8.2.5.2 enum l2_err_trans_source_t

L2 cache error type.

Enumerator

L2_ERR_TRANS_SNOOP_EXT Tag multi-way hit detected.
L2_ERR_TRANS_INST_INT Tag Multiple-bit ECC error detected.
L2_ERR_TRANS_DATA_INT Tag ECC error detected.
L2_ERR_TRANS_UNKNOWN Data Multiple-bit ECC error detected.

2.4.8.2.5.3 enum l2_error_type_t

L2 cache error type.

Enumerator

L2_TAG_MULTI_WAY_HIT Tag multi-way hit detected.
L2_TAG_MULTIPLE_BIT_ECC_ERROR Tag Multiple-bit ECC error detected.
L2_TAG_ECC_ERROR Tag ECC error detected.
L2_DATA_MULTIPLE_BIT_ECC_ERROR Data Multiple-bit ECC error detected.
L2_DATA_ECC_ERROR Data ECC error detected.
L2_CONFIGURATION_ERROR L2 configuration error detected.

2.4.8.2.5.4 enum os_cme_external_query_type

Type of cache external query instruction.

Enumerator

CME_DQUERY_L1 Queries the MMU and L1 data cache.
CME_DQUERY_L12 Queries the MMU, L1 data cache and L2 cache.
CME_DQUERY_MEM Data query for memory (DDR)
CME_PQUERY_L1 Program query for L1 cache.

CME_PQUERY_L12 Program query for both L1 & L2 Caches.
CME_PQUERY_MEM Program query for memory (DDR)
CME_PQSYNC_MEM global ICache invalidation
CME_PBAR_L1SYNC clearance of ICache internal FIFOs
CME_PBAR_HWSYNC HWSYNC through ICache bus.

2.4.8.2.5.5 enum os_cme_maintenance_instruction_type

Type of cache maintenance instruction.

Enumerator

CME_DFETCHB_L12 Fetches the specified cache granule to the L2 cache and L1 data cache.
CME_DFETCHB_L2 Fetches the specified cache granule to the L2 cache.
CME_DFETCHB_LCK_L2 Fetches the specified block of cache granules to the L2 cache and locks them there.
CME_DFETCHB_L12_ITW Fetches the specified cache granule to the L2 cache and L1 data cache with intend to write attribute.
CME_DFETCHB_L2_ITW Fetches the specified cache granule to the L2 cache with intend to write attribute.
CME_DFETCHB_LCK_L2_ITW Fetches the specified block of cache granules to the L2 cache and locks them there with intend to write attribute.
CME_DUNLOCKB_L2 Unlocks the specified block of cache granules in the L2 cache.
CME_DCMB_SYNC_L1 Synchronizes the L1 data cache granules with the L2 cache. Flushes SGB

CME_DCMB_FLUSH_L12 Invalidates the specific block of L1 data cache granules and flushes this block from L2 cache to the memory. Does not flush SGB.
CME_DCMB_INVAL_L1 Invalidates the specific block of L1 data cache granules. Does not flush SGB.
CME_DCMB_INVAL_L12 Invalidates the specific block of L1 data cache and L2 cache granules. Does not flush SGB.
CME_PFETCHB_L12 Fetches the specified cache granule to the L2 cache and L1 program cache.
CME_PFETCHB_L2 Fetches the specified cache granule to the L2 cache.
CME_PFETCHB_LCK_L2 Fetches the specified block of cache granules to the L2 cache and locks them there.
CME_PUNLOCKB_L2 Unlocks the specified block of cache granules in the L2 cache.
CME_PCMB_INVAL_L1 Invalidates the specific block of L1 program cache granules.
CME_PCMB_INVAL_L12 Invalidates the specific block of L1 program cache and L2 cache granules.

2.4.8.2.5.6 enum os_l2_cache_bank_id

Enumeration of L2 cache Bank numbers.

Enumerator

L2_CACHE_BANK_0 L2 Cache bank 0.

L2_CACHE_BANK_1 L2 Cache bank 1.
L2_CACHE_BANK_2 L2 Cache bank 2.
L2_CACHE_BANK_3 L2 Cache bank 3.

2.4.8.2.5.7 enum os_l2_cache_way_id

Enumeration of L2 cache way numbers.

Enumerator

L2_CACHE_WAY_0 L2 Cache way 0.
L2_CACHE_WAY_1 L2 Cache way 1.
L2_CACHE_WAY_2 L2 Cache way 2.
L2_CACHE_WAY_3 L2 Cache way 3.
L2_CACHE_WAY_4 L2 Cache way 4.
L2_CACHE_WAY_5 L2 Cache way 5.
L2_CACHE_WAY_6 L2 Cache way 6.
L2_CACHE_WAY_7 L2 Cache way 7.
L2_CACHE_WAY_8 L2 Cache way 8.
L2_CACHE_WAY_9 L2 Cache way 9.
L2_CACHE_WAY_10 L2 Cache way 10.
L2_CACHE_WAY_11 L2 Cache way 11.
L2_CACHE_WAY_12 L2 Cache way 12.
L2_CACHE_WAY_13 L2 Cache way 13.
L2_CACHE_WAY_14 L2 Cache way 14.
L2_CACHE_WAY_15 L2 Cache way 15.

2.4.8.2.5.8 enum os_l2_cache_lock_status

Enumeration of L2 Lock status.

Enumerator

L2_CACHE_ADDRESS_UNLOCKED Address is not Locked in the L2 Cache.
L2_CACHE_ADDRESS_LOCKED Address is Locked in the L2 Cache.

2.4.8.2.5.9 enum os_l2_cache_coherency_status

Enumeration of L2 coherency status.

Enumerator

L2_CACHE_ADDRESS_INVALID Address is invalid in the L2 Cache.
L2_CACHE_ADDRESS_SHARED Address is shared between more then one L2 cache.
L2_CACHE_ADDRESS_EXCLUSIVE Address is valid in the only in this L2 Cache.
L2_CACHE_ADDRESS_MODIFIED Address is dirty in the L2 Cache.

2.4.8.2.5.10 enum os_l1_cache_way_id

Enumeration of L1 cache way numbers.

Enumerator

L1_CACHE_WAY_0 L1 Cache way 0.
L1_CACHE_WAY_1 L1 Cache way 1.
L1_CACHE_WAY_2 L1 Cache way 2.
L1_CACHE_WAY_3 L1 Cache way 3.
L1_CACHE_WAY_4 L1 Cache way 4.
L1_CACHE_WAY_5 L1 Cache way 5.
L1_CACHE_WAY_6 L1 Cache way 6.
L1_CACHE_WAY_7 L1 Cache way 7.

2.4.8.2.5.11 enum os_l1_cache_hit_indication

Enumeration of L1 Hit indication.

Enumerator

L1_CACHE_ADDRESS_MISS Address is not in the L1 Cache.
L1_CACHE_ADDRESS_HIT Address is in the L1 Cache.

2.4.8.2.5.12 enum os_sgb_hit_indication

Enumeration of SGB Hit indication.

Enumerator

SGB_ADDRESS_MISS Address is not in the SGB Cache.
SGB_ADDRESS_HIT Address is in the SGB Cache.

2.4.8.2.5.13 enum os_l1_cache_edc_error_status

Enumeration of L1 EDC error status.

Enumerator

L1_CACHE_NO_EDC_ERROR No EDC error in L1.
L1_CACHE_EDC_ERROR_WAS_DETECTED L1 EDC error was detected. Returned data is not corrected

2.4.8.2.5.14 enum os_mem_L1_ext_err

Enumeration of L1 external memory error indication.

Enumerator

L1_NO_EXTER_MEM_ERROR No external memory error in L1.
L1_EXTER_MEM_ERROR_WAS_DETECTED External memory error was detected.

2.4.8.2.5.15 enum os_mem_cacheable_descriptor_hit

Enumeration of Non cacheable descriptor hit indication.

Enumerator

CACHEABLE_DESCRIPTOR_HIT Cacheable descriptor hit.
NO_CACHEABLE_DESCRIPTOR_HIT Non cacheable descriptor hit.

2.4.8.2.5.16 enum os_mem_bank_0_indication

Enumeration of Bank 0 indication.

Enumerator

EXTERNAL_ACCESS External access that goes through the L2 cache.
BANK_0_ACCESS Bank 0 access.

2.4.8.2.5.17 enum os_mem_mmu_error_status

Enumeration of MMU error status.

Enumerator

MMUE_MULTIPLE_SEGMENT_HIT Query has multiple segment descriptor hit error reported by MMU.
MMUE_SEGMENT_MISS_ERROR Query has segment miss error reported by MMU.
MMUE_PERIPHERAL_ERROR Query has peripheral error reported by MMU. DQUERY.L12 accesses bank 0 descriptor
MMUE_SEGMENT_MISS_WHEN_PROTECTION_DISABLED Query detected segment miss when protection is disabled.
QUERY_HAS_NO_MMU_ERROR_REPORTED Query has no errors reported by MMU.

2.4.8.2.6 Function Documentation

2.4.8.2.6.1 os_status osCmeExtQuery (uint8_t *task_id,* os_const_virt_ptr *virt_addr,* os_cme_external_query_type *cme_external_query,* cme_external_query_result_t * *cme_external_query_result*)

Returns the virtual address cache parameters.

Parameters

in	*task_id*	- Task ID of the cache instruction - Valid for query instructions.
in	*virt_addr*	- virtual address to verify its cache parameters - Valid for query instructions..
in	*cme_external_query*	- The type of external query to perform.
out	*cme_external_query_result*	- Holds the result of the query operation - Valid for query instructions.

Return values

OS_SUCCESS	

2.4.8.2.6.2 os_status osCmeMaintenanceInstruction (os_cme_maintenance_instruction_params_t *cme_maintenance_instruction_params,* uint8_t * *channel*)

Perform CME maintenance instruction.

Parameters

in	*cme_maintenance_instruction_params*	- Holds the l2 cache parameters of the virtual address.
out	*channel*	- Index of CME channel allocated for core cache maintenance instruction.

Return values

OS_SUCCESS	

2.4.8.2.6.3 uint8_t osCmeIsDataMaintenanceInstructionInProgress (uint8_t *channel_mask*)

returns if CME maintenance instruction is is still in progress.

Parameters

in	*channel_mask*	- channels(bit mask) to query about- for all channels (0xff) .

Return values

channel	mask: '1' - Channels are in working state (active). '0' - Channels are in empty state (idle).

2.4.8.2.6.4 uint8_t osCmeIsProgMaintenanceInstructionInProgress (uint8_t *channel_mask*)

returns if CME maintenance instruction is still in progress.

Parameters

in	*channel_mask*	- channels(bit mask) to query about- for all channels (0xff) .

Return values

channel	mask: '1' - Channels are in working state (active). '0' - Channels are in empty state (idle).

2.4.8.2.6.5 os_status osCacheStoreBarrier (uint32_t *cache_policy*)

Perform store barrier between two stores action to ensure chronological order.

Parameters

in	*cache_policy*	- Cache policy of the application. when cache policy is unknown it is recommended to use NOT_CACHED. options: NOT_CACHED Not cacheable in any level of cache. L1_CACHED Order is important only in L1 cache. L2_CACHED Cacheable in L2 cache only L1_L2_CAC↩ HED Cacheable in L1 and L2 cache

Return values

OS_SUCCESS	

2.4.8.2.6.6 os_status osCacheLoadBarrier (uint32_t *cache_policy*)

Perform Load barrier between two loads action to ensure chronological order.

Parameters

in	*cache_policy*	- Cache policy of the application. when cache policy is unknown it is recommended to use NOT_CACHED. options: NOT_CACHE↩ D Not cacheable in any level of cache. L1_CACHED Cacheable in L1 cache only L2_CACHED Cacheable in L2 cache only L1_L2_CACH↩ ED Cacheable in L1 and L2 cache

Return values

OS_SUCCESS	

2.4.8.2.6.7 os_status osCacheL2DataLock (l2_cache_pf_params_t *params*, bool *operation*)

Perform prefetch of data to l2 cache and lock/unlock the prefetch mechanism (M2).

Parameters

in	*params*	- L2 Cache prefetch configuration parameters
in	*operation*	- OS_LOCK_L2/OS_UNLOCK_L2 commands.

Return values

OS_SUCCESS	- The return value indicate a successful allocation of a CME channel.

&cautions This function does not intend to ensure that the CME command has been executed. To ensure that the Data has been locked in the L2 cache you can use osCacheL2Query() on the same address which indicate if this address has been locked in the L2 cache.

2.4.8.2.6.8 os_status osCacheL2ProgLock (l2_cache_pf_params_t *params*, bool *operation*)

Perform prefetch of Program to l2 cache and lock/unlock the prefetch mechanism (M2).

Parameters

in	*params*	- L2 Cache prefetch configuration parameters
in	*operation*	- LOCK_L2/UNLOCK_L2 commands.

Return values

OS_SUCCESS	- The return value indicate a successful allocation of a CME channel.

2.4.8.2.6.9 os_status osCacheL2SetPartitioning (uint8_t *Partition_Id*, uint32_t *allocation_mask*, uint32_t *way_mask*)

Set partition configuration for selected partition identifier.

Parameters

in	*Partition_Id*	- L2 Cache partition identifier.

in	*allocation_mask*	- The allocation possibilities bit-mask for the selected partition identifier.
in	*way_mask*	- L2 Cache Way-mask for the selected partition identifier.

Return values

OS_SUCCESS	

2.4.8.2.6.10 uint32_t osCacheL2GetDefaultPartitionWayMask ( )

Get the currently default partition Way configuration.

Return values

L2	Cache Way-mask of the default partition slot.

2.4.8.2.6.11 os_status osCacheL2Query (os_const_virt_ptr *virt_addr,* l2_query_params_t * *l2_query_params*)

Returns the virtual address L2 cache parameters.

Parameters

in	*virt_addr*	- virtual address to verify its cache parameters.
out	*l2_query_params*	- Holds the l2 cache parameters of the virtual address.

Return values

OS_SUCCESS	

2.4.8.2.6.12 os_status osCacheL2GlobalLockClear ( )

clears lock bits regardless of cache enablement.

Return values

OS_SUCCESS	

2.4.8.2.6.13 void osCacheL2ClearPartition ( )

Restores L2 cache partition policy to default.

Return values

None	

2.4.8.2.6.14 os_status osCacheL2FastGlobalFlush ( )

causes all coherency granules in the modified state to be written back to backing store and all L2 entries changed to the invalid state. Clears the Cache partitions settings.

Return values

OS_SUCCESS	

2.4.8.2.6.15 os_status osSpecificCacheL2GlobalFlush (uint8_t *cluster_num*)

in a given cluster, causes all coherency granules in the modified state to be written back to backing store and all L2 entries changed to the invalid state.

Parameters

in	*cluster_num*	- cluster number for which L2 cache will be flushed .

Return values

OS_SUCCESS	

2.4.8.2.6.16 os_status osCacheL2GlobalFlush ( )

causes all coherency granules in the modified state to be written back to backing store and all L2 entries changed to the invalid state.

Return values

OS_SUCCESS	

2.5 Interrupts

2.5.1 Overview

Modules

- Software Interrupts API
- Hardware Interrupts API
- Virtual Interrupts API

2.5.2 Software Interrupts API

2.5.2.1 Overview

```
Software Interrupts setup and control.
```

Modules

- Software Interrupts Runtime

2.5.2.2 Software Interrupts Runtime

2.5.2.2.1 Overview

Functions

- os_status osSwiCreate (os_swi_function handler, os_swi_handle swi_num, os_swi_priority priority, os_user_id user_id)
- os_status osSwiDelete (os_swi_handle swi_num)
- os_status osSwiActivate (os_swi_handle swi_num)
- void osSwiEnable (os_swi_priority priority)
- os_swi_priority osSwiDisable ()
- os_status osSwiCountInc (os_swi_handle swi_num)
- os_status osSwiCountDec (os_swi_handle swi_num)
- os_status osSwiCountGet (os_swi_handle swi_num, os_swi_count *count)
- os_status osSwiCountSet (os_swi_handle swi_num, os_swi_count count)
- os_status osSwiPriorityGet (os_swi_handle swi_num, os_swi_priority *priority)
- os_status osSwiPrioritySet (os_swi_handle swi_num, os_swi_priority priority)
- os_swi_status osSwiStatus (os_swi_handle swi_num)
- os_status osSwiFind (os_swi_handle *swi_num)
- os_status osSwiSelf (os_swi_handle *swi_num)

SWI Objects values

```
SWI objects can be created dynamically by using osSwiFind() or
they can be statically referenced by the definitions below.
```

- #define OS_SWI0 0
- #define OS_SWI1 1
- #define OS_SWI2 2
- #define OS_SWI3 3
- #define OS_SWI4 4
- #define OS_SWI5 5
- #define OS_SWI6 6
- #define OS_SWI7 7
- #define OS_SWI8 8
- #define OS_SWI9 9
- #define OS_SWI10 10
- #define OS_TIMER_SWI OS_SWI0

SWI Priorities

- #define OS_SWI_PRIORITY0 0
- #define OS_SWI_PRIORITY1 1
- #define OS_SWI_PRIORITY2 2
- #define OS_SWI_PRIORITY3 3
- #define OS_SWI_PRIORITY4 4
- #define OS_SWI_PRIORITY5 5
- #define OS_SWI_PRIORITY6 6
- #define OS_SWI_PRIORITY7 7
- #define OS_SWI_PRIORITY8 8
- #define OS_SWI_PRIORITY9 9
- #define OS_SWI_PRIORITY10 10
- #define OS_SWI_PRIORITY11 11
- #define OS_SWI_PRIORITY12 12
- #define OS_SWI_PRIORITY13 13
- #define OS_SWI_PRIORITY14 14
- #define OS_SWI_PRIORITY15 15
- #define OS_SWI_PRIORITY_ALL 16
- #define OS_SWI_PRIORITY_NONE 254

SWI Status values (of os_swi_status)

- #define OS_SWI_UNUSED 0
- #define OS_SWI_DELETED 1
- #define OS_SWI_ACQUIRED 2
- #define OS_SWI_CREATED 3
- #define OS_SWI_READY 4

2.5.2.2.2 Macro Definition Documentation

2.5.2.2.2.1 #define OS_SWI0 0

Reserved for OS Tick functionality.

2.5.2.2.2.2 #define OS_SWI1 1

software interrupt number 1

2.5.2.2.2.3 #define OS_SWI2 2

software interrupt number 2

2.5.2.2.2.4 #define OS_SWI3 3

software interrupt number 3

2.5.2.2.2.5 #define OS_SWI4 4

software interrupt number 4

2.5.2.2.2.6 #define OS_SWI5 5

software interrupt number 5

2.5.2.2.2.7 #define OS_SWI6 6

software interrupt number 6

2.5.2.2.2.8 #define OS_SWI7 7

software interrupt number 7

2.5.2.2.2.9 #define OS_SWI8 8

software interrupt number 8

2.5.2.2.2.10 #define OS_SWI9 9

software interrupt number 9

2.5.2.2.2.11 #define OS_SWI10 10

software interrupt number 10

2.5.2.2.2.12 #define OS_TIMER_SWI OS_SWI0

Equals OS_SWI0 - reserved for OS Tick functionality.

2.5.2.2.2.13 #define OS_SWI_PRIORITY0 0

Highest SWI priority.

2.5.2.2.2.14 #define OS_SWI_PRIORITY1 1

software intterrupt priority 1

2.5.2.2.2.15 #define OS_SWI_PRIORITY2 2

software intterrupt priority 2

2.5.2.2.2.16 #define OS_SWI_PRIORITY3 3

software intterrupt priority 3

2.5.2.2.2.17 #define OS_SWI_PRIORITY4 4

software intterrupt priority 4

2.5.2.2.2.18 #define OS_SWI_PRIORITY5 5

software intterrupt priority 5

2.5.2.2.2.19 #define OS_SWI_PRIORITY6 6

software intterrupt priority 6

2.5.2.2.2.20 #define OS_SWI_PRIORITY7 7

software intterrupt priority 7

2.5.2.2.2.21 #define OS_SWI_PRIORITY8 8

software intterrupt priority 8

2.5.2.2.2.22 #define OS_SWI_PRIORITY9 9

software intterrupt priority 9

2.5.2.2.2.23 #define OS_SWI_PRIORITY10 10

software intterrupt priority 10

2.5.2.2.2.24 #define OS_SWI_PRIORITY11 11

software intterrupt priority 11

2.5.2.2.2.25 #define OS_SWI_PRIORITY12 12

software intterrupt priority 12

2.5.2.2.2.26 #define OS_SWI_PRIORITY13 13

software intterrupt priority 13

2.5.2.2.2.27 #define OS_SWI_PRIORITY14 14

software intterrupt priority 14

2.5.2.2.2.28 #define OS_SWI_PRIORITY15 15

software intterrupt priority 15

2.5.2.2.2.29 #define OS_SWI_PRIORITY_ALL 16

software intterrupt priority 16

2.5.2.2.2.30 #define OS_SWI_PRIORITY_NONE 254

Disabled SWI priority.

2.5.2.2.2.31 #define OS_SWI_UNUSED 0

software interrupt status unused

2.5.2.2.2.32 #define OS_SWI_DELETED 1

software interrupt status deleted

2.5.2.2.2.33 #define OS_SWI_ACQUIRED 2

software interrupt status aquired

2.5.2.2.2.34 #define OS_SWI_CREATED 3

software interrupt status created

2.5.2.2.2.35 #define OS_SWI_READY 4

software interrupt status ready

2.5.2.2.3 Function Documentation

2.5.2.2.3.1 os_status osSwiCreate (os_swi_function *handler,* os_swi_handle *swi_num,* os_swi_priority *priority,* os_user_id *user_id*)

Attaches an interrupt handler to a software interrupt.

Parameters

in	*handler*	- The software interrupt function.
in	*swi_num*	- The software interrupt number.
in	*priority*	- The software interrupt priority.
in	*user_id*	- User ID - currently not in use.

Return values

OS_SUCCESS	- The software interrupt handler was successfully attached.
OS_ERR_SWI_INVALID	- Invalid software interrupt number.
OS_ERR_SWI_FUNCTION_INVALID	- NULL software interrupt handler.
OS_ERR_SWI_PRIORITY_INVALID	- Invalid software interrupt priority.
OS_ERR_SWI_ALREADY_CREATED	- The software interrupt was already created.

2.5.2.2.3.2 os_status osSwiDelete (os_swi_handle *swi_num*)

Detaches the given software interrupt from its handler.

Parameters

in	*swi_num*	- The software interrupt number.

Return values

OS_SUCCESS	- The software interrupt handler was successfully detached.
OS_ERR_SWI_INVALID	- Invalid software interrupt number.
OS_ERR_SWI_NOT_C↩ REATED	- The software interrupt was not created, or it was already deleted.

2.5.2.2.3.3 os_status osSwiActivate (os_swi_handle *swi_num*)

Activates the given software interrupt.

Parameters

in	*swi_num*	- The software interrupt number.

Return values

OS_SUCCESS	- The software interrupt was successfully activated.
OS_ERR_SWI_INVALID	- Invalid software interrupt number.
OS_ERR_SWI_NOT_C↩ REATED	- The software interrupt was not created, or it was already deleted.

2.5.2.2.3.4 void osSwiEnable (os_swi_priority *priority*)

Enables all software interrupts with a given priority.

Parameters

in	*priority*	- Priority of software interrupts to enable.

2.5.2.2.3.5 os_swi_priority osSwiDisable ( )

Disables all software interrupts.

Returns

Previous enabled priority of software interrupts

2.5.2.2.3.6 os_status osSwiCountInc (os_swi_handle *swi_num*)

Increments the count of the given software interrupt.

The software interrupt will also be activated.

Parameters

in	*swi_num*	- The software interrupt number.

Return values

OS_SUCCESS	- The software interrupt count was incremented and activated.
OS_ERR_SWI_INVALID	- Invalid software interrupt number.
OS_ERR_SWI_NOT_C↩ REATED	- The software interrupt was not created, or it was already deleted.

2.5.2.2.3.7 os_status osSwiCountDec (os_swi_handle *swi_num*)

Decrements the count of the given software interrupt.

The software interrupt will also be activated if the count is zero after the operation.

Parameters

in	*swi_num*	- The software interrupt number.

Return values

OS_SUCCESS	- The count was decremented successfully (may be followed by activation).
OS_ERR_SWI_INVALID	- Invalid software interrupt number.
OS_ERR_SWI_NOT_C↩ REATED	- The software interrupt was not created, or it was already deleted.

2.5.2.2.3.8 os_status osSwiCountGet (os_swi_handle *swi_num,* os_swi_count * *count*)

Gets the count of the given software interrupt.

Parameters

in	*swi_num*	- The software interrupt number.
out	*count*	- Receives the software interrupt count.

Return values

OS_SUCCESS	- The value in *count* is valid.
OS_ERR_SWI_INVALID	- Invalid software interrupt number.
OS_ERR_SWI_NOT_C↩ REATED	- The software interrupt was not created, or it was already deleted.

2.5.2.2.3.9 os_status osSwiCountSet (os_swi_handle *swi_num,* os_swi_count *count*)

Sets the count of the given software interrupt.

Parameters

in	*swi_num*	- The software interrupt number.
in	*count*	- The new software interrupt count.

Return values

OS_SUCCESS	- The software interrupt count was set.
OS_ERR_SWI_INVALID	- Invalid software interrupt number.
OS_ERR_SWI_NOT_CREATED	- The software interrupt was not created, or it was already deleted.

2.5.2.2.3.10 os_status osSwiPriorityGet (os_swi_handle *swi_num*, os_swi_priority * *priority*)

Gets the priority of the given software interrupt.

Parameters

in	*swi_num*	- The software interrupt number.
out	*priority*	- Receives the software interrupt priority.

Return values

OS_SUCCESS	- The value in *priority* is valid.
OS_ERR_SWI_INVALID	- Invalid software interrupt number.
OS_ERR_SWI_NOT_CREATED	- The software interrupt was not created, or it was already deleted.

2.5.2.2.3.11 os_status osSwiPrioritySet (os_swi_handle *swi_num*, os_swi_priority *priority*)

Sets the priority of the given software interrupt.

Parameters

in	*swi_num*	- The software interrupt number.
in	*priority*	- The new software interrupt priority.

Return values

OS_SUCCESS	- The software interrupt priority was set.
OS_ERR_SWI_INVALID	- Invalid software interrupt number.

OS_ERR_SWI_PRIORITY_INVALID	- Invalid software interrupt priority.
OS_ERR_SWI_NOT_CREATED	- The software interrupt was not created, or it was already deleted.

Warning

The new priority will be valid only for the next activation.

2.5.2.2.3.12 os_swi_status osSwiStatus (os_swi_handle *swi_num*)

Gets the status of the given software interrupt.

Parameters

in	*swi_num*	- The software interrupt number.

Returns

The status of the specified software interrupt; if the software interrupt is not valid, SWI_UNUSED will be returned.

2.5.2.2.3.13 os_status osSwiFind (os_swi_handle * *swi_num*)

Finds the first available software interrupt number.

Parameters

out	*swi_num*	- Receives the software interrupt number.

Return values

OS_SUCCESS	- Found an available software interrupt number.
OS_ERR_SWI_UNAVAILABLE	- No software interrupt number is available.

2.5.2.2.3.14 os_status osSwiSelf (os_swi_handle * *swi_num*)

Gets the number of the currently active software interrupt.

Parameters

out	*swi_num*	- Receives the active software interrupt number.

Return values

OS_SUCCESS	- The value in *swi_num* is valid.
OS_ERR_SWI_NOT_ACTIVE	- No software interrupt is active; the function was not called from within an active software interrupt function.

2.5.3 Hardware Interrupts API

2.5.3.1 Overview

Hardware Interrupts setup and control.

Macros

- #define osHwiRemove osHwiDelete
- #define osHwiGetGPIO(hwi_num) osHwiGpioMap(hwi_num)

Functions

- void EI ()
- void DI ()
- void osHwiSwiftDisableNoLog ()
- void osHwiSwiftEnableNoLog ()
- void osHwiSwiftNestReduce ()
- os_status osHwiCreate (os_hwi_handle hwi_num, os_hwi_priority priority, os_hwi_mode mode, os_hwi_function handler, os_hwi_arg argument)
- os_status osHwiDispatcherCreate (os_hwi_handle hwi_num, os_hwi_priority priority, os_hwi_mode mode, os_hwi_function handler, os_hwi_arg argument, os_hwi_dispatcher dispatcher)
- os_status osHwiIsCreated (os_hwi_handle hwi_num, bool *created)
- os_status osHwiDelete (os_hwi_handle hwi_num)
- os_status osHwiMultiplexedCreate (os_hwi_handle hwi_num, os_hwi_priority priority, os_hwi_mode mode, os_hwi_function handler, os_hwi_arg argument)
- os_status osHwiMultiplexedDispatcherCreate (os_hwi_handle hwi_num, os_hwi_priority priority, os_hwi_mode mode, os_hwi_function handler, os_hwi_arg argument, os_hwi_dispatcher dispatcher)
- os_status osHwiMultiplexedDelete (os_hwi_handle hwi_num, os_hwi_function handler, os_hwi_arg argument)
- os_status osHwiPrioritySet (os_hwi_handle hwi_num, os_hwi_priority priority)
- os_status osHwiPendingClear (os_hwi_handle hwi_num)
- os_status osHwiVectorInstall (os_hwi_handle hwi_num, os_hwi_priority priority, os_hwi_function handler)
- os_hwi_status osHwiDisable ()
- void osHwiEnable (os_hwi_status prev_status)
- void osHwiSwiftDisable ()

- void osHwiSwiftEnable ()
- os_status osHwiGpioMap (os_hwi_handle hwi_num, unsigned int *gpio_pin)
- os_status osHwiGpioIrqConfigure (os_hwi_handle hwi_num)
- void invoke_interrupt (uint32_t num)

2.5.3.2 Macro Definition Documentation

2.5.3.2.1 #define osHwiRemove osHwiDelete

see 'osHwiDelete'

2.5.3.2.2 #define osHwiGetGPIO(*hwi_num*) osHwiGpioMap(hwi_num)

see 'osHwiGpioMap'

2.5.3.3 Function Documentation

2.5.3.3.1 void EI ( )

Interrupts Enable compiler function.

Warning

The function signature is for documentation purposes only

2.5.3.3.2 void DI ( )

Interrupts Disable compiler function.

Warning

The function signature is for documentation purposes only

2.5.3.3.3 void osHwiSwiftDisableNoLog ( )

Disable interrupts, no OS logging performed.

Warning

The function signature is for documentation purposes only

2.5.3.3.4 void osHwiSwiftEnableNoLog ( )

Enable interrupts, no OS logging performed.

Warning

The function signature is for documentation purposes only

2.5.3.3.5 void osHwiSwiftNestReduce ( )

Reduce interrupt disabling nesting.

Warning

The function signature is for documentation purposes only

2.5.3.3.6 os_status osHwiCreate (os_hwi_handle *hwi_num,* os_hwi_priority *priority,* os_hwi_mode *mode,* os_hwi_function *handler,* os_hwi_arg *argument*)

Installs an interrupt handler in the OS dispatcher.

```
This function sets the interrupt priority in the interrupt
assignment register and enables the interrupt.
```

Parameters

in	*hwi_num*	- The interrupt number (source).
in	*priority*	- The interrupt priority.
in	*mode*	- The interrupt mode.
in	*handler*	- The interrupt function (ISR).
in	*argument*	- Argument to be passed to the interrupt function.

Return values

OS_SUCCESS	- The interrupt handler was successfully installed.
OS_ERR_HWI_INVALID	- Invalid interrupt number (source).
OS_ERR_HWI_MODE_INVALID	- Invalid interrupt mode.
OS_ERR_HWI_FUNCTION_INVALID	- NULL interrupt handler.
OS_ERR_HWI_PRIORITY_INVALID	- Invalid interrupt priority.

2.5.3.3.7 os_status osHwiDispatcherCreate (os_hwi_handle *hwi_num,* os_hwi_priority *priority,* os_hwi_mode *mode,* os_hwi_function *handler,* os_hwi_arg *argument,* os_hwi_dispatcher *dispatcher*)

Installs an interrupt handler on a specific OS dispatcher.

This function sets the interrupt priority in the interrupt assignment register and enables the interrupt.

Parameters

in	*hwi_num*	- The interrupt number (source).
in	*priority*	- The interrupt priority.
in	*mode*	- The interrupt mode.
in	*handler*	- The interrupt function (ISR).
in	*argument*	- Argument to be passed to the interrupt function.
in	*dispatcher*	- Dispatcher to use.

Return values

OS_SUCCESS	- The interrupt handler was successfully installed.
OS_ERR_HWI_INVALID	- Invalid interrupt number (source).
OS_ERR_HWI_MODE_INVALID	- Invalid interrupt mode.
OS_ERR_HWI_FUNCTION_INVALID	- NULL interrupt handler.
OS_ERR_HWI_PRIORITY_INVALID	- Invalid interrupt priority.

2.5.3.3.8 os_status osHwiIsCreated (os_hwi_handle *hwi_num,* bool * *created*)

Queries whether the interrupt was already registered.

Parameters

in	*hwi_num*	- The interrupt number (source).
out	*created*	- Will be set to TRUE if created or FALSE if not.

Return values

OS_SUCCES	- The interrupt handler was installed.
OS_ERR_HWI_INVALID	- Invalid interrupt number (source).

Warning

May only be called for interrupts in the Hardware Interrupt Controller

2.5.3.3.9 os_status osHwiDelete (os_hwi_handle *hwi_num*)

Detaches an interrupt handler from the OS dispatcher.

Parameters

in	*hwi_num*	- The interrupt number (source).

Return values

OS_SUCCESS	- The interrupt handler was successfully detached.
OS_ERR_HWI_INVALID	- Invalid interrupt number (source).

2.5.3.3.10 os_status osHwiMultiplexedCreate (os_hwi_handle *hwi_num,* os_hwi_priority *priority,* os_hwi_mode *mode,* os_hwi_function *handler,* os_hwi_arg *argument*)

Installs or adds to a multiplexed interrupt handler in the OS dispatcher.

This function sets the interrupt priority in the interrupt assignment register and enables the interrupt.

Parameters

in	*hwi_num*	- The interrupt number (source).
in	*priority*	- The interrupt priority.
in	*mode*	- The interrupt mode.
in	*handler*	- The interrupt function (ISR).
in	*argument*	- Argument to be passed to the interrupt function.

Return values

OS_SUCCESS	- The interrupt handler was successfully installed.
OS_ERR_HWI_INVALID	- Invalid interrupt number (source).
OS_ERR_HWI_MODE_INVALID	- Invalid interrupt mode.
OS_ERR_HWI_FUNCTION_INVALID	- NULL interrupt handler.
OS_ERR_HWI_PRIORITY_INVALID	- Invalid interrupt priority.
OS_ERR_NO_MEMORY	- can't create an entry for multiplexed interrupt information

2.5.3.3.11 os_status osHwiMultiplexedDispatcherCreate (os_hwi_handle *hwi_num*, os_hwi_priority *priority*, os_hwi_mode *mode*, os_hwi_function *handler*, os_hwi_arg *argument*, os_hwi_dispatcher *dispatcher*)

Installs or adds to a multiplexed interrupt handler in the OS dispatcher.

This function sets the interrupt priority in the interrupt assignment register and enables the interrupt.

Parameters

in	*hwi_num*	- The interrupt number (source).
in	*priority*	- The interrupt priority.
in	*mode*	- The interrupt mode.
in	*handler*	- The interrupt function (ISR).
in	*argument*	- Argument to be passed to the interrupt function.
in	*dispatcher*	- Dispatcher to use.

Return values

OS_SUCCESS	- The interrupt handler was successfully installed.
OS_ERR_HWI_INVALID	- Invalid interrupt number (source).
OS_ERR_HWI_MODE_INVALID	- Invalid interrupt mode.
OS_ERR_HWI_FUNCTION_INVALID	- NULL interrupt handler.
OS_ERR_HWI_PRIORITY_INVALID	- Invalid interrupt priority.

2.5.3.3.12 os_status osHwiMultiplexedDelete (os_hwi_handle *hwi_num*, os_hwi_function *handler*, os_hwi_arg *argument*)

Removes an entry from a multiplexed interrupt handler in the OS dispatcher.

Parameters

in	*hwi_num*	- The interrupt number (source).
in	*handler*	- The interrupt function (ISR).
in	*argument*	- Argument to be passed to the interrupt function.

Return values

OS_SUCCESS	- The interrupt handler was successfully installed.
OS_FAIL	- Removing during active interrupt.
OS_ERR_HWI_INVALID	- Invalid interrupt number (source).
OS_ERR_HWI_COMM↩ AND_INVALID	- Called for a non multiplexed interrupt.
OS_ERR_ALREADY_F↩ REE	- Already removed this entry.

2.5.3.3.13 os_status osHwiPrioritySet (os_hwi_handle *hwi_num,* os_hwi_priority *priority*)

Sets the priority of the given hardware interrupt.

Parameters

in	*hwi_num*	- The interrupt number (source).
in	*priority*	- The new interrupt priority.

Return values

OS_SUCCESS	- The interrupt priority was set.
OS_ERR_HWI_INVALID	- Invalid interrupt number (source).
OS_ERR_HWI_PRIORI↩ TY_INVALID	- Invalid interrupt priority.

2.5.3.3.14 os_status osHwiPendingClear (os_hwi_handle *hwi_num*)

Clear pending interrupt bit in interrupt controller

Parameters

in	*hwi_num*	- The interrupt number (source).

Return values

OS_SUCCESS	- The interrupt priority was set.
OS_ERR_HWI_INVALID	- Invalid interrupt number (source).

2.5.3.3.15 os_status osHwiVectorInstall (os_hwi_handle *hwi_num,* os_hwi_priority *priority,* os_hwi_function *handler*)

Installs an interrupt vector directly into the interrupt vector table.

This function lets the interrupt bypass the OS dispatcher. The interrupt must manage the stack, and it cannot call OS functions that try to make a task ready.

Parameters

in	*hwi_num*	- The interrupt number (source).
in	*priority*	- The interrupt priority.
in	*handler*	- Pointer to the interrupt function.

Return values

OS_SUCCESS	- The interrupt vector was successfully installed.
OS_ERR_HWI_INVALID	- Invalid interrupt number (source).
OS_ERR_HWI_FUNCT↩ ION_INVALID	- NULL interrupt handler.
OS_ERR_HWI_PRIORI↩ TY_INVALID	- Invalid interrupt priority.

Warning

This function is not supported on all architectures

2.5.3.3.16 os_hwi_status osHwiDisable ( )

Disables interrupts and returns last interrupt status (priority).

Returns

Last hardware interrupt status (priority).

2.5.3.3.17 void osHwiEnable (os_hwi_status *prev_status*)

Enables interrupts with a given interrupt status (priority).

This function can be used to keep track of nested calls.

Parameters

in	*prev_status*	- Value returned from osHwiDisable().

2.5.3.3.18 void osHwiSwiftDisable ( )

Disables all interrupts immediately.

2.5.3.3.19 void osHwiSwiftEnable ( )

Enables interrupts immediately (does not change priorities).

2.5.3.3.20 os_status osHwiGpioMap (os_hwi_handle *hwi_num,* unsigned int * *gpio_pin*)

Maps external IRQ to GPIO pins.

Parameters

in	*hwi_num*	- The interrupt number (source).
out	*gpio_pin*	- The GPIO pin (source).

Return values

OS_SUCCESS	- if mapping is legal
OS_ERR_HWI_INVALID	- if hwi_num is not legal
OS_ERR_COMMAND_UNSUPPORTED	- on non-supporting architectures

2.5.3.3.21 os_status osHwiGpioIrqConfigure (os_hwi_handle *hwi_num*)

Configures a GPIO pin to IRQ functionality.

Parameters

in	*hwi_num*	- The interrupt number (source).

Return values

OS_SUCCESS	- if mapping is legal
OS_ERR_HWI_INVALID	- if hwi_num is not legal
OS_ERR_COMMAND_UNSUPPORTED	- on non-supporting architectures

2.5.3.3.22 void invoke_interrupt (uint32_t *num*)

invoke an given interrupt

Parameters

in	*num*	- The number of the Hardware interrupt to invoke.

2.5.4 Virtual Interrupts API

2.5.4.1 Overview

Virtual interrupts setup and control.

Functions

- os_status osVirtualInterruptsInitialize ()
- os_status osVirtualInterruptFind (uint32_t *interrupt_handle)
- os_status osVirtualInterruptRelease (uint32_t interrupt_handle)
- os_status osVirtualInterruptReserve (uint32_t interrupt_handle)
- os_status osVirtualInterruptActivate (uint32_t interrupt_handle)

2.5.4.2 Function Documentation

2.5.4.2.1 os_status osVirtualInterruptsInitialize ( )

Initialize the virtual interrupts mechanism.

Return values

os_status	

2.5.4.2.2 os_status osVirtualInterruptFind (uint32_t * *interrupt_handle*)

Get a virtual interrupt handle.

Parameters

out	*interrupt_handle*	- Interrupt number, can be used to call osHwiCreate.

Return values

OS_SUCCESS	: Found.
OS_ERR_VIRQ_UNAVAILABLE	: No more virq in the system.

2.5.4.2.3 os_status osVirtualInterruptRelease (uint32_t *interrupt_handle*)

Free a virtual interrupt handle.

Parameters

in	*interrupt_handle*	- Handle to an interrupt number.

Return values

OS_SUCCESS.	

2.5.4.2.4 os_status osVirtualInterruptReserve (uint32_t *interrupt_handle*)

Reserve a virtual interrupt handle; Same as osVirtualInterruptFind() but for a specific VIRQ number.

Parameters

in	*interrupt_handle*	- Handle to an interrupt number.

Return values

OS_SUCCESS.	

2.5.4.2.5 os_status osVirtualInterruptActivate (uint32_t *interrupt_handle*)

Activate virtual interrupt.

Parameters

out	*interrupt_handle*	- Interrupt number, received from osVirtualInterruptFind.

Return values

OS_SUCCESS	: Activated.

2.6 Timers

2.6.1 Overview

Group including all timers supported by the kernel.

Modules

- Software Timers API
- Hardware Timers APIF
- CORE Watchdogs API
- OS Tick API
- Hardware Timers Initialization API

2.6.2 Software Timers API

2.6.2.1 Overview

Software Timers setup and control.

Modules

- Software Timers

OS Software Timer objects values

```
Software Timer objects can be created dynamically by using
osTimerFind() or they can be statically referenced by the
definitions below.
```

- #define **OS_TIMER0** 0
- #define **OS_TIMER1** 1
- #define **OS_TIMER2** 2
- #define **OS_TIMER3** 3
- #define **OS_TIMER4** 4
- #define **OS_TIMER5** 5
- #define **OS_TIMER6** 6
- #define **OS_TIMER7** 7
- #define **OS_TIMER8** 8
- #define **OS_TIMER9** 9
- #define **OS_TIMER10** 10

2.6.2.2 Software Timers

2.6.2.2.1 Overview

```
Software Timers setup and control.
```

Functions

- os_status osTimerCreate (os_timer_handle timer_num, os_timer_mode mode, os_timer_interval interval, os_timer_function handler)
- os_status osTimerDelete (os_timer_handle timer_num)
- os_status osTimerStart (os_timer_handle timer_num)
- os_status osTimerStartDelayed (os_timer_handle timer_num, os_timer_interval delay)
- os_status osTimerStop (os_timer_handle timer_num)
- os_status osTimerSetInterval (os_timer_handle timer_num, os_timer_interval interval)
- os_status osTimerGetDelay (os_timer_handle timer_num, os_timer_interval ∗delay)
- os_status osTimerFind (os_timer_handle ∗timer_num)
- os_status osTimerSelf (os_timer_handle ∗timer_num)

2.6.2.2.2 Function Documentation

2.6.2.2.2.1 os_status osTimerCreate (os_timer_handle *timer_num,* os_timer_mode *mode,* os_timer_interval *interval,* os_timer_function *handler*)

Creates a software timer and sets its parameters.

The timer is not started until you call osStart()

Parameters

in	*timer_num*	- The software timer number.
in	*mode*	- Timer mode (one-shot, periodic, free-run).
in	*interval*	- The timer interval (in system ticks).
in	*handler*	- The function to call when the timer expires.

Return values

OS_SUCCESS	- The timer was successfully created.
OS_ERR_TM_INVALID	- Invalid timer number.
OS_ERR_TM_MODE_I↩ NVALID	- Invalid timer mode.
OS_ERR_TM_FUNCTI↩ ON_INVALID	- NULL timer function.
OS_ERR_TM_ALREAD↩ Y_CREATED	- The timer is already created.

2.6.2.2.2.2 os_status osTimerDelete (os_timer_handle *timer_num*)

Deletes the given software timer from the system.

Parameters

in	*timer_num*	- The software timer number.

Return values

OS_SUCCESS	- The timer was successfully deleted.
OS_ERR_TM_INVALID	- Invalid timer number.
OS_ERR_TM_NOT_CR↩ EATED	- The timer was not created, or it was already deleted.

2.6.2.2.2.3 os_status osTimerStart (os_timer_handle *timer_num*)

Starts the given software timer.

This function should be used to start a software timer This function can be used to start/restart a one-shot software timer.

Parameters

in	*timer_num*	- The software timer number.

Return values

OS_SUCCESS	- The timer was successfully started.
OS_ERR_TM_INVALID	- Invalid timer number.
OS_ERR_TM_NOT_CR↩ EATED	- The timer was not created, or it was already deleted.

2.6.2.2.2.4 os_status osTimerStartDelayed (os_timer_handle *timer_num,* os_timer_interval *delay*)

Starts the given software timer.

This function should be used to start a software timer This function can be used to start/restart a one-shot software timer.

Parameters

in	*timer_num*	- The software timer number.
in	*delay*	- Delay of first period

Return values

OS_SUCCESS	- The timer was successfully started.
OS_ERR_TM_INVALID	- Invalid timer number.
OS_ERR_TM_NOT_CR↩ EATED	- The timer was not created, or it was already deleted.

2.6.2.2.2.5 os_status osTimerStop (os_timer_handle *timer_num*)

Stops the given software timer.

Parameters

in	*timer_num*	- The software timer number.

Return values

OS_SUCCESS	- The timer was successfully stopped.
OS_ERR_TM_INVALID	- Invalid timer number.
OS_ERR_TM_NOT_CR↩ EATED	- The timer was not created, or it was already deleted.

2.6.2.2.2.6 os_status osTimerSetInterval (os_timer_handle *timer_num,* os_timer_interval *interval*)

Sets the interval of the given software timer.

Parameters

in	*timer_num*	- The software timer number.
in	*interval*	- The new interval for the timer.

Return values

OS_SUCCESS	- The timer interval was successfully set.
OS_ERR_TM_INVALID	- Invalid timer number.
OS_ERR_TM_NOT_CR↩ EATED	- The timer was not created, or it was already deleted.

2.6.2.2.2.7 os_status osTimerGetDelay (os_timer_handle *timer_num,* os_timer_interval * *delay*)

Gets number of ticks left for the timer to fire.

Parameters

in	*timer_num*	- The software timer number.
out	*delay*	- Ticks left for timer to fire.

Return values

OS_SUCCESS	- The timer interval was successfully set.
OS_ERR_TM_INVALID	- Invalid timer number.
OS_ERR_TM_NOT_CR↩ EATED	- The timer was not created, or it was already deleted.

2.6.2.2.2.8 os_status osTimerFind (os_timer_handle * *timer_num*)

Finds the first available software timer number.

Parameters

out	*timer_num*	- Receives the software timer number.

Return values

OS_SUCCESS	- Found an available timer number.
OS_ERR_TM_UNAVAI↩ LABLE	- No timer number is available.

2.6.2.2.2.9 os_status osTimerSelf (os_timer_handle * *timer_num*)

Gets the number of the currently active software timer.

Parameters

out	*timer_num*	- Receives the active software timer number.

Return values

OS_SUCCESS	- The value in *timer_num* is valid.
OS_ERR_TM_NOT_ACTIVE	- No timer is active; the function was not called from within an active timer function.

2.6.3 Hardware Timers APIF

2.6.3.1 Overview

Hardware timers in the SoC.

Hardware Timers setup and control.

Modules

- Hardware Timers

Data Structures

- struct timer32_open_params_t

Enumerations

- enum { HWT_TIMER_0 = 0x1, HWT_TIMER_1 = 0x2, HWT_TIMER_2 = 0x4, HWT_TIMER_3 = 0x8 }
- enum { HWT_MODULE_0, HWT_MODULE_1 }
- enum timer32_operation_mode_t { SOC_TIMER32_ONE_SHOT = 1, SOC_TIMER32_PERIODIC = 2, SOC_TIMER32_FREE_RUN = 3 }
- enum timer32_counting_mode_t { SOC_TIMER32_DEFAULT_COUNT = 0, SOC_TIMER32_COUNT_RISING_EDGE = 1, SOC_TIMER32_COUNT_RISING_AND_FALL_EDGE = 2 , SOC_TIMER32_COUNT_QUADRATURE = 4, SOC_TIMER32_COUNT_PRIMARY_DIR_SECONDARY = 5, SOC_TIMER32_COUNT_CASCADED = 7 }
- enum timer32_primary_source_t { SOC_TIMER32_TIN0SOURCE = 0, SOC_TIMER32_TIN1SOURCE = 0x1, SOC_TIMER32_TIN2SOURCE = 0x2, SOC_TIMER32_TIN3SOURCE = 0x3, SOC_TIMER32_OFLAG0SOURCE = 0x4, SOC_TIMER32_OFLAG1SOURCE = 0x5, SOC_TIMER32_OFLAG2SOURCE = 0x6, SOC_TIMER32_OFLAG3SOURCE = 0x7, SOC_T

IMER32_PRESCALER1SOURCE = 0x8,
SOC_TIMER32_PRESCALER2SOURCE = 0x9, SOC_TIMER32_PRESCALER4SOURCE = 0xa, SOC_TIMER32_PRESCALER8SOURCE = 0xb,
SOC_TIMER32_PRESCALER16SOURCE = 0xc, SOC_TIMER32_PRESCALER32SOURCE = 0xd, SOC_TIMER32_PRESCALE64SOURCE = 0xe,
SOC_TIMER32_PRESCALER128SOURCE = 0xf, SOC_TIMER32_NOT_IN_USE }

- enum timer32_secondary_source_t {
SOC_TIMER32_SECONDARY_TIN0, SOC_TIMER32_SECONDARY_TIN1, SOC_TIME↩
R32_SECONDARY_TIN2,
SOC_TIMER32_SECONDARY_TIN3 }
- enum timer32_count_direction_t
- enum timer32_number_in_module_t {
SOC_TIMER32_IN_MODULE_TIMER_0 = 0x1, SOC_TIMER32_IN_MODULE_TIMER_1 = 0x2, SOC_TIMER32_IN_MODULE_TIMER_2 = 0x4,
SOC_TIMER32_IN_MODULE_TIMER_3 = 0x8 }
- enum timer32_preload_ctrl_t { SOC_TIMER32_PRELOAD_NEVER = 0x0, SOC_TIMER32_P↩
RELOAD_UPON_CMP1 = 0x1, SOC_TIMER32_PRELOAD_UPON_CMP2 = 0x2 }
- enum
- enum timer_clock_input_t {
SOC_CLOCK_IN_DISABLED = 0, SOC_CLOCK_IN_PLAFORM2 = 1, SOC_CLOCK_IN_C↩
PRI_PLL1 = 2,
SOC_CLOCK_IN_CPRI_PLL2 = 3, SOC_CLOCK_IN_CPRI_RECOVERY_PLL1 = 4, SOC_C↩
LOCK_IN_CPRI_RECOVERY_PLL2 = 5 }
- enum soc_timer32_num_t {

SOC_TIMER32_0 = 0, SOC_TIMER32_1 = 1, SOC_TIMER32_2 = 2,
SOC_TIMER32_3 = 3, SOC_TIMER32_4 = 4, SOC_TIMER32_5 = 5,
SOC_TIMER32_6 = 6, SOC_TIMER32_7 = 7, SOC_TIMER32_8 = 8,
SOC_TIMER32_9 = 9, SOC_TIMER32_10 = 10, SOC_TIMER32_11 = 11,
SOC_TIMER32_12 = 12, SOC_TIMER32_13 = 13, SOC_TIMER32_14 = 14,
SOC_TIMER32_15 = 15, SOC_TIMER32_16 = 16, SOC_TIMER32_17 = 17,
SOC_TIMER32_18 = 18, SOC_TIMER32_19 = 19, SOC_TIMER32_20 = 20,
SOC_TIMER32_21 = 21, SOC_TIMER32_22 = 22, SOC_TIMER32_23 = 23,
SOC_TIMER32_24 = 24, SOC_TIMER32_25 = 25, SOC_TIMER32_26 = 26,
SOC_TIMER32_27 = 27, SOC_TIMER32_28 = 28, SOC_TIMER32_29 = 29,
SOC_TIMER32_30 = 30, SOC_TIMER32_31 = 31, SOC_TIMER32_32,
SOC_TIMER32_33, SOC_TIMER32_34, SOC_TIMER32_35,
SOC_TIMER32_36, SOC_TIMER32_37, SOC_TIMER32_38,
SOC_TIMER32_39, SOC_TIMER32_40, SOC_TIMER32_41,
SOC_TIMER32_42, SOC_TIMER32_43, SOC_TIMER32_44,
SOC_TIMER32_45, SOC_TIMER32_46, SOC_TIMER32_47,
SOC_TIMER32_48, SOC_TIMER32_49, SOC_TIMER32_50,
SOC_TIMER32_51, SOC_TIMER32_52, SOC_TIMER32_53,
SOC_TIMER32_54, SOC_TIMER32_55, SOC_TIMER32_56,
SOC_TIMER32_57, SOC_TIMER32_58, SOC_TIMER32_59,
SOC_TIMER32_60, SOC_TIMER32_61, SOC_TIMER32_62,
SOC_TIMER32_63, SOC_TIMER32_64, SOC_TIMER32_65,
SOC_TIMER32_66, SOC_TIMER32_67, SOC_TIMER32_68,
SOC_TIMER32_69, SOC_TIMER32_70, SOC_TIMER32_71,
SOC_TIMER32_72, SOC_TIMER32_73, SOC_TIMER32_74,
SOC_TIMER32_75, SOC_TIMER32_76, SOC_TIMER32_77,
SOC_TIMER32_78, SOC_TIMER32_79, SOC_TIMER32_80,
SOC_TIMER32_81, SOC_TIMER32_82, SOC_TIMER32_83,
SOC_TIMER32_84, SOC_TIMER32_85, SOC_TIMER32_86,
SOC_TIMER32_87, SOC_TIMER32_88, SOC_TIMER32_89,
SOC_TIMER32_90, SOC_TIMER32_91, SOC_TIMER32_92,
SOC_TIMER32_93, SOC_TIMER32_94, SOC_TIMER32_95,
SOC_TIMER32_96, SOC_TIMER32_97, SOC_TIMER32_98,
SOC_TIMER32_99, SOC_TIMER32_100, SOC_TIMER32_101,
SOC_TIMER32_102, SOC_TIMER32_103, SOC_TIMER32_104,
SOC_TIMER32_105, SOC_TIMER32_106, SOC_TIMER32_107,
SOC_TIMER32_108, SOC_TIMER32_109, SOC_TIMER32_110,
SOC_TIMER32_111, SOC_TIMER32_112, SOC_TIMER32_113,
SOC_TIMER32_114, SOC_TIMER32_115, SOC_TIMER32_116,
SOC_TIMER32_117, SOC_TIMER32_118, SOC_TIMER32_119,
SOC_TIMER32_120, SOC_TIMER32_121, SOC_TIMER32_122,
SOC_TIMER32_123, SOC_TIMER32_124, SOC_TIMER32_125,
SOC_TIMER32_126, SOC_TIMER32_127 }

- enum soc_timer32_module_t {
  SOC_TIMER32_MODULE0 = 0, SOC_TIMER32_MODULE1 = 1, SOC_TIMER32_MODULE2

= 2,
SOC_TIMER32_MODULE3 = 3, SOC_TIMER32_MODULE4 = 4, SOC_TIMER32_MODULE5 = 5,
SOC_TIMER32_MODULE6 = 6, SOC_TIMER32_MODULE7 = 7, SOC_TIMER32_MODULE8 = 8,
SOC_TIMER32_MODULE9 = 9, SOC_TIMER32_MODULE10 = 10, SOC_TIMER32_MODU↩
LE11 = 11,
SOC_TIMER32_MODULE12 = 12, SOC_TIMER32_MODULE13 = 13, SOC_TIMER32_MO↩
DULE14 = 14,
SOC_TIMER32_MODULE15 = 15, SOC_TIMER32_MODULE16 = 16, SOC_TIMER32_MO↩
DULE17 = 17,
SOC_TIMER32_MODULE18 = 18, SOC_TIMER32_MODULE19 = 19, SOC_TIMER32_MO↩
DULE20 = 20,
SOC_TIMER32_MODULE21 = 21, SOC_TIMER32_MODULE22 = 22, SOC_TIMER32_MO↩
DULE23 = 23,
SOC_TIMER32_MODULE24 = 24, SOC_TIMER32_MODULE25 = 25, SOC_TIMER32_MO↩
DULE26 = 26,
SOC_TIMER32_MODULE27 = 27, SOC_TIMER32_MODULE28 = 28, SOC_TIMER32_MO↩
DULE29 = 29,
SOC_TIMER32_MODULE30 = 30, SOC_TIMER32_MODULE31 = 31 }

- enum soc_timer32_group_t {
  SOC_TIMER32_GROUP0 = 0, SOC_TIMER32_GROUP1 = 1, SOC_TIMER32_GROUP2 = 2,
  SOC_TIMER32_GROUP3 = 3, SOC_TIMER32_GROUP4 = 4, SOC_TIMER32_GROUP5 = 5,
  SOC_TIMER32_GROUP6 = 6, SOC_TIMER32_GROUP7 = 7 }
- enum soc_timer32_maple_t { SOC_TIMER32_MAPLE0 = 0, SOC_TIMER32_MAPLE1 = 1, S↩
  OC_TIMER32_MAPLE2 = 2 }

Functions

- uint32_t osHwTimerGlobalGet (unsigned int module)
- os_status osHwTimerGlobalConfig (unsigned int module, unsigned int timers, uint32_t hold0, uint32_t hold1, uint32_t hold2)
- os_status osHwTimerLock (unsigned int module, unsigned int timers)
- os_status osHwTimerUnlock (unsigned int module, unsigned int timers)
- os_status hwTimer32Open (soc_timer32_num_t timer_num, timer32_open_params_t *params)
- os_status hwTimer32SetInterval (soc_timer32_num_t timer_num, os_timer_interval interval)
- os_status hwTimer32Delete (soc_timer32_num_t timer_num)
- os_status hwTimer32Start (soc_timer32_num_t timer_num)
- os_status hwTimer32Stop (soc_timer32_num_t timer_num)
- void hwTimer32ClearEvent (soc_timer32_num_t timer_num)
- os_status hwTimer32ValueGet (soc_timer32_num_t timer_num, os_timer_interval *value)
- os_status hwTimer32HoldGet (soc_timer32_num_t timer_num, uint32_t *value)
- os_status hwTimer32CounterSet (soc_timer32_num_t timer_num, uint32_t value)
- os_status hwTimer32PreloadSet (soc_timer32_num_t timer_num, uint32_t preload_value)
- os_status hwTimer32CompareSet (soc_timer32_num_t timer_num, uint32_t compare)
- os_status hwTimer32SetAndForget (soc_timer32_module_t module, timer32_input_select_↩
  t timer_input)
- os_status hwTimer32LockClear (soc_timer32_module_t module, timer32_input_select_t timer_↩

input)
- os_status hwTimer32GlobalConfig (soc_timer32_module_t module, timer32_number_in_module_t timers, uint32_t hold0, uint32_t hold1, uint32_t hold2)
- uint32_t hwTimer32GlobalGet (soc_timer32_module_t module)
- os_status hwTimer32SetInitiatorMode (soc_timer32_module_t module, timer32_number_in_module_t initiator_timer, uint32_t reinit_timers, uint32_t output_signal_timers)
- os_status osHwTimerCreate (os_timer_handle timer_num, os_timer_mode mode, os_timer_interval interval, os_hw_timer_source source, os_hwi_function handler, os_hwi_priority priority)
- os_status osHwTimerSetInterval (os_timer_handle timer_num, os_timer_interval interval)
- os_status osHwTimerDelete (os_timer_handle timer_num)
- os_status osHwTimerStart (os_timer_handle timer_num)
- os_status osHwTimerStop (os_timer_handle timer_num)
- void osHwTimerClearEvent (os_timer_handle timer_num)
- os_status osHwTimerFind (os_timer_handle *timer_num)
- os_status osHwTimerTypeFind (os_timer_handle *timer_num, int type)
- os_status osHwTimerFindForce (os_timer_handle timer_num)
- os_status osHwTimerValueGet (os_timer_handle timer_num, os_timer_interval *value)
- void osHwTimerSyncStart (uint32_t core_mask)
- void osHwTimerSyncStop (uint32_t core_mask)
- os_status osHwTimerFreeze (os_timer_handle timer_num)
- os_status osHwTimerUnfreeze (os_timer_handle timer_num)
- os_status hwTimer32MapleInterruptAssign (soc_timer32_num_t timer_num, soc_timer32_maple_t maple_num)

Clocking sources for SoC timers

For clocking sources of the format PRESCALERnSOURCE the input clock is osSlowSystemClockGet()

- #define TIN0SOURCE 0x0
- #define TIN1SOURCE 0x1
- #define TIN2SOURCE 0x2
- #define TIN3SOURCE 0x3
- #define OFLAG0SOURCE 0x4
- #define OFLAG1SOURCE 0x5
- #define OFLAG2SOURCE 0x6
- #define OFLAG3SOURCE 0x7
- #define PRESCALER1SOURCE 0x8
- #define PRESCALER2SOURCE 0x9
- #define PRESCALER4SOURCE 0xA
- #define PRESCALER8SOURCE 0xB
- #define PRESCALER16SOURCE 0xC
- #define PRESCALER32SOURCE 0xD
- #define PRESCALER64SOURCE 0xE
- #define PRESCALER128SOURCE 0xF
- #define DEFAULT_TIMER_SOURCE PRESCALER1SOURCE
- #define NUM_OF_HW_TIMERS_SOURCES 16

Types of Hardware Timers

When calling osHwTimerFindForce(), use one of the values, in addition to the requested timer, to specify which counter is being used.

For example: osHwTimerFindForce(HWT_SOC_16BIT | 5) for timer #5 of the 16 bit type

- #define HWT_SOC_16BIT 0x0000
- #define HWT_SOC_32BIT 0x8000
- #define HWT_SOC_64BIT 0x4000

Counting modes for SoC timers

- #define COUNT_RISING_EDGE 0x00010000
- #define COUNT_RISING_AND_FALL_EDGE 0x00020000
- #define COUNT_RISING_PRIMARY 0x00030000
- #define **COUNT_QUADRATURE** 0x00040000
- #define COUNT_PRIMARY_DIR_SECONDARY 0x00050000
- #define COUNT_SECONDARY_TRIGGER 0x00060000
- #define COUNT_CASCADED 0x00070000

Clocking b4860 specific, trigger sources for SoC timers

- enum hw_timer_trigger_input_t {
  SOC_MUX_IN_TMR0_OUT = 0x00000000, SOC_MUX_IN_TMR1_OUT = 0x00000001, SO↩
  C_MUX_IN_TMR2_OUT = 0x00000002,
  SOC_MUX_IN_TMR3_OUT = 0x00000003, SOC_MUX_IN_TMR4_OUT = 0x00000004, SO↩
  C_MUX_IN_TMR5_OUT = 0x00000005,
  SOC_MUX_IN_TMR6_OUT = 0x00000006, SOC_MUX_IN_TMR7_OUT = 0x00000007, SO↩
  C_MUX_IN_CPRI0_RX_HFP_OUT = 0x00000008,
  SOC_MUX_IN_CPRI1_RX_HFP_OUT = 0x00000009, SOC_MUX_IN_CPRI2_RX_HFP_OU↩
  T = 0x0000000a, SOC_MUX_IN_CPRI3_RX_HFP_OUT = 0x0000000b,
  SOC_MUX_IN_CPRI4_RX_HFP_OUT = 0x0000000c, SOC_MUX_IN_CPRI5_RX_HFP_OU↩
  T = 0x0000000d, SOC_MUX_IN_CPRI6_RX_HFP_OUT = 0x0000000e,
  SOC_MUX_IN_CPRI7_RX_HFP_OUT = 0x0000000f, SOC_MUX_IN_CPRI0_TX_HFP_OUT =
  0x00000010, SOC_MUX_IN_CPRI1_TX_HFP_OUT = 0x00000011,
  SOC_MUX_IN_CPRI2_TX_HFP_OUT = 0x00000012, SOC_MUX_IN_CPRI3_TX_HFP_OUT =
  0x00000013, SOC_MUX_IN_CPRI4_TX_HFP_OUT = 0x00000014,
  SOC_MUX_IN_CPRI5_TX_HFP_OUT = 0x00000015, SOC_MUX_IN_CPRI6_TX_HFP_OUT =
  0x00000016, SOC_MUX_IN_CPRI7_TX_HFP_OUT = 0x00000017,
  SOC_MUX_IN_CPRI0_RX_BFP_OUT = 0x00000018, SOC_MUX_IN_CPRI1_RX_BFP_OU↩
  T = 0x00000019, SOC_MUX_IN_CPRI2_RX_BFP_OUT = 0x0000001a,
  SOC_MUX_IN_CPRI3_RX_BFP_OUT = 0x0000001b, SOC_MUX_IN_CPRI4_RX_BFP_OU↩
  T = 0x0000001c, SOC_MUX_IN_CPRI5_RX_BFP_OUT = 0x0000001d,
  SOC_MUX_IN_CPRI6_RX_BFP_OUT = 0x0000001e, SOC_MUX_IN_CPRI7_RX_BFP_OUT =
  0x0000001f, SOC_MUX_IN_CPRI0_TX_BFP_OUT = 0x00000020,
  SOC_MUX_IN_CPRI1_TX_BFP_OUT = 0x00000021, SOC_MUX_IN_CPRI2_TX_BFP_OUT =
  0x00000022, SOC_MUX_IN_CPRI3_TX_BFP_OUT = 0x00000023,
  SOC_MUX_IN_CPRI4_TX_BFP_OUT = 0x00000024, SOC_MUX_IN_CPRI5_TX_BFP_OUT =
  0x00000025, SOC_MUX_IN_CPRI6_TX_BFP_OUT = 0x00000026,
  SOC_MUX_IN_CPRI7_TX_BFP_OUT = 0x00000027, SOC_MUX_IN_CPRI0_RX_RF_OUT =

0x00000028, SOC_MUX_IN_CPRI1_RX_RF_OUT = 0x00000029,
SOC_MUX_IN_CPRI2_RX_RF_OUT = 0x0000002a, SOC_MUX_IN_CPRI3_RX_RF_OUT = 0x0000002b, SOC_MUX_IN_CPRI4_RX_RF_OUT = 0x0000002c,
SOC_MUX_IN_CPRI5_RX_RF_OUT = 0x0000002d, SOC_MUX_IN_CPRI6_RX_RF_OUT = 0x0000002e, SOC_MUX_IN_CPRI7_RX_RF_OUT = 0x0000002f,
SOC_MUX_IN_CPRI0_TX_RF_OUT = 0x00000030, SOC_MUX_IN_CPRI1_TX_RF_OUT = 0x00000031, SOC_MUX_IN_CPRI2_TX_RF_OUT = 0x00000032,
SOC_MUX_IN_CPRI3_TX_RF_OUT = 0x00000033, SOC_MUX_IN_CPRI4_TX_RF_OUT = 0x00000034, SOC_MUX_IN_CPRI5_TX_RF_OUT = 0x00000035,
SOC_MUX_IN_CPRI6_TX_RF_OUT = 0x00000036, SOC_MUX_IN_CPRI7_TX_RF_OUT = 0x00000037, SOC_MUX_IN_SYS_CLOCK = 0x00000038,
SOC_MUX_IN_1588_1 = 0x00000039, SOC_MUX_IN_1588_2 = 0x0000003a, SOC_MUX_IN↩
_CPRI0_RX_TRANSACTION = 0x0000040,
SOC_MUX_IN_CPRI1_RX_TRANSACTION = 0x0000041, SOC_MUX_IN_CPRI2_RX_TRA↩
NSACTION = 0x0000042, SOC_MUX_IN_CPRI3_RX_TRANSACTION = 0x0000043,
SOC_MUX_IN_CPRI4_RX_TRANSACTION = 0x0000044, SOC_MUX_IN_CPRI5_RX_TRA↩
NSACTION = 0x0000045, SOC_MUX_IN_CPRI6_RX_TRANSACTION = 0x0000046,
SOC_MUX_IN_CPRI7_RX_TRANSACTION = 0x0000047, SOC_MUX_IN_CPRI0_TX_TRA↩
NSACTION = 0x0000048, SOC_MUX_IN_CPRI1_TX_TRANSACTION = 0x0000049,
SOC_MUX_IN_CPRI2_TX_TRANSACTION = 0x000004a, SOC_MUX_IN_CPRI3_TX_TRA↩
NSACTION = 0x000004b, SOC_MUX_IN_CPRI4_TX_TRANSACTION = 0x000004c,
SOC_MUX_IN_CPRI5_TX_TRANSACTION = 0x000004d, SOC_MUX_IN_CPRI6_TX_TRA↩
NSACTION = 0x000004e, SOC_MUX_IN_CPRI7_TX_TRANSACTION = 0x000004f,
SOC_MUX_IN_CPRI0_RX2_TRANSACTION = 0x0000050, SOC_MUX_IN_CPRI1_RX2_T↩
RANSACTION = 0x0000051, SOC_MUX_IN_CPRI2_RX2_TRANSACTION = 0x0000052,
SOC_MUX_IN_CPRI3_RX2_TRANSACTION = 0x0000053, SOC_MUX_IN_CPRI4_RX2_T↩
RANSACTION = 0x0000054, SOC_MUX_IN_CPRI5_RX2_TRANSACTION = 0x0000055,
SOC_MUX_IN_CPRI6_RX2_TRANSACTION = 0x0000056, SOC_MUX_IN_CPRI7_RX2_T↩
RANSACTION = 0x0000057 }

- #define MAX_HW_TIMERS_SECONDARY_SOURCES SOC_MUX_IN_CPRI7_RX2_TRAN↩ SACTION
- #define TIMER32_NUM_OF_TRIGGERS_PER_GROUP 4
- #define TIMER32_MAX_SOC_INTERRUPTS (16∗TIMER32_NUM_OF_TRIGGERS_PER_G↩ ROUP)

2.6.3.2 Data Structure Documentation

2.6.3.2.1 struct timer32_open_params_t

Timer32 initialization structure.

Data Fields

- os_timer_interval interval
- os_timer_interval preload

- timer32_operation_mode_t operation_mode
- timer32_counting_mode_t counting_mode
- timer32_count_direction_t direction
- os_hwi_function handler
- os_hwi_priority priority
- timer32_primary_source_t primary_source
- timer32_secondary_source_t secondary_source
- uint32_t output_enable:1
- uint32_t preload_ctrl1:2

2.6.3.2.1.1 Field Documentation

2.6.3.2.1.1.1 os_timer_interval timer32_open_params_t::interval

counting interval.

Any 32bit value is valid

2.6.3.2.1.1.2 os_timer_interval timer32_open_params_t::preload

value load to timer after successful comparison

2.6.3.2.1.1.3 timer32_operation_mode_t timer32_open_params_t::operation_mode

general mode of operation: OS_TIMER_ONE_SHOT, OS_TIMER_PERIODIC, OS_TIMER_FREE_↩ RUN

2.6.3.2.1.1.4 timer32_counting_mode_t timer32_open_params_t::counting_mode

arch specific mode of counting

2.6.3.2.1.1.5 timer32_count_direction_t timer32_open_params_t::direction

counting direction (up/down)

2.6.3.2.1.1.6 os_hwi_function timer32_open_params_t::handler

interrupt handler, when NULL value supplied, interrupts are disabled

user is reponsible to clearing interrupt with hwTimer32ClearEvent

2.6.3.2.1.1.7 os_hwi_priority timer32_open_params_t::priority

interrupt priority.

Irrelevant when interrupt handler is NULL

2.6.3.2.1.1.8 timer32_primary_source_t timer32_open_params_t::primary_source

timer primary source clock

2.6.3.2.1.1.9 timer32_secondary_source_t timer32_open_params_t::secondary_source

secondary counting source input selection

2.6.3.2.1.1.10 uint32_t timer32_open_params_t::output_enable

if TRUE output is enabled

2.6.3.2.1.1.11 uint32_t timer32_open_params_t::preload_ctrl1

wheather and how to preload compare1 value

2.6.3.3 Macro Definition Documentation

2.6.3.3.1 #define TIN0SOURCE 0x0

Timer source clock is TIN0.

2.6.3.3.2 #define TIN1SOURCE 0x1

Timer source clock is TIN1.

2.6.3.3.3 #define TIN2SOURCE 0x2

Timer source clock is TIN2.

2.6.3.3.4 #define TIN3SOURCE 0x3

Timer source clock is TIN3.

2.6.3.3.5 #define OFLAG0SOURCE 0x4

Timer source clock is timer 0.

2.6.3.3.6 #define OFLAG1SOURCE 0x5

Timer source clock is timer 1.

2.6.3.3.7 #define OFLAG2SOURCE 0x6

Timer source clock is timer 2.

2.6.3.3.8 #define OFLAG3SOURCE 0x7

Timer source clock is timer 3.

2.6.3.3.9 #define PRESCALER1SOURCE 0x8

Timer source clock is (input clock)/1.

2.6.3.3.10 #define PRESCALER2SOURCE 0x9

Timer source clock is (input clock)/2.

2.6.3.3.11 #define PRESCALER4SOURCE 0xA

Timer source clock is (input clock)/4.

2.6.3.3.12 #define PRESCALER8SOURCE 0xB

Timer source clock is (input clock)/8.

2.6.3.3.13 #define PRESCALER16SOURCE 0xC

Timer source clock is (input clock)/16.

2.6.3.3.14 #define PRESCALER32SOURCE 0xD

Timer source clock is (input clock)/32.

2.6.3.3.15 #define PRESCALER64SOURCE 0xE

Timer source clock is (input clock)/64.

2.6.3.3.16 #define PRESCALER128SOURCE 0xF

Timer source clock is (input clock)/128.

2.6.3.3.17 #define DEFAULT_TIMER_SOURCE PRESCALER1SOURCE

Default source.

2.6.3.3.18 #define NUM_OF_HW_TIMERS_SOURCES 16

number of SOC timer inputs

2.6.3.3.19 #define HWT_SOC_16BIT 0x0000

16 bit timer

2.6.3.3.20 #define HWT_SOC_32BIT 0x8000

32 bit timer

2.6.3.3.21 #define HWT_SOC_64BIT 0x4000

64 bit timer

2.6.3.3.22 #define COUNT_RISING_EDGE 0x00010000

Count rising edges of the primary source.

2.6.3.3.23 #define COUNT_RISING_AND_FALL_EDGE 0x00020000

Count rising and falling edges of the primary source.

2.6.3.3.24 #define COUNT_RISING_PRIMARY 0x00030000

Count rising edges of the primary source while the secondary input is high active.

2.6.3.3.25 #define COUNT_PRIMARY_DIR_SECONDARY 0x00050000

Quadrature count mode, uses primary clock and secondary input.
Count rising edges of the primary clock; secondary input specifies direction (1 = minus)

2.6.3.3.26 #define COUNT_SECONDARY_TRIGGER 0x00060000

Edge of the secondary input triggers primary count until a compare occurs.

2.6.3.3.27 #define COUNT_CASCADED 0x00070000

Cascaded counter mode (up/down).

2.6.3.3.28 #define MAX_HW_TIMERS_SECONDARY_SOURCES SOC_MUX_IN_CPRI7_RX2_TRANSACTION

number of secondary sources

2.6.3.3.29 #define TIMER32_NUM_OF_TRIGGERS_PER_GROUP 4

number of triggers per group

2.6.3.3.30 #define TIMER32_MAX_SOC_INTERRUPTS (16*TIMER32_NUM_OF_TRIGGERS_PER_GROUP)

number of timer interrupts

2.6.3.4 Enumeration Type Documentation

2.6.3.4.1 anonymous enum

Hardware Timer in Quandrate Module.

Enumerator

HWT_TIMER_0 Timer #0 in quadrate.
HWT_TIMER_1 Timer #1 in quadrate.
HWT_TIMER_2 Timer #2 in quadrate.
HWT_TIMER_3 Timer #3 in quadrate.

2.6.3.4.2 anonymous enum

Hardware Timer Modules.

Enumerator

HWT_MODULE_0 Timer Module #0.
HWT_MODULE_1 Timer Module #1.

2.6.3.4.3 enum timer32_operation_mode_t

Timer32 modes of operation.

Enumerator

SOC_TIMER32_ONE_SHOT count once
SOC_TIMER32_PERIODIC restart after a succesfull comparison
SOC_TIMER32_FREE_RUN run endlessly without comparison

2.6.3.4.4 enum timer32_counting_mode_t

SoC HW timers32 modes of counting.

Enumerator

SOC_TIMER32_DEFAULT_COUNT Default count (SOC_TIMER32_COUNT_RISING_EDG↩ E).
SOC_TIMER32_COUNT_RISING_EDGE Count rising edges of the primary source.
SOC_TIMER32_COUNT_RISING_AND_FALL_EDGE Count rising and falling edges of the primary source.
SOC_TIMER32_COUNT_QUADRATURE Count rising edges of the primary source while the secondary input is high active.
SOC_TIMER32_COUNT_PRIMARY_DIR_SECONDARY Quadrature count mode, uses primary clock and secondary input. Count rising edges of the primary clock; secondary input specifies direction (1 = minus)
SOC_TIMER32_COUNT_CASCADED Edge of the secondary input triggers primary count until a compare occurs. Cascaded counter mode (up/down).

2.6.3.4.5 enum timer32_primary_source_t

Timer32 primary sources.

Enumerator

SOC_TIMER32_TIN0SOURCE Timer source clock is TIN0.
SOC_TIMER32_TIN1SOURCE Timer source clock is TIN1.
SOC_TIMER32_TIN2SOURCE Timer source clock is TIN2.
SOC_TIMER32_TIN3SOURCE Timer source clock is TIN2.
SOC_TIMER32_OFLAG0SOURCE Timer source clock is timer 0 output.
SOC_TIMER32_OFLAG1SOURCE Timer source clock is timer 1 output.
SOC_TIMER32_OFLAG2SOURCE Timer source clock is timer 2 output.
SOC_TIMER32_OFLAG3SOURCE Timer source clock is timer 3 output.
SOC_TIMER32_PRESCALER1SOURCE Timer source clock is (input clock)/1.
SOC_TIMER32_PRESCALER2SOURCE Timer source clock is (input clock)/2.
SOC_TIMER32_PRESCALER4SOURCE Timer source clock is (input clock)/4.
SOC_TIMER32_PRESCALER8SOURCE Timer source clock is (input clock)/8.
SOC_TIMER32_PRESCALER16SOURCE Timer source clock is (input clock)/16.
SOC_TIMER32_PRESCALER32SOURCE Timer source clock is (input clock)/32.
SOC_TIMER32_PRESCALE64SOURCE Timer source clock is (input clock)/64.
SOC_TIMER32_PRESCALER128SOURCE Timer source clock is (input clock)/128.
SOC_TIMER32_NOT_IN_USE NOT IN USE.

2.6.3.4.6 enum timer32_secondary_source_t

Timer32 secondary sources.

Enumerator

SOC_TIMER32_SECONDARY_TIN0 timer input signal is taken from MUX0
SOC_TIMER32_SECONDARY_TIN1 timer input signal is taken from MUX1
SOC_TIMER32_SECONDARY_TIN2 timer input signal is taken from MUX2
SOC_TIMER32_SECONDARY_TIN3 timer input signal is taken from MUX3

2.6.3.4.7 enum timer32_count_direction_t

Timer32 primary sources.

2.6.3.4.8 enum timer32_number_in_module_t

Hardware Timer in Quandrate Module.

Enumerator

SOC_TIMER32_IN_MODULE_TIMER_0 Timer #0 in quadrate.
SOC_TIMER32_IN_MODULE_TIMER_1 Timer #1 in quadrate.
SOC_TIMER32_IN_MODULE_TIMER_2 Timer #2 in quadrate.
SOC_TIMER32_IN_MODULE_TIMER_3 Timer #3 in quadrate.

2.6.3.4.9 enum timer32_preload_ctrl_t

Hardware Timer preload control.

Enumerator

SOC_TIMER32_PRELOAD_NEVER Preloading is disabled.
SOC_TIMER32_PRELOAD_UPON_CMP1 Load preload value to compare1 upon successfull compare1 comparison.
SOC_TIMER32_PRELOAD_UPON_CMP2 Load preload value to compare2 upon successfull compare2 comparison.

2.6.3.4.10 anonymous enum

OS Hardware Timer objects values.

Hardware Timer objects can be created dynamically by using osHwTimerFind() or they can be statically referenced by the definitions below.

2.6.3.4.11 enum hw_timer_trigger_input_t

Enumerator

SOC_MUX_IN_TMR0_OUT Timer input signal is TMR0 output.
SOC_MUX_IN_TMR1_OUT Timer input signal is TMR1 output.
SOC_MUX_IN_TMR2_OUT Timer input signal is TMR2 output.
SOC_MUX_IN_TMR3_OUT Timer input signal is TMR3 output.
SOC_MUX_IN_TMR4_OUT Timer input signal is TMR4 output.
SOC_MUX_IN_TMR5_OUT Timer input signal is TMR5 output.
SOC_MUX_IN_TMR6_OUT Timer input signal is TMR5 output.
SOC_MUX_IN_TMR7_OUT Timer input signal is TMR5 output.
SOC_MUX_IN_CPRI0_RX_HFP_OUT Timer input signal is CPRI 0 Rx HFP output (66usec)
SOC_MUX_IN_CPRI1_RX_HFP_OUT Timer input signal is CPRI 1 Rx HFP output (66usec)
SOC_MUX_IN_CPRI2_RX_HFP_OUT Timer input signal is CPRI 2 Rx HFP output (66usec)
SOC_MUX_IN_CPRI3_RX_HFP_OUT Timer input signal is CPRI 3 Rx HFP output (66usec)
SOC_MUX_IN_CPRI4_RX_HFP_OUT Timer input signal is CPRI 4 Rx HFP output (66usec)
SOC_MUX_IN_CPRI5_RX_HFP_OUT Timer input signal is CPRI 5 Rx HFP output (66usec)
SOC_MUX_IN_CPRI6_RX_HFP_OUT Timer input signal is CPRI 6 Rx HFP output (66usec)
SOC_MUX_IN_CPRI7_RX_HFP_OUT Timer input signal is CPRI 7 Rx HFP output (66usec)
SOC_MUX_IN_CPRI0_TX_HFP_OUT Timer input signal is CPRI 0 Tx HFP output (66usec)
SOC_MUX_IN_CPRI1_TX_HFP_OUT Timer input signal is CPRI 1 Tx HFP output (66usec)
SOC_MUX_IN_CPRI2_TX_HFP_OUT Timer input signal is CPRI 2 Tx HFP output (66usec)
SOC_MUX_IN_CPRI3_TX_HFP_OUT Timer input signal is CPRI 3 Tx HFP output (66usec)
SOC_MUX_IN_CPRI4_TX_HFP_OUT Timer input signal is CPRI 4 Tx HFP output (66usec)
SOC_MUX_IN_CPRI5_TX_HFP_OUT Timer input signal is CPRI 5 Tx HFP output (66usec)
SOC_MUX_IN_CPRI6_TX_HFP_OUT Timer input signal is CPRI 6 Tx HFP output (66usec)
SOC_MUX_IN_CPRI7_TX_HFP_OUT Timer input signal is CPRI 7 Tx HFP output (66usec)
SOC_MUX_IN_CPRI0_RX_BFP_OUT Timer input signal is CPRI 0 Rx BFP output (260nsec)
SOC_MUX_IN_CPRI1_RX_BFP_OUT Timer input signal is CPRI 1 Rx BFP output (260nsec)
SOC_MUX_IN_CPRI2_RX_BFP_OUT Timer input signal is CPRI 2 Rx BFP output (260nsec)
SOC_MUX_IN_CPRI3_RX_BFP_OUT Timer input signal is CPRI 3 Rx BFP output (260nsec)
SOC_MUX_IN_CPRI4_RX_BFP_OUT Timer input signal is CPRI 4 Rx BFP output (260nsec)
SOC_MUX_IN_CPRI5_RX_BFP_OUT Timer input signal is CPRI 5 Rx BFP output (260nsec)
SOC_MUX_IN_CPRI6_RX_BFP_OUT Timer input signal is CPRI 6 Rx BFP output (260nsec)
SOC_MUX_IN_CPRI7_RX_BFP_OUT Timer input signal is CPRI 7 Rx BFP output (260nsec)
SOC_MUX_IN_CPRI0_TX_BFP_OUT Timer input signal is CPRI 0 Tx BFP output (260nsec)
SOC_MUX_IN_CPRI1_TX_BFP_OUT Timer input signal is CPRI 1 Tx BFP output (260nsec)
SOC_MUX_IN_CPRI2_TX_BFP_OUT Timer input signal is CPRI 2 Tx BFP output (260nsec)
SOC_MUX_IN_CPRI3_TX_BFP_OUT Timer input signal is CPRI 3 Tx BFP output (260nsec)
SOC_MUX_IN_CPRI4_TX_BFP_OUT Timer input signal is CPRI 4 Tx BFP output (260nsec)
SOC_MUX_IN_CPRI5_TX_BFP_OUT Timer input signal is CPRI 5 Tx BFP output (260nsec)
SOC_MUX_IN_CPRI6_TX_BFP_OUT Timer input signal is CPRI 6 Tx BFP output (260nsec)
SOC_MUX_IN_CPRI7_TX_BFP_OUT Timer input signal is CPRI 7 Tx BFP output (260nsec)
SOC_MUX_IN_CPRI0_RX_RF_OUT Timer input signal is CPRI 0 Rx Radio Frame output

(10msec)

SOC_MUX_IN_CPRI1_RX_RF_OUT Timer input signal is CPRI 1 Rx Radio Frame output (10msec)

SOC_MUX_IN_CPRI2_RX_RF_OUT Timer input signal is CPRI 2 Rx Radio Frame output (10msec)

SOC_MUX_IN_CPRI3_RX_RF_OUT Timer input signal is CPRI 3 Rx Radio Frame output (10msec)

SOC_MUX_IN_CPRI4_RX_RF_OUT Timer input signal is CPRI 4 Rx Radio Frame output (10msec)

SOC_MUX_IN_CPRI5_RX_RF_OUT Timer input signal is CPRI 5 Rx Radio Frame output (10msec)

SOC_MUX_IN_CPRI6_RX_RF_OUT Timer input signal is CPRI 6 Rx Radio Frame output (10msec)

SOC_MUX_IN_CPRI7_RX_RF_OUT Timer input signal is CPRI 7 Rx Radio Frame output (10msec)

SOC_MUX_IN_CPRI0_TX_RF_OUT Timer input signal is CPRI 0 Tx Radio Frame output (10msec)

SOC_MUX_IN_CPRI1_TX_RF_OUT Timer input signal is CPRI 1 Tx Radio Frame output (10msec)

SOC_MUX_IN_CPRI2_TX_RF_OUT Timer input signal is CPRI 2 Tx Radio Frame output (10msec)

SOC_MUX_IN_CPRI3_TX_RF_OUT Timer input signal is CPRI 3 Tx Radio Frame output (10msec)

SOC_MUX_IN_CPRI4_TX_RF_OUT Timer input signal is CPRI 4 Tx Radio Frame output (10msec)

SOC_MUX_IN_CPRI5_TX_RF_OUT Timer input signal is CPRI 5 Tx Radio Frame output (10msec)

SOC_MUX_IN_CPRI6_TX_RF_OUT Timer input signal is CPRI 6 Tx Radio Frame output (10msec)

SOC_MUX_IN_CPRI7_TX_RF_OUT Timer input signal is CPRI 7 Tx Radio Frame output (10msec)

SOC_MUX_IN_SYS_CLOCK Timer input signal is system clock.

SOC_MUX_IN_1588_1 Timer input signal is 1588_1.

SOC_MUX_IN_1588_2 Timer input signal is 1588_2.

SOC_MUX_IN_CPRI0_RX_TRANSACTION Timer input signal is CPRI 0 Rx DMA transaction count.

SOC_MUX_IN_CPRI1_RX_TRANSACTION Timer input signal is CPRI 1 Rx DMA transaction count.

SOC_MUX_IN_CPRI2_RX_TRANSACTION Timer input signal is CPRI 2 Rx DMA transaction count.

SOC_MUX_IN_CPRI3_RX_TRANSACTION Timer input signal is CPRI 3 Rx DMA transaction count.

SOC_MUX_IN_CPRI4_RX_TRANSACTION Timer input signal is CPRI 4 Rx DMA transaction count.

SOC_MUX_IN_CPRI5_RX_TRANSACTION Timer input signal is CPRI 5 Rx DMA transaction

count.

SOC_MUX_IN_CPRI6_RX_TRANSACTION Timer input signal is CPRI 6 Rx DMA transaction count.

SOC_MUX_IN_CPRI7_RX_TRANSACTION Timer input signal is CPRI 7 Rx DMA transaction count.

SOC_MUX_IN_CPRI0_TX_TRANSACTION Timer input signal is CPRI 0 Tx DMA transaction count.

SOC_MUX_IN_CPRI1_TX_TRANSACTION Timer input signal is CPRI 1 Tx DMA transaction count.

SOC_MUX_IN_CPRI2_TX_TRANSACTION Timer input signal is CPRI 2 Tx DMA transaction count.

SOC_MUX_IN_CPRI3_TX_TRANSACTION Timer input signal is CPRI 3 Tx DMA transaction count.

SOC_MUX_IN_CPRI4_TX_TRANSACTION Timer input signal is CPRI 4 Tx DMA transaction count.

SOC_MUX_IN_CPRI5_TX_TRANSACTION Timer input signal is CPRI 5 Tx DMA transaction count.

SOC_MUX_IN_CPRI6_TX_TRANSACTION Timer input signal is CPRI 6 Tx DMA transaction count.

SOC_MUX_IN_CPRI7_TX_TRANSACTION Timer input signal is CPRI 7 Tx DMA transaction count.

SOC_MUX_IN_CPRI0_RX2_TRANSACTION Timer input signal is CPRI 0 Rx DMA transaction count.

SOC_MUX_IN_CPRI1_RX2_TRANSACTION Timer input signal is CPRI 0 Rx DMA transaction count.

SOC_MUX_IN_CPRI2_RX2_TRANSACTION Timer input signal is CPRI 0 Rx DMA transaction count.

SOC_MUX_IN_CPRI3_RX2_TRANSACTION Timer input signal is CPRI 0 Rx DMA transaction count.

SOC_MUX_IN_CPRI4_RX2_TRANSACTION Timer input signal is CPRI 0 Rx DMA transaction count.

SOC_MUX_IN_CPRI5_RX2_TRANSACTION Timer input signal is CPRI 0 Rx DMA transaction count.

SOC_MUX_IN_CPRI6_RX2_TRANSACTION Timer input signal is CPRI 0 Rx DMA transaction count.

SOC_MUX_IN_CPRI7_RX2_TRANSACTION Timer input signal is CPRI 0 Rx DMA transaction count.

2.6.3.4.12 enum timer_clock_input_t

Clocking B4860 specific, clock input for SoC timers.

Enumerator

SOC_CLOCK_IN_DISABLED timer module is disabled

SOC_CLOCK_IN_PLAFORM2 platform timer diveded by 2 (333Mhz)
SOC_CLOCK_IN_CPRI_PLL1 122.8Mhz, CPRI clock SERDES PLL1
SOC_CLOCK_IN_CPRI_PLL2 122.8Mhz, CPRI clock SERDES PLL2
SOC_CLOCK_IN_CPRI_RECOVERY_PLL1 122.8Mhz, CPRI external recovered clock from PLL1
SOC_CLOCK_IN_CPRI_RECOVERY_PLL2 122.8Mhz, CPRI external recovered clock from PLL2

2.6.3.4.13 enum soc_timer32_num_t

SoC HW timers IDs.

Enumerator

SOC_TIMER32_0 first group, first module, first timer
SOC_TIMER32_1 first group, first module, second timer
SOC_TIMER32_2 first group, first module, third timer
SOC_TIMER32_3 first group, first module, forth timer
SOC_TIMER32_4 first group, second module, first timer
SOC_TIMER32_5 first group, second module, second timer
SOC_TIMER32_6 first group, second module, third timer
SOC_TIMER32_7 first group, second module, forth timer
SOC_TIMER32_8 first group, third module, first timer
SOC_TIMER32_9 first group, third module, second timer
SOC_TIMER32_10 first group, third module, third timer
SOC_TIMER32_11 first group, third module, forth timer
SOC_TIMER32_12 first group, forth module, first timer
SOC_TIMER32_13 first group, forth module, second timer
SOC_TIMER32_14 first group, forth module, third timer
SOC_TIMER32_15 first group, forth module, forth timer
SOC_TIMER32_16 second group, first module, first timer
SOC_TIMER32_17 second group, first module, second timer
SOC_TIMER32_18 second group, first module, third timer
SOC_TIMER32_19 second group, first module, forth timer
SOC_TIMER32_20 second group, second module, first timer
SOC_TIMER32_21 second group, second module, second timer
SOC_TIMER32_22 second group, second module, third timer
SOC_TIMER32_23 second group, second module, forth timer
SOC_TIMER32_24 second group, third module, first timer
SOC_TIMER32_25 second group, third module, second timer
SOC_TIMER32_26 second group, third module, third timer
SOC_TIMER32_27 second group, third module, forth timer
SOC_TIMER32_28 second group, forth module, first timer
SOC_TIMER32_29 second group, forth module, second timer

SOC_TIMER32_30	second group, forth module, third timer
SOC_TIMER32_31	second group, forth module, forth timer
SOC_TIMER32_32	third group, first module, first timer
SOC_TIMER32_33	third group, first module, second timer
SOC_TIMER32_34	third group, first module, third timer
SOC_TIMER32_35	third group, first module, forth timer
SOC_TIMER32_36	third group, second module, first timer
SOC_TIMER32_37	third group, second module, second timer
SOC_TIMER32_38	third group, second module, third timer
SOC_TIMER32_39	third group, second module, forth timer
SOC_TIMER32_40	third group, third module, first timer
SOC_TIMER32_41	third group, third module, second timer
SOC_TIMER32_42	third group, third module, third timer
SOC_TIMER32_43	third group, third module, forth timer
SOC_TIMER32_44	third group, forth module, first timer
SOC_TIMER32_45	third group, forth module, second timer
SOC_TIMER32_46	third group, forth module, third timer
SOC_TIMER32_47	third group, forth module, forth timer
SOC_TIMER32_48	forth group, first module, first timer
SOC_TIMER32_49	forth group, first module, second timer
SOC_TIMER32_50	forth group, first module, third timer
SOC_TIMER32_51	forth group, first module, forth timer
SOC_TIMER32_52	forth group, second module, first timer
SOC_TIMER32_53	forth group, second module, second timer
SOC_TIMER32_54	forth group, second module, third timer
SOC_TIMER32_55	forth group, second module, forth timer
SOC_TIMER32_56	forth group, third module, first timer
SOC_TIMER32_57	forth group, third module, second timer
SOC_TIMER32_58	forth group, third module, third timer
SOC_TIMER32_59	forth group, third module, forth timer
SOC_TIMER32_60	forth group, forth module, first timer
SOC_TIMER32_61	forth group, forth module, second timer
SOC_TIMER32_62	forth group, forth module, third timer
SOC_TIMER32_63	forth group, forth module, forth timer
SOC_TIMER32_64	fifth group, first module, first timer
SOC_TIMER32_65	fifth group, first module, second timer
SOC_TIMER32_66	fifth group, first module, third timer
SOC_TIMER32_67	fifth group, first module, forth timer
SOC_TIMER32_68	fifth group, second module, first timer
SOC_TIMER32_69	fifth group, second module, second timer
SOC_TIMER32_70	fifth group, second module, third timer
SOC_TIMER32_71	fifth group, second module, forth timer
SOC_TIMER32_72	fifth group, third module, first timer
SOC_TIMER32_73	fifth group, third module, second timer
SOC_TIMER32_74	fifth group, third module, third timer

SOC_TIMER32_75 fifth group, third module, forth timer
SOC_TIMER32_76 fifth group, forth module, first timer
SOC_TIMER32_77 fifth group, forth module, second timer
SOC_TIMER32_78 fifth group, forth module, third timer
SOC_TIMER32_79 fifth group, forth module, forth timer
SOC_TIMER32_80 sixth group, first module, first timer
SOC_TIMER32_81 sixth group, first module, second timer
SOC_TIMER32_82 sixth group, first module, third timer
SOC_TIMER32_83 sixth group, first module, forth timer
SOC_TIMER32_84 sixth group, second module, first timer
SOC_TIMER32_85 sixth group, second module, second timer
SOC_TIMER32_86 sixth group, second module, third timer
SOC_TIMER32_87 sixth group, second module, forth timer
SOC_TIMER32_88 sixth group, third module, first timer
SOC_TIMER32_89 sixth group, third module, second timer
SOC_TIMER32_90 sixth group, third module, third timer
SOC_TIMER32_91 sixth group, third module, forth timer
SOC_TIMER32_92 sixth group, forth module, first timer
SOC_TIMER32_93 sixth group, forth module, second timer
SOC_TIMER32_94 sixth group, forth module, third timer
SOC_TIMER32_95 sixth group, forth module, forth timer
SOC_TIMER32_96 seventh group, first module, first timer
SOC_TIMER32_97 seventh group, first module, second timer
SOC_TIMER32_98 seventh group, first module, third timer
SOC_TIMER32_99 seventh group, first module, forth timer
SOC_TIMER32_100 seventh group, second module, first timer
SOC_TIMER32_101 seventh group, second module, second timer
SOC_TIMER32_102 seventh group, second module, third timer
SOC_TIMER32_103 seventh group, second module, forth timer
SOC_TIMER32_104 seventh group, third module, first timer
SOC_TIMER32_105 seventh group, third module, second timer
SOC_TIMER32_106 seventh group, third module, third timer
SOC_TIMER32_107 seventh group, third module, forth timer
SOC_TIMER32_108 seventh group, forth module, first timer
SOC_TIMER32_109 seventh group, forth module, second timer
SOC_TIMER32_110 seventh group, forth module, third timer
SOC_TIMER32_111 seventh group, forth module, forth timer
SOC_TIMER32_112 eighth group, first module, first timer
SOC_TIMER32_113 eighth group, first module, second timer
SOC_TIMER32_114 eighth group, first module, third timer
SOC_TIMER32_115 eighth group, first module, forth timer
SOC_TIMER32_116 eighth group, second module, first timer
SOC_TIMER32_117 eighth group, second module, second timer
SOC_TIMER32_118 eighth group, second module, third timer
SOC_TIMER32_119 eighth group, second module, forth timer

SOC_TIMER32_120 eighth group, third module, first timer
SOC_TIMER32_121 eighth group, third module, second timer
SOC_TIMER32_122 eighth group, third module, third timer
SOC_TIMER32_123 eighth group, third module, forth timer
SOC_TIMER32_124 eighth group, forth module, first timer
SOC_TIMER32_125 eighth group, forth module, second timer
SOC_TIMER32_126 eighth group, forth module, third timer
SOC_TIMER32_127 eighth group, forth module, forth timer

2.6.3.4.14 enum soc_timer32_module_t

TIMER32 module IDs.

Enumerator

SOC_TIMER32_MODULE0 first module in first group
SOC_TIMER32_MODULE1 second module in first group
SOC_TIMER32_MODULE2 third module in first group
SOC_TIMER32_MODULE3 forth module in first group
SOC_TIMER32_MODULE4 first module in second group
SOC_TIMER32_MODULE5 second module in second group
SOC_TIMER32_MODULE6 third module in second group
SOC_TIMER32_MODULE7 forth module in second group
SOC_TIMER32_MODULE8 first module in third group
SOC_TIMER32_MODULE9 second module in third group
SOC_TIMER32_MODULE10 third module in third group
SOC_TIMER32_MODULE11 forth module in third group
SOC_TIMER32_MODULE12 first module in forth group
SOC_TIMER32_MODULE13 second module in forth group
SOC_TIMER32_MODULE14 third module in forth group
SOC_TIMER32_MODULE15 forth module in forth group
SOC_TIMER32_MODULE16 first module in fifth group
SOC_TIMER32_MODULE17 second module in fifth group
SOC_TIMER32_MODULE18 third module in fifth group
SOC_TIMER32_MODULE19 forth module in fifth group
SOC_TIMER32_MODULE20 first module in sixth group
SOC_TIMER32_MODULE21 second module in sixth group
SOC_TIMER32_MODULE22 third module in sixth group
SOC_TIMER32_MODULE23 forth module in sixth group
SOC_TIMER32_MODULE24 first module in seventh group
SOC_TIMER32_MODULE25 second module in seventh group
SOC_TIMER32_MODULE26 third module in seventh group
SOC_TIMER32_MODULE27 forth module in seventh group
SOC_TIMER32_MODULE28 first module in eighth group

SOC_TIMER32_MODULE29	second module in eighth group
SOC_TIMER32_MODULE30	third module in eighth group
SOC_TIMER32_MODULE31	forth module in eighth group

2.6.3.4.15 enum soc_timer32_group_t

TIMER32 groups.

Enumerator

SOC_TIMER32_GROUP0	TIMER32 group 0.
SOC_TIMER32_GROUP1	TIMER32 group 1.
SOC_TIMER32_GROUP2	TIMER32 group 2.
SOC_TIMER32_GROUP3	TIMER32 group 3.
SOC_TIMER32_GROUP4	TIMER32 group 4.
SOC_TIMER32_GROUP5	TIMER32 group 5.
SOC_TIMER32_GROUP6	TIMER32 group 6.
SOC_TIMER32_GROUP7	TIMER32 group 7.

2.6.3.4.16 enum soc_timer32_maple_t

TIMER32 MPALE IDs.

Enumerator

SOC_TIMER32_MAPLE0	MAPLE 0.
SOC_TIMER32_MAPLE1	MPALE 1.
SOC_TIMER32_MAPLE2	MAPLE 2.

2.6.3.5 Function Documentation

2.6.3.5.1 uint32_t osHwTimerGlobalGet (unsigned int *module*)

Returns the global timer in the module.

Should be called after osHwTimerGlobalConfig(). Works on 32 bit SoC timers

Parameters

in	*module*	- Number of module, HWT_MODULE_0 or HWT_MODULE_1

Returns

Global system timer value

2.6.3.5.2 os_status osHwTimerGlobalConfig (unsigned int *module,* unsigned int *timers,* uint32_t *hold0,* uint32_t *hold1,* uint32_t *hold2*)

Configures the global system timer.

Works on 32 bit SoC timers

Parameters

in	*module*	- Number of module, HWT_MODULE_0 or HWT_MODULE_1
in	*timers*	- Bitmask of timers to lock (e.g. HWT_TIMER_0 | HWT_TIMER_3)
in	*hold0*	- Range starting from from 0, of bits to hold
in	*hold1*	- Range starting from from hold0, of bits to hold
in	*hold2*	- Range starting from from hold1, of bits to hold

Return values

OS_SUCCESS	
OS_ERR_TM_MODE_INVALID	- The combination of timers and holds requires more than 32 bits

Warning

If one fo the hold0-hold2 are not in use, must set to 0

2.6.3.5.3 os_status osHwTimerLock (unsigned int *module,* unsigned int *timers*)

Locks the specified timers in the specified module.

Works on 32 bit SoC timers

Parameters

in	*module*	- Number of module, HWT_MODULE_0 or HWT_MODULE_1
in	*timers*	- Bitmask of timers to lock (e.g. HWT_TIMER_0 | HWT_TIMER_3)

Return values

OS_SUCCESS	

2.6.3.5.4 os_status osHwTimerUnlock (unsigned int *module,* unsigned int *timers*)

Unlocks the specified timers in the specified module.

Works on 32 bit SoC timers

Parameters

in	*module*	- Number of module, HWT_MODULE_0 or HWT_MODULE_1
in	*timers*	- Bitmask of timers to lock (e.g. HWT_TIMER_0 | HWT_TIMER_3)

Return values

OS_SUCCESS	

2.6.3.5.5 os_status hwTimer32Open (soc_timer32_num_t *timer_num,* timer32_open_params_t * *params*)

Open an hardware timer in an initialized module.

Parameters

in	*timer_num*	- The hardware timer number.
in	*params*	- timer initization parameters.

Return values

OS_SUCCESS	- The hardware timer was successfully created.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.

OS_ERR_HW_TM_MODE_INVALID	- Invalid or unsupported hardware timer mode.
OS_ERR_HW_TM_INTERVAL_UNSUPPORTED	- Unsupported interval.
OS_ERR_HW_TM_SOURCE_INVALID	- Invalid source clock.
OS_ERR_HW_TM_PRIORITY_INVALID	- Invalid HWI priority.
OS_ERR_HW_TM_ALREADY_CREATED	- The hardware timer is already created.

2.6.3.5.6 os_status hwTimer32SetInterval (soc_timer32_num_t *timer_num,* os_timer_interval *interval*)

set interval for an hardware timer32. valid for stopped timers only

Parameters

in	*timer_num*	- The hardware timer number.
in	*interval*	- The timer interval.

Return values

OS_SUCCESS	- The interval was successfully set.
OS_ERR_HW_TM_INVALID	- Invalid hardware timer number.
OS_ERR_HW_TM_INTERVAL_UNSUPPORTED	- Unsupported interval.
OS_ERR_HW_TM_NOT_CREATED	- The hardware timer was not created, or it was already deleted.

2.6.3.5.7 os_status hwTimer32Delete (soc_timer32_num_t *timer_num*)

Stops and deletes the given hardware timer32 from the system. valid for open timers only

Parameters

in	*timer_num*	- The hardware timer number.

Return values

OS_SUCCESS	- The hardware timer was successfully deleted.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.
OS_ERR_HW_TM_NO↩ T_CREATED	- The hardware timer was not created, or it was already deleted.

2.6.3.5.8 os_status hwTimer32Start (soc_timer32_num_t *timer_num*)

Starts the given hardware timer32. valid for non-running open timers.

Parameters

in	*timer_num*	- The hardware timer number.

Return values

OS_SUCCESS	- The hardware timer was successfully started.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.
OS_ERR_HW_TM_NO↩ T_CREATED	- The hardware timer was not created, or it was already deleted.

2.6.3.5.9 os_status hwTimer32Stop (soc_timer32_num_t *timer_num*)

Stops the given hardware timer32. valid for running timers.

Parameters

in	*timer_num*	- The hardware timer number.

Return values

OS_SUCCESS	- The hardware timer was successfully stopped.

OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.
OS_ERR_HW_TM_NO↩ T_CREATED	- The hardware timer was not created, or it was already deleted.

2.6.3.5.10 void hwTimer32ClearEvent (soc_timer32_num_t *timer_num*)

Clears the event bit of the given hardware timer. Application must clear timer event for normal timer operation.

Parameters

in	*timer_num*	- The hardware timer number.

2.6.3.5.11 os_status hwTimer32ValueGet (soc_timer32_num_t *timer_num*, os_timer_interval * *value*)

Read the counter of the given hardware timer32.

Parameters

in	*timer_num*	- The hardware timer number.
out	*value*	- The hardware timer value

Return values

OS_SUCCESS	- Succeeded in reading the value.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.

2.6.3.5.12 os_status hwTimer32HoldGet (soc_timer32_num_t *timer_num*, uint32_t * *value*)

Get the hold value of a given hardware timer32.

Parameters

in	*timer_num*	- The hardware timer number.
out	*value*	- The hardware hold value

Return values

OS_SUCCESS	- Succeeded in reading the value.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.

2.6.3.5.13 os_status hwTimer32CounterSet (soc_timer32_num_t *timer_num,* uint32_t *value*)

Set a value for a counter of a given hardware timer32. Any 32bit value is valid. Valid for open and running timers

Parameters

in	*timer_num*	- The hardware timer number.
in	*value*	- the load value to set

Return values

OS_SUCCESS	- Succeeded in reading the value.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.

2.6.3.5.14 os_status hwTimer32PreloadSet (soc_timer32_num_t *timer_num,* uint32_t *preload_value*)

Set preload value to a given hardware timer32. Any 32bit value is valid. preload value is set according to the counting direction defined in the hwTimer32Open call.

Parameters

in	*timer_num*	- the hardware timer number.
in	*preload_value*	- preload value to set

Return values

OS_SUCCESS	- Succeeded in reading the value.

OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.

2.6.3.5.15 os_status hwTimer32CompareSet (soc_timer32_num_t *timer_num*, uint32_t *compare*)

Set compare values to a given hardware timer32. Any 32bit value is valid. compare value is set according to the counting direction defined in the hwTimer32Open call.

Parameters

in	*timer_num*	- The hardware timer number.
in	*compare*	- compare value to set

Return values

OS_SUCCESS	- Succeeded in reading the value.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.

2.6.3.5.16 os_status hwTimer32SetAndForget (soc_timer32_module_t *module*, timer32_input_select_t *timer_input*)

Lock timer on high input signal.

Parameters

in	*module*	- module to lock on the selected trigger.
in	*timer_input*	- The mux input on which timer will be locked.

Return values

OS_SUCCESS	- Succeeded in reading the value.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.

2.6.3.5.17 os_status hwTimer32LockClear (soc_timer32_module_t *module*, timer32_input_select_t *timer_input*)

release lock of hardware timer32. valid for open and running timers.

Parameters

in	*module*	- module to unlock.
in	*timer_input*	- The locked mux input.

Return values

OS_SUCCESS	- Succeeded in reading the value.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.

2.6.3.5.18 os_status hwTimer32GlobalConfig (soc_timer32_module_t *module*, timer32_number_in_module_t *timers*, uint32_t *hold0*, uint32_t *hold1*, uint32_t *hold2*)

Configures the global system timer.

Works on 32 bit SoC timers

Parameters

in	*module*	- Number of module, HWT_MODULE_0 or HWT_MODULE_1
in	*timers*	- Bitmask of timers to lock (e.g. SOC_TIMER32_IN_MODULE_TI↩ MER_0 | SOC_TIMER32_IN_MODULE_TIMER_3)
in	*hold0*	- Range starting from from 0, of bits to hold
in	*hold1*	- Range starting from from hold0, of bits to hold
in	*hold2*	- Range starting from from hold1, of bits to hold

Return values

OS_SUCCESS	
OS_ERR_TM_MODE_I↩ NVALID	- The combination of timers and holds requires more than 32 bits

Warning

If one fo the hold0-hold2 are not in use, must set to 0

2.6.3.5.19 uint32_t hwTimer32GlobalGet (soc_timer32_module_t *module*)

Returns the global timer in the module.

Should be called after hwTimer32GlobalConfig(). Works on 32 bit SoC timers

Parameters

in	*module*	- Number of module, SOC_TIMER32_MODULEx (arhcitecture dependent)

Returns

Global system timer value

2.6.3.5.20 os_status hwTimer32SetInitiatorMode (soc_timer32_module_t *module*, timer32_number_in_module_t *initiator_timer*, uint32_t *reinit_timers*, uint32_t *output_signal_timers*)

Configures broadcast from an Initiator Timer Works on 32 bit SoC timers

Parameters

in	*module*	- Number of module, HWT_MODULE_0 or HWT_MODULE_1
in	*initiator_timer*	- Selected timer to be the initiator timer (if more than one timer is selcted only the lowest timer will be used)
in	*reinit_timers*	- Bitmask of other timers to be reinitialized when a compare event occurs on the initiator timer(e.g. SOC_TIMER32_IN_MODULE_TIMER_0 | SOC_TIMER32_IN_MODULE_TIMER_3)
in	*output_signal_timers*	- Bitmask of other timers to forces their output flag signal when a compare event occurs on the initiator timer (e.g. SOC_TIMER32_IN_MODULE_TIMER_0 | SOC_TIMER32_IN_MODULE_TIMER_3)

Return values

OS_SUCCESS	
OS_ERR_TM_MODE_INVALID	- The combination of timers and holds requires more than 32 bits

Warning

The selected initiator timer cannot be selected as part of

2.6.3.5.21 os_status osHwTimerCreate (os_timer_handle *timer_num*, os_timer_mode *mode*, os_timer_interval *interval*, os_hw_timer_source *source*, os_hwi_function *handler*, os_hwi_priority *priority*)

Creates a hardware timer and sets its parameters.

Parameters

in	*timer_num*	- The hardware timer number.
in	*mode*	- Timer mode (one-shot, periodic, etc.).
in	*interval*	- The timer interval.
in	*source*	- The source clock for the timer.
in	*handler*	- The function to call when the timer expires.
in	*priority*	- The hardware timer priority.

Return values

OS_SUCCESS	- The hardware timer was successfully created.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.
OS_ERR_HW_TM_MO↩ DE_INVALID	- Invalid or unsupported hardware timer mode.
OS_ERR_HW_TM_INT↩ ERVAL_UNSUPPORT↩ ED	- Unsupported interval.
OS_ERR_HW_TM_SO↩ URCE_INVALID	- Invalid source clock.
OS_ERR_HW_TM_PRI↩ ORITY_INVALID	- Invalid HWI priority.
OS_ERR_HW_TM_ALR↩ EADY_CREATED	- The hardware timer is already created.

2.6.3.5.22 os_status osHwTimerSetInterval (os_timer_handle *timer_num,* os_timer_interval *interval*)

Creates a hardware timer and sets its parameters.

Parameters

in	*timer_num*	- The hardware timer number.
in	*interval*	- The timer interval.

Return values

OS_SUCCESS	- The interval was successfully set.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.

OS_ERR_HW_TM_INT↩ ERVAL_UNSUPPORT↩ ED	- Unsupported interval.
OS_ERR_HW_TM_NO↩ T_CREATED	- The hardware timer was not created, or it was already deleted.

2.6.3.5.23 os_status osHwTimerDelete (os_timer_handle *timer_num*)

Stops and deletes the given hardware timer from the system.

Parameters

in	*timer_num*	- The hardware timer number.

Return values

OS_SUCCESS	- The hardware timer was successfully deleted.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.
OS_ERR_HW_TM_NO↩ T_CREATED	- The hardware timer was not created, or it was already deleted.

2.6.3.5.24 os_status osHwTimerStart (os_timer_handle *timer_num*)

Starts the given hardware timer.

Parameters

in	*timer_num*	- The hardware timer number.

Return values

OS_SUCCESS	- The hardware timer was successfully started.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.
OS_ERR_HW_TM_NO↩ T_CREATED	- The hardware timer was not created, or it was already deleted.

2.6.3.5.25 os_status osHwTimerStop (os_timer_handle *timer_num*)

Stops the given hardware timer.

Parameters

in	*timer_num*	- The hardware timer number.

Return values

OS_SUCCESS	- The hardware timer was successfully stopped.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.
OS_ERR_HW_TM_NO↩ T_CREATED	- The hardware timer was not created, or it was already deleted.

2.6.3.5.26 void osHwTimerClearEvent (os_timer_handle *timer_num*)

Clears the event bit of the given hardware timer.

Parameters

in	*timer_num*	- The hardware timer number.

2.6.3.5.27 os_status osHwTimerFind (os_timer_handle * *timer_num*)

Finds the first available hardware timer number.

Parameters

out	*timer_num*	- Receives the hardware timer number.

Return values

OS_SUCCESS	- Found an available hardware timer number.
OS_ERR_HW_TM_UN↩ AVAILABLE	- No hardware timer number is available.

2.6.3.5.28 os_status osHwTimerTypeFind (os_timer_handle * *timer_num,* int *type*)

Finds the first available hardware timer number of a specific type.

Parameters

out	*timer_num*	- Receives the hardware timer number.
out	*type*	- Type of timer to find - architecture dependent.

Return values

OS_SUCCESS	- Found an available hardware timer number.
OS_ERR_HW_TM_UNAVAILABLE	- No hardware timer number is available.

Warning

Not all architectures support this API or support types other than 0

2.6.3.5.29 os_status osHwTimerFindForce (os_timer_handle *timer_num*)

Allocates a given hardware timer number.

Parameters

in	*timer_num*	- The hardware timer number.

Return values

OS_SUCCESS	- The hardware timer was successfully allocated.
OS_ERR_HW_TM_INVALID	- Invalid hardware timer number.
OS_ERR_HW_TM_UNAVAILABLE	- The hardware timer number is not available (already allocated).

2.6.3.5.30 os_status osHwTimerValueGet (os_timer_handle *timer_num,* os_timer_interval * *value*)

Return the hardware timer value.

Parameters

in	*timer_num*	- The hardware timer number.
out	*value*	- The hardware timer value

Return values

OS_SUCCESS	- Succeeded in reading the value.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.

2.6.3.5.31 void osHwTimerSyncStart (uint32_t *core_mask*)

Start a synchronized run.

Parameters

out	*core_mask*	- The synchronized cores mask. In order to start all six cores timers together, one should write 0x0000003F to this register.

2.6.3.5.32 void osHwTimerSyncStop (uint32_t *core_mask*)

Stop a synchronized run.

Parameters

out	*core_mask*	- The synchronized cores mask. In order to stop all six cores timers together, one should write 0x0000003F to this register.

2.6.3.5.33 os_status osHwTimerFreeze (os_timer_handle *timer_num*)

Freeze the hardware timer run.

Parameters

in	*timer_num*	- The hardware timer number.

Return values

OS_SUCCESS	- Succeeded to freezing the timer.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.
OS_ERR_HW_TM_NO↩ T_CREATED	- The hardware timer was not created, or it was already deleted.

2.6.3.5.34 os_status osHwTimerUnfreeze (os_timer_handle *timer_num*)

Unfreeze the hardware timer run.

Parameters

in	*timer_num*	- The hardware timer number.

Return values

OS_SUCCESS	- Succeeded to unfreezing the timer.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.
OS_ERR_HW_TM_NO↩ T_CREATED	- The hardware timer was not created, or it was already deleted.

2.6.3.5.35 os_status hwTimer32MapleInterruptAssign (soc_timer32_num_t *timer_num,* soc_timer32_maple_t *maple_num*)

Assign timer interrupts to MAPLE. NOTE: User can avoid using this function by rising maple_ie flag when calling hwTimer32Open().

Parameters

in	*timer_num*	- the timer number of the timer which is trigerring the interrupt
in	*maple_num*	- The MAPLE(0-2) which interrupts are assigned to

Return values

OS_SUCCESS	- The hardware interrupt was successfully registered to MAPLE.

2.6.3.6 Hardware Timers

2.6.4 CORE Watchdogs API

2.6.4.1 Overview

Hardware Watchdogs setup and control.

Enumerations

- enum watchdog_wrs_mode {
  GENERATE_ONLY_NMI = 0, GENERATE_MPIC_INTERRUPT = 1, GENERATE_RESET_↩
  REQ_B = 2,
  GENERATE_AUTO_RESET = 3 }

Functions

- os_status osCoreWatchdogSupport (uint64_t timer_count, watchdog_wrs_mode wrs_second_↩
  expiration, os_swi_priority priority, os_hwi_function nmi_handler, os_hwi_arg argument, uint32_t
  tick_parameter)
- os_status osCoreWatchdogCreate (uint64_t timer_count, watchdog_wrs_mode wrs_second_↩
  expiration, os_hwi_function nmi_handler, os_hwi_arg argument)
- void osCoreWatchdogEnable ()
- void osCoreWatchdogDisable ()
- void osCoreWatchdogService ()
- uint64_t osCoreWatchdogCounterGet ()

SC39xx Watchdogs Handling Modes

- #define OS_WDT_AUTO_HANDLING 0
- #define OS_WDT_USER_HANDLING 1
- #define OS_WDT_DISABLED 2

SC39xx Watchdogs registers configuration MACROs

- #define OS_CORE_WDT_AUTO_HANDLING 0
- #define OS_CORE_WDT_USER_HANDLING 1
- #define **WRS_SECOND_EXP_BIT_OFFSET**(mode) (uint32_t)((mode)<<8)
- #define **OS_CORE_WATCHDOG_WDTC_TEN** 0x00000001
- #define **OS_CORE_WATCHDOG_WDTC_INTEN** 0x00000040

2.6.4.2 Macro Definition Documentation

2.6.4.2.1 #define OS_WDT_AUTO_HANDLING 0

watchdog timers automatic handle mode

2.6.4.2.2 #define OS_WDT_USER_HANDLING 1

watchdog timers manual handle mode

2.6.4.2.3 #define OS_WDT_DISABLED 2

watchdog timers disable mode

2.6.4.2.4 #define OS_CORE_WDT_AUTO_HANDLING 0

core watchdog timers automatic handling mode

2.6.4.2.5 #define OS_CORE_WDT_USER_HANDLING 1

core watchdog timers manual handling mode

2.6.4.3 Enumeration Type Documentation

2.6.4.3.1 enum watchdog_wrs_mode

SC39xx Watchdogs WRS modes.

Enumerator

GENERATE_ONLY_NMI WDT only serve NMI.
GENERATE_MPIC_INTERRUPT Internal interrupt sent over to the MPIC.
GENERATE_RESET_REQ_B Internal interrupt sent over to the MPIC and device output RESE↩ T_REQ_B is asserted to external world.
GENERATE_AUTO_RESET Internal interrupt sent over to the MPIC and automatic reset of that core.

2.6.4.4 Function Documentation

2.6.4.4.1 os_status osCoreWatchdogSupport (uint64_t *timer_count,* watchdog_wrs_mode *wrs_second_expiration,* os_swi_priority *priority,* os_hwi_function *nmi_handler,* os_hwi_arg *argument,* uint32_t *tick_parameter*)

Sets the SC39xx watchdog to be served by the OS. Configure and reserved SOC timer and SW interrupt for this cause. After this function the watchdog timer start counting.

Parameters

timer_count	- The timeout period of the watchdog, in system clocks.
priority	- The SW Interrupt priority of the watchdog service handler.
wrs_second_↩ expiration	- Watchdog operation mode. Permitted values: GENERATE_ONLY_NMI - WDT only serve NMI. GENERATE_MPIC_INTERRUPT - Internal interrupt sent over to the MPIC. GENERATE_RESET_REQ_B - Internal interrupt sent over to the MPIC and device output RESET_REQ_B is asserted to external world. GENERATE_A↩ UTO_RESET - Internal interrupt sent over to the MPIC and automatic reset of that core.
nmi_handler	- optional interrupt function (ISR), relevant only if mode is GENERATE_NMI. if mode is GENERATE_RESET or if there is no specific handler, this parameter must be NULL.
argument	- optional argument to be passed to the interrupt handler function.
tick_parameter	- number ticks in a second

Return values

OS_SUCCESS	- the watchdog SWI and timer were initialized successfully.
OS_FAIL	- the OS failed to initialize the watchdog SWI and timer.

2.6.4.4.2 os_status osCoreWatchdogCreate (uint64_t *timer_count,* watchdog_wrs_mode *wrs_second_expiration,* os_hwi_function *nmi_handler,* os_hwi_arg *argument*)

Sets the core watchdog parameters. The watchdog is not enabled until osCoreWatchdogEnable() is called.

Parameters

timer_count	- The timeout period of the watchdog, in system clocks.
wrs_second_↩ expiration	- Watchdog operation mode. Permitted values: GENERATE_ONLY_NMI - WDT only serve NMI. GENERATE_MPIC_INTERRUPT - Internal interrupt sent over to the MPIC. GENERATE_RESET_REQ_B - Internal interrupt sent over to the MPIC and device output RESET_REQ_B is asserted to external world. GENERATE_A↩ UTO_RESET - Internal interrupt sent over to the MPIC and automatic reset of that core.
nmi_handler	- optional interrupt function (ISR) . If there is no specific handler, this parameter must be NULL.
argument	- optional argument to be passed to the interrupt handler function.

Return values

OS_SUCCESS	- The watchdog was created successfully.
OS_ERR_WDT_INVALI↩ D_TIMEOUT	- The specified timeout value is not valid.

2.6.4.4.3 void osCoreWatchdogEnable ( )

Enables the watchdog.

Return values

OS_SUCCESS	- The watchdog was enabled successfully.
OS_ERR_WDT_NOT_CREATED	- The specified watchdog has not been created.

2.6.4.4.4 void osCoreWatchdogDisable ( )

Disables the watchdog.

2.6.4.4.5 void osCoreWatchdogService ( )

Services the watchdog.

2.6.4.4.6 uint64_t osCoreWatchdogCounterGet ( )

Retrieves the remaining time for the specified watchdog.

Returns

The remaining time for the specified watchdog.

2.6.5 OS Tick API

2.6.5.1 Overview

`OS Tick access functions.`

Modules

- OS Tick runtime API

2.6.5.2 OS Tick runtime API

2.6.5.2.1 Overview

OS Tick runtime API.

Functions

- void osTickCount (uint32_t *up_time)
- uint64_t osTickTime ()

2.6.5.2.2 Function Documentation

2.6.5.2.2.1 void osTickCount (uint32_t * *up_time*)

Retrieves the OS Tick count for the user.

Parameters

out	*up_time*	- Receives the OS Tick count.

2.6.5.2.2.2 uint64_t osTickTime ( )

Calculates the number of cycles since the Tick operation started.

Returns

64 bit time stamp

This function can be interrupted. This function is intended to be used for taking time stamps for logging purposes. Since it can be interrupted the returned time cannot be guaranteed for accuracy.

2.7 Utilities

2.7.1 Overview

Modules

- General System Utilities
- Debug Utilities

2.7.2 General System Utilities

2.7.2.1 Overview

Modules

- Data Types
- OS Error Coding
- Accessor API
- General System Functions
- System Information

2.7.2.2 Data Types

2.7.2.2.1 Overview

```
Data types used in SmartDSP OS
```

Macros

- #define HAVE_C99_DATATYPES
- #define INLINE static inline
- #define OS_BYTE_TYPE char
- #define OS_SHORT_TYPE short
- #define OS_LONG_TYPE long
- #define OS_LONG_LONG_TYPE long long
- #define OS_FLOAT_TYPE float
- #define OS_DOUBLE_TYPE double
- #define NULL ((void∗)0)

Boolean definitions

```
If not compiling with C++ support, this is used to define
boolean support
```

- typedef unsigned char **bool**
- #define FALSE (0)
- #define TRUE (1)

2.7.2.2.2 Macro Definition Documentation

2.7.2.2.2.1 #define HAVE_C99_DATATYPES

Line should be uncommented if external C99 header files are included.

2.7.2.2.2.2 #define INLINE static inline

static inline function macro

2.7.2.2.2.3 #define OS_BYTE_TYPE char

1 byte object

2.7.2.2.2.4 #define OS_SHORT_TYPE short

2 byte object

2.7.2.2.2.5 #define OS_LONG_TYPE long

4 byte object

2.7.2.2.2.6 #define OS_LONG_LONG_TYPE long long

8 byte object

2.7.2.2.2.7 #define OS_FLOAT_TYPE float

floating point object

2.7.2.2.2.8 #define OS_DOUBLE_TYPE double

double floating point object

2.7.2.2.2.9 #define FALSE (0)

Boolean false.

2.7.2.2.2.10 #define TRUE (1)

Boolean true.

2.7.2.2.2.11 #define NULL ((void∗)0)

Null pointer definition.

2.7.2.3 OS Error Coding

2.7.2.3.1 Overview

```
All error codes in SmartDSP OS
```

The error ([31:0]) is encoded according to the following scheme:

error[31:31] - set to 1 for architecture specific error.

error[30:30] - reserved, set to 0.

error[29:29] - set to 1 for system events indication.

error[28:28] - set to 1 for system information indication.

error[27:24] - architecture ID, valid only if error[31] is 1.

error[23:22] - reserved, set to 0.

error[21:16] - module ID.

error[15:0] - error ID.

Macros

- #define OS_ERR_ARCH_SPECIFIC 0x80000000
- #define OS_ERROR_BITS 0x00000000
- #define OS_EVENT_BITS 0x20000000
- #define OS_INFORMATION_BITS 0x10000000
- #define OS_ERROR(err, mod) ((err) | (mod) | OS_ERROR_BITS)
- #define OS_EVENT(err, mod) ((err) | (mod) | OS_EVENT_BITS)
- #define OS_INFO(err, mod) ((err) | (mod) | OS_INFORMATION_BITS)
- #define OS_ARCH_ERROR(err, mod, arch) ((err) | (mod) | (arch) | OS_ERR_ARCH_SPECIFIC)
- #define OS_FAIL 0x00000000
- #define OS_SUCCESS 0x00000001
- #define OS_ERR_UNKNOWN 0x00000002
- #define OS_ERR_NO_MEMORY 0x0000FFFF
- #define OS_ERR_ADDRESS_ERROR 0x0000FFF2
- #define OS_ERR_TRANSFER_ERROR 0x0000FFF1
- #define OS_ERR_INVALID_CONFIGURATION 0x0000FFF0
- #define OS_ERR_HANDLE_INVALID 0x0000FFEF
- #define OS_ERR_PARAM_INVALID 0x0000FFEE
- #define OS_ERR_MODE_INVALID 0x0000FFED
- #define OS_ERR_FUNCTION_INVALID 0x0000FFEC
- #define OS_ERR_COMMAND_INVALID 0x0000FFEB
- #define OS_ERR_COMMAND_UNSUPPORTED 0x0000FFEA
- #define OS_ERR_SPINLOCK_TAKEN 0x0000FFE9
- #define OS_ERR_UNAVAILABLE 0x0000FFDF
- #define OS_ERR_ALREADY_CREATED 0x0000FFDE
- #define OS_ERR_NOT_CREATED 0x0000FFDD
- #define OS_ERR_NOT_ACTIVE 0x0000FFDC
- #define OS_ERR_ALREADY_FREE 0x0000FFDB
- #define OS_ERR_PRIORITY_INVALID 0x0000FFDA
- #define OS_ERR_SOURCE_INVALID 0x0000FFD9
- #define OS_ERR_DEST_INVALID 0x0000FFD8
- #define OS_ERR_INTERVAL_UNSUPPORTED 0x0000FFD7
- #define OS_ERR_EMPTY 0x0000FFD6
- #define OS_ERR_FULL 0x0000FFD4
- #define OS_ERR_TOO_MUCH 0x0000FFD3
- #define OS_ERR_BUSY 0x0000FFD2
- #define OS_ERR_NO_ROOM 0x0000FFD1
- #define OS_ERR_TYPE_INVALID 0x0000FFD0
- #define OS_ERR_LENGTH_INVALID 0x0000FFCF
- #define OS_ERR_TOO_BIG 0x0000FFCE
- #define OS_ERR_TOO_SMALL 0x0000FFCD
- #define OS_ERR_VERSION_ERROR 0x0000FFCC
- #define OS_ERR_TIMEOUT 0x0000FFCB
- #define OS_ERR_PEND_IN_INTERRUPT 0x0000FFCA
- #define OS_ERR_OVERFLOW 0x0000FFC9
- #define OS_ERR_DELAY_IN_INTERRUPT 0x0000FFC8
- #define OS_ERR_YIELD_IN_INTERRUPT 0x0000FFC7

- #define OS_ERR_YIELD_TO_ILLEGAL_TASK 0x0000FFC6
- #define OS_ERR_YIELD_TO_SINGLE_TASK 0x0000FFC5
- #define OS_ERR_PEND_IN_LOCK 0x0000FFC4
- #define OS_ERR_ALREADY_ACTIVE 0x0000FFC3
- #define OS_ERR_DELETE_LOCKED 0x0000FFC2
- #define OS_ERR_DELAY_IN_LOCK 0x0000FFC1
- #define OS_ERR_UNDERRUN 0x0000FFC0
- #define OS_RETURN_NO_SUCCESS(status) if(status != OS_SUCCESS) RETURN_ERRO↩ R(status);
- #define OS_EVENT_BUSY OS_EVENT(OS_ERR_BUSY, OS_ERRMODULE_GENERAL)

Functions

- void osReportEvent (uint32_t event, const char *file, int line, const char *remarks)
- void REPORT_EVENT (uint32_t _err)
- void REPORT_EVENT_MESSAGE (uint32_t _err, const char *message)
- void REPORT_ERROR (uint32_t _err)
- void REPORT_ERROR_MESSAGE (uint32_t _err, const char *message)
- void RETURN_ERROR (uint32_t _err)
- void RETURN_ERROR_MESSAGE (uint32_t _err, const char *message)

Error architecture coding

- #define OS_ERRARCH_MSC815X 0x07000000
- #define OS_ERRARCH_PSC9X3X 0x08000000

Error module coding

- #define OS_ERRMODULE_GENERAL 0x00010000
- #define OS_ERRMODULE_HWI 0x00020000
- #define OS_ERRMODULE_SWI 0x00030000
- #define OS_ERRMODULE_TIMERS 0x00040000
- #define OS_ERRMODULE_HW_TIMERS 0x00050000
- #define OS_ERRMODULE_QUEUES 0x00060000
- #define OS_ERRMODULE_MESSAGES 0x00070000
- #define OS_ERRMODULE_DMA 0x00080000
- #define OS_ERRMODULE_NET 0x00090000
- #define OS_ERRMODULE_BIO 0x000A0000
- #define OS_ERRMODULE_CIO 0x000B0000
- #define OS_ERRMODULE_SIO 0x000C0000
- #define OS_ERRMODULE_WDT 0x000D0000
- #define OS_ERRMODULE_TSK 0x000E0000
- #define OS_ERRMODULE_EVENT 0x000F0000
- #define OS_ERRMODULE_EVENT_QUEUE 0x00100000
- #define OS_ERRMODULE_EVENT_SEMAPHORE 0x00110000
- #define OS_ERRMODULE_LLD 0x00120000
- #define OS_ERRMODULE_TICK 0x00130000
- #define OS_ERRMODULE_COP 0x00140000

Error Low Level Driver (LLD) coding

- #define OS_ERRMODULE_TDM 0x00200000
- #define OS_ERRMODULE_ETHERNET 0x00210000
- #define OS_ERRMODULE_UART 0x00220000
- #define OS_ERRMODULE_HDI 0x00230000
- #define OS_ERRMODULE_DSI 0x00240000
- #define OS_ERRMODULE_MMU 0x00250000
- #define OS_ERRMODULE_SRIO 0x00260000
- #define OS_ERRMODULE_I2C 0x00270000
- #define OS_ERRMODULE_SPI 0x00280000
- #define OS_ERRMODULE_CLOCK 0x00290000
- #define OS_ERRMODULE_CLASS 0x002A0000
- #define OS_ERRMODULE_PERF_MON 0x002B0000
- #define OS_ERRMODULE_L2_CACHE 0x002C0000
- #define OS_ERRMODULE_MAPLE 0x002D0000
- #define OS_ERRMODULE_VIRQ 0x002E0000
- #define OS_ERRMODULE_DEBUG_PRINT 0x002F0000
- #define OS_ERRMODULE_EMSG 0x00300000
- #define OS_ERRMODULE_QML 0x00310000
- #define OS_ERRMODULE_BML 0x00320000
- #define OS_ERRMODULE_HSSI 0x00330000
- #define OS_ERRMODULE_HDLC 0x00340000
- #define OS_ERRMODULE_AIC 0x00350000
- #define OS_ERRMODULE_CPRI 0x00360000
- #define OS_ERRMODULE_BMAN 0x00370000
- #define OS_ERRMODULE_QMAN 0x00380000
- #define OS_ERRMODULE_DTU 0x00390000
- #define OS_ERRMODULE_QDMA 0x003A0000
- #define OS_ERRMODULE_BOOT 0x003F0000

Task related error codes

- #define OS_ERR_TSK_BLOCKING OS_ERROR(OS_ERR_NOT_ACTIVE, OS_ERRMODUL↩ E_TSK)
- #define OS_ERR_TSK_ALREADY_SUSPENDED OS_ERROR(0x0000EFFE, OS_ERRMOD↩ ULE_TSK)
- #define OS_ERR_TSK_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERR↩ MODULE_TSK)
- #define OS_ERR_TSK_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_CREATED, OS_ERRMODULE_TSK)
- #define OS_ERR_TSK_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERR↩ MODULE_TSK)
- #define OS_ERR_TSK_DELAY_IN_INTERR OS_ERROR(OS_ERR_DELAY_IN_INTERRU↩ PT, OS_ERRMODULE_TSK)
- #define OS_ERR_TSK_YIELD_TO_ILLEGAL_TASK OS_ERROR(OS_ERR_YIELD_TO_IL↩ LEGAL_TASK, OS_ERRMODULE_TSK)
- #define OS_ERR_TSK_YIELD_TO_SINGLE_TASK OS_ERROR(OS_ERR_YIELD_TO_SIN↩ GLE_TASK, OS_ERRMODULE_TSK)
- #define OS_ERR_TSK_OS_ERR_YIELD_IN_INTERR OS_ERROR(OS_ERR_YIELD_IN_IN↩ TERRUPT, OS_ERRMODULE_TSK)
- #define OS_ERR_TSK_OS_ERR_ALREADY_ACTIVE OS_ERROR(OS_ERR_ALREADY_A↩

CTIVE, OS_ERRMODULE_TSK)
- #define OS_ERR_TSK_NOT_ACTIVE OS_ERROR(OS_ERR_NOT_ACTIVE, OS_ERRMOD↩ ULE_TSK)
- #define OS_ERR_TSK_DELAY_IN_LOCK OS_ERROR(OS_ERR_DELAY_IN_LOCK, OS_E↩ RRMODULE_TSK)
- #define OS_ERR_TSK_DELETE_LOCKED OS_ERROR(OS_ERR_DELETE_LOCKED, OS_↩ ERRMODULE_TSK)
- #define OS_ERR_TSK_MMU_SET_IN_INTERR OS_ERROR(0x0000EFFD, OS_ERRMODU↩ LE_TSK)
- #define OS_ERR_TSK_STACK_TOO_BIG OS_ERROR(OS_ERR_TOO_BIG, OS_ERRMOD↩ ULE_TSK)
- #define OS_ERR_TSK_STACK_UNALIGNED OS_ERROR(OS_ERR_LENGTH_INVALID, OS_ERRMODULE_TSK)

Event related error codes

- #define OS_ERR_EVENT_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ER↩ RMODULE_EVENT)
- #define OS_ERR_EVENT_PEND_IN_LOCK OS_ERROR(OS_ERR_PEND_IN_LOCK, OS_E↩ RRMODULE_EVENT)
- #define OS_ERR_EVENT_QUEUE_TIMEOUT OS_ERROR(OS_ERR_TIMEOUT, OS_ERRM↩ ODULE_EVENT_QUEUE)
- #define OS_ERR_EVENT_QUEUE_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_EVENT_QUEUE)
- #define OS_ERR_EVENT_QUEUE_PEND_INTERR OS_ERROR(OS_ERR_PEND_IN_INTE↩ RRUPT, OS_ERRMODULE_EVENT_QUEUE)
- #define OS_ERR_EVENT_QUEUE_OVERFLOW OS_ERROR(OS_ERR_OVERFLOW, OS_E↩ RRMODULE_EVENT_QUEUE)
- #define OS_ERR_EVENT_QUEUE_ALREADY_CREATED OS_ERROR(OS_ERR_ALREAD↩ Y_CREATED, OS_ERRMODULE_EVENT_QUEUE)
- #define OS_ERR_EVENT_QUEUE_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_EVENT_QUEUE)
- #define OS_ERR_EVENT_SEMA_TIMEOUT OS_ERROR(OS_ERR_TIMEOUT, OS_ERRM↩ ODULE_EVENT_SEMAPHORE)
- #define OS_ERR_EVENT_SEMA_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_EVENT_SEMAPHORE)
- #define OS_ERR_EVENT_SEMA_PEND_INTERR OS_ERROR(OS_ERR_PEND_IN_INTER↩ RUPT, OS_ERRMODULE_EVENT_SEMAPHORE)
- #define OS_ERR_EVENT_SEMA_OVERFLOW OS_ERROR(OS_ERR_OVERFLOW, OS_ER↩ RMODULE_EVENT_SEMAPHORE)
- #define OS_ERR_EVENT_SEMA_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY↩ _CREATED, OS_ERRMODULE_EVENT_QUEUE)
- #define OS_ERR_EVENT_SEMA_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS↩ _ERRMODULE_EVENT_QUEUE)

Hardware Interrupt related error codes

- #define OS_ERR_HWI_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_HWI)
- #define OS_ERR_HWI_MODE_INVALID OS_ERROR(OS_ERR_MODE_INVALID, OS_ERRMODULE_HWI)
- #define OS_ERR_HWI_FUNCTION_INVALID OS_ERROR(OS_ERR_FUNCTION_INVALID, OS_ERRMODULE_HWI)
- #define OS_ERR_HWI_PRIORITY_INVALID OS_ERROR(OS_ERR_PRIORITY_INVALID, OS_ERRMODULE_HWI)
- #define OS_ERR_HWI_COMMAND_INVALID OS_ERROR(OS_ERR_COMMAND_INVALID, OS_ERRMODULE_HWI)
- #define OS_ERR_HWI_DISPATCHER_INVALID OS_ERROR(OS_ERR_PARAM_INVALID, OS_ERRMODULE_HWI)
- #define OS_ERR_HWI_NO_ROOM OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_HWI)

Software interrupt related error codes

- #define OS_ERR_SWI_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_SWI)
- #define OS_ERR_SWI_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_SWI)
- #define OS_ERR_SWI_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_CREATED, OS_ERRMODULE_SWI)
- #define OS_ERR_SWI_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_SWI)
- #define OS_ERR_SWI_NOT_ACTIVE OS_ERROR(OS_ERR_NOT_ACTIVE, OS_ERRMODULE_SWI)
- #define OS_ERR_SWI_FUNCTION_INVALID OS_ERROR(OS_ERR_FUNCTION_INVALID, OS_ERRMODULE_SWI)
- #define OS_ERR_SWI_PRIORITY_INVALID OS_ERROR(OS_ERR_PRIORITY_INVALID, OS_ERRMODULE_SWI)

Software timer error codes

- #define OS_ERR_TM_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_TIMERS)
- #define OS_ERR_TM_MODE_INVALID OS_ERROR(OS_ERR_MODE_INVALID, OS_ERRMODULE_TIMERS)
- #define OS_ERR_TM_FUNCTION_INVALID OS_ERROR(OS_ERR_FUNCTION_INVALID, OS_ERRMODULE_TIMERS)
- #define OS_ERR_TM_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_TIMERS)
- #define OS_ERR_TM_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_CREATED, OS_ERRMODULE_TIMERS)
- #define OS_ERR_TM_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_TIMERS)

- #define OS_ERR_TM_NOT_ACTIVE OS_ERROR(OS_ERR_NOT_ACTIVE, OS_ERRMODULE_TIMERS)

Hardware timer related error codes

- #define OS_ERR_HW_TM_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_HW_TIMERS)
- #define OS_ERR_HW_TM_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_HW_TIMERS)
- #define OS_ERR_HW_TM_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_CREATED, OS_ERRMODULE_HW_TIMERS)
- #define OS_ERR_HW_TM_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_HW_TIMERS)
- #define OS_ERR_HW_TM_MODE_INVALID OS_ERROR(OS_ERR_MODE_INVALID, OS_ERRMODULE_HW_TIMERS)
- #define OS_ERR_HW_TM_PRIORITY_INVALID OS_ERROR(OS_ERR_PRIORITY_INVALID, OS_ERRMODULE_HW_TIMERS)
- #define OS_ERR_HW_TM_SOURCE_INVALID OS_ERROR(OS_ERR_SOURCE_INVALID, OS_ERRMODULE_HW_TIMERS)
- #define OS_ERR_HW_TM_INTERVAL_UNSUPPORTED OS_ERROR(OS_ERR_INTERVAL_UNSUPPORTED, OS_ERRMODULE_HW_TIMERS)
- #define OS_ERR_HW_TM_ALREADY_ACTIVE OS_ERROR(OS_ERR_ALREADY_ACTIVE, OS_ERRMODULE_HW_TIMERS)
- #define OS_ERR_HW_TM_NOT_ACTIVE OS_ERROR(OS_ERR_NOT_ACTIVE, OS_ERRMODULE_HW_TIMERS)
- #define OS_ERR_HW_TM_OS_ERR_BUSY OS_ERROR(OS_ERR_BUSY, OS_ERRMODULE_HW_TIMERS)

Queue related error codes

- #define OS_ERR_Q_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_QUEUES)
- #define OS_ERR_Q_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_CREATED, OS_ERRMODULE_QUEUES)
- #define OS_ERR_Q_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_QUEUES)
- #define OS_ERR_Q_EMPTY OS_ERROR(OS_ERR_EMPTY, OS_ERRMODULE_QUEUES)
- #define OS_ERR_Q_FULL OS_ERROR(OS_ERR_FULL , OS_ERRMODULE_QUEUES)

Messaging related error codes

- #define OS_ERR_MSG_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_MESSAGES)
- #define OS_ERR_MSG_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_MESSAGES)
- #define OS_ERR_MSG_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_CREATED, OS_ERRMODULE_MESSAGES)

- #define OS_ERR_MSG_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERR↩ MODULE_MESSAGES)
- #define OS_ERR_MSG_FUNCTION_INVALID OS_ERROR(OS_ERR_FUNCTION_INVALID, OS_ERRMODULE_MESSAGES)
- #define OS_ERR_MSG_NUM_LIMIT OS_ERROR(OS_ERR_TOO_MUCH, OS_ERRMODUL↩ E_MESSAGES)
- #define OS_ERR_MSG_DEST_INVALID OS_ERROR(OS_ERR_DEST_INVALID, OS_ERR↩ MODULE_MESSAGES)
- #define OS_ERR_MSG_SRC_INVALID OS_ERROR(OS_ERR_SOURCE_INVALID, OS_ER↩ RMODULE_MESSAGES)
- #define OS_ERR_MSG_BUSY OS_ERROR(OS_ERR_BUSY, OS_ERRMODULE_MESSAGE↩ S)
- #define OS_ERR_MSG_DEST_LOCKED OS_ERR_MSG_BUSY
- #define OS_ERR_MSG_NO_MSG_TO_GET OS_ERROR(OS_ERR_ALREADY_FREE, OS_E↩ RRMODULE_MESSAGES)

DMA related error codes

- #define OS_ERR_DMA_RESOURCE_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILAB↩ LE, OS_ERRMODULE_DMA)
- #define OS_ERR_DMA_RESOURCE_ALREADY_FREE OS_ERROR(OS_ERR_ALREADY_↩ FREE, OS_ERRMODULE_DMA)
- #define OS_ERR_DMA_NO_ROOM_FOR_CHAIN OS_ERROR(OS_ERR_NO_ROOM, OS_E↩ RRMODULE_DMA)
- #define OS_ERR_DMA_CHAIN_FULL OS_ERROR(OS_ERR_FULL, OS_ERRMODULE_D↩ MA)
- #define OS_ERR_DMA_CHAIN_TYPE_INVALID OS_ERROR(OS_ERR_TYPE_INVALID, OS_ERRMODULE_DMA)
- #define OS_ERR_DMA_CHAIN_LENGTH_INVALID OS_ERROR(OS_ERR_LENGTH_INV↩ ALID, OS_ERRMODULE_DMA)
- #define OS_ERR_DMA_CHAIN_BOUND OS_ERROR(0x0000EFFB, OS_ERRMODULE_D↩ MA)
- #define OS_ERR_DMA_CHANNEL_BOUND OS_ERROR(0x0000EFFA, OS_ERRMODULE↩ _DMA)
- #define OS_ERR_DMA_CHANNEL_NOT_BOUND OS_ERROR(0x0000EFF9, OS_ERRMOD↩ ULE_DMA)
- #define OS_ERR_DMA_CHANNEL_INDEX_INVALID OS_ERROR(0x0000EFF8, OS_ERR↩ MODULE_DMA)
- #define OS_ERR_DMA_PRIORITY_TAKEN OS_ERROR(0x0000EFF7, OS_ERRMODULE_↩ DMA)
- #define OS_ERR_DMA_DEST_NOT_ALIGNED OS_ERROR(0x0000EFF6, OS_ERRMODUL↩ E_DMA)
- #define OS_ERR_DMA_SOURCE_NOT_ALIGNED OS_ERROR(0x0000EFF5, OS_ERRMO↩ DULE_DMA)
- #define OS_ERR_DMA_SIZE_NOT_ALIGNED OS_ERROR(0x0000EFF4, OS_ERRMODUL↩ E_DMA)

Network stack related error codes

- #define OS_ERR_NET_CONSUMED OS_ERROR(0x0000EFFF, OS_ERRMODULE_NET)
- #define OS_ERR_NET_TOO_LARGE OS_ERROR(OS_ERR_TOO_BIG, OS_ERRMODULE_↩ NET)
- #define OS_ERR_NET_TOO_SMALL OS_ERROR(OS_ERR_TOO_SMALL, OS_ERRMODU↩ LE_NET)
- #define OS_ERR_SOCKET_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_E↩ RRMODULE_NET)
- #define OS_ERR_RTP_LENGTH_ERROR OS_ERROR(OS_ERR_LENGTH_INVALID, OS_E↩ RRMODULE_NET)
- #define OS_ERR_RTP_VERSION_ERROR OS_ERROR(OS_ERR_VERSION_ERROR, OS_E↩ RRMODULE_NET)
- #define OS_ERR_RTCP_VERSION_ERROR OS_ERROR(OS_ERR_VERSION_ERROR, OS_↩ ERRMODULE_NET)
- #define OS_ERR_RTCP_TYPE_INVALID OS_ERROR(OS_ERR_TYPE_INVALID, OS_ERR↩ MODULE_NET)
- #define OS_ERR_RTCP_LENGTH_ERROR OS_ERROR(OS_ERR_LENGTH_INVALID, OS_↩ ERRMODULE_NET)
- #define OS_ERR_NET_UNREACHABLE OS_ERROR(0x0000EFFC, OS_ERRMODULE_NET)
- #define OS_ERR_NET_SP_MISMATCH OS_ERROR(0x0000EFFB, OS_ERRMODULE_NET)
- #define OS_ERR_NO_BUFFERS OS_ERROR(OS_ERR_NO_MEMORY, OS_ERRMODULE_↩ GENERAL)
- #define OS_ERR_NO_FRAMES OS_ERROR(OS_ERR_NO_MEMORY, OS_ERRMODULE_↩ GENERAL)
- #define OS_ERR_ARP_TABLE_FULL OS_ERROR(OS_ERR_FULL, OS_ERRMODULE_NET)
- #define OS_ERR_NET_UNKNOWN_PROTOCOL OS_ERROR(0x0000EFFA, OS_ERRMOD↩ ULE_NET)
- #define OS_ERR_NET_ND_TABLE_FULL OS_ERROR(OS_ERR_FULL, OS_ERRMODULE↩ _NET)

MMU related error codes

- #define OS_ERR_MMU_VIRT_OVERLAPPED OS_ERROR(0x0000FFFF, OS_ERRMODULE↩ _MMU)
- #define OS_ERR_MMU_PHYS_OVERLAPPED OS_ERROR(0x0000FFFE, OS_ERRMODUL↩ E_MMU)
- #define OS_ERR_MMU_WRONG_ALIGNMENT OS_ERROR(0x0000FFFD, OS_ERRMODU↩ LE_MMU)
- #define OS_ERR_MMU_WRONG_SIZE OS_ERROR(0x0000FFFC, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_WRONG_DESC OS_ERROR(0x0000FFFB, OS_ERRMODULE_MM↩ U)
- #define OS_ERR_MMU_WRONG_BOUNDARY OS_ERROR(0x0000FFFA, OS_ERRMODU↩ LE_MMU)
- #define OS_ERR_MMU_MULTIPLE_HIT OS_ERROR(0x0000FF0F, OS_ERRMODULE_MM↩ U)
- #define OS_ERR_MMU_NON_MAPPED OS_ERROR(0x0000FF1F, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_PRIVILEGE OS_ERROR(0x0000FF2F, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_NOT_ALIGNED OS_ERROR(0x0000FF3F, OS_ERRMODULE_MM↩ U)

- #define OS_ERR_MMU_THESAME_ADDR OS_ERROR(0x0000FF4F, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_SEG_MISS OS_ERROR(0x0000FF5F, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_EDC OS_ERROR(0x0000FF6F, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DSPE_ERR OS_ERROR(0x0000FF7F, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_PSPE_ERR OS_ERROR(0x0000FF8F, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DPV_ERR OS_ERROR(0x0000FF9F, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DCCV_ERR OS_ERROR(0x0000FFAF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DAVW_ERR OS_ERROR(0x0000FFBF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DAVD_ERR OS_ERROR(0x0000FFCF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DECC_ERR OS_ERROR(0x0000FFDF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DRE_ERR OS_ERROR(0x0000FFEF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DNC_ERR OS_ERROR(0x0000F0FF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DNCC_ERR OS_ERROR(0x0000F1FF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DSOV_ERR OS_ERROR(0x0000F2FF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DSAS_ERR OS_ERROR(0x0000F3FF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_DPAS_ERR OS_ERROR(0x0000F4FF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_PECC_ERR OS_ERROR(0x0000F5FF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_PFE_ERR OS_ERROR(0x0000F6FF, OS_ERRMODULE_MMU)
- #define OS_ERR_MMU_PNC_ERR OS_ERROR(0x0000F7FF, OS_ERRMODULE_MMU)
- #define OS_ERROR_MEM_TYPE_ATTR_NOT_METCHING OS_ERROR(0x0000F8FF, OS_ERRMODULE_MMU)
- #define **OS_ERR_MMU_L1_CACHE_CANNOT_BE_DISABLED** OS_ERROR(0x0000F9FF, OS_ERRMODULE_MMU)
- #define **OS_ERR_MMU_STACK_IS_NOT_RESERVED** OS_ERROR(0x0000F9FF, OS_ERRMODULE_MMU)

BIO related error codes

- #define OS_EVENT_BIO_DEQUEUE OS_EVENT(OS_ERR_MODE_INVALID, OS_ERRMODULE_BIO)

COP related error codes

- #define OS_ERR_COP_QUEUE_SIZE_MISMATCH OS_ERROR(OS_ERR_LENGTH_INVALID, OS_ERRMODULE_COP)

L2 cache related error codes

- #define OS_ERR_L2_CACHE_INVALID_FUNCTION OS_ERROR(OS_ERR_FUNCTION_INVALID, OS_ERRMODULE_L2_CACHE)
- #define OS_ERR_L2_CACHE_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_L2_CACHE)
- #define OS_ERR_L2_CACHE_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_L2_CACHE)
- #define OS_ERR_L2_CACHE_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_CREATED, OS_ERRMODULE_L2_CACHE)
- #define OS_ERR_L2_CACHE_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_L2_CACHE)

- #define OS_ERR_L2_CACHE_WRONG_ALIGNMENT OS_ERROR(0x0000FFFF, OS_ERRM↩ ODULE_L2_CACHE)
- #define OS_ERR_L2_CACHE_WRONG_SIZE OS_ERROR(0x0000FFFE, OS_ERRMODULE↩ _L2_CACHE)
- #define OS_ERR_L2_CACHE_NOT_CACHEABLE OS_ERROR(0x0000FFFD, OS_ERRMOD↩ ULE_L2_CACHE)
- #define OS_ERR_L2_CACHE_PARTITION_FULL OS_ERROR(0x0000FFFC, OS_ERRMOD↩ ULE_L2_CACHE)
- #define OS_ERR_L2_CACHE_CME_LACK_OF_SPACE OS_ERROR(0x0000FFFB, OS_ERR↩ MODULE_L2_CACHE)
- #define OS_ERR_L2_CACHE_CME_DISABLE_IN_CONTROL_REGS OS_ERROR(0x0000F↩ FFA, OS_ERRMODULE_L2_CACHE)
- #define OS_ERR_L2_CACHE_CME_INVALID_COMMAND OS_ERROR(0x0000FFFA, OS_↩ ERRMODULE_L2_CACHE)
- #define OS_ERR_L2_CACHE_CME_IS_BUSY OS_ERROR(0x0000FFF9, OS_ERRMODULE↩ _L2_CACHE)
- #define OS_ERR_L2_CACHE_CME_INVALID_PARAMS OS_ERROR(0x0000FFF8, OS_ER↩ RMODULE_L2_CACHE)

Debug and Trace Unit related error codes

- #define OS_ERR_DTU_TRIAD_COUNTERS_ARE_ALREADY_ALLOCATED OS_ERRO↩ R(0x0000EFFF, OS_ERRMODULE_DTU)

I2C related error codes

- #define OS_ERR_I2C_INVALID_PARAM OS_ERROR(0x0000EFFF, OS_ERRMODULE_I2C)
- #define OS_ERR_I2C_LOST_ARBITRATION OS_ERROR(0x0000EFFE, OS_ERRMODULE↩ _I2C)
- #define OS_ERR_I2C_9TH_CLOCK_TIMEOUT OS_ERROR(0x0000EFFD, OS_ERRMODU↩ LE_I2C)
- #define OS_ERR_I2C_BUS_STUCK OS_ERROR(0x0000EFFC, OS_ERRMODULE_I2C)
- #define OS_ERR_I2C_UNAVAILABLE OS_ERROR(0x0000EFFB, OS_ERRMODULE_I2C)
- #define OS_ERR_I2C_BUS_BUSY OS_ERROR(0x0000EFFA, OS_ERRMODULE_I2C)

Qman related error codes

- #define OS_ERR_QMAN_INVALID_PARAM OS_ERROR(0x0000EFFF, OS_ERRMODULE_↩ QMAN)
- #define OS_ERR_QMAN_INVALID_STATE OS_ERROR(0x0000EFFE, OS_ERRMODULE_↩ QMAN)
- #define OS_ERR_QMAN_QUEUE_IS_EMPTY OS_ERROR(0x0000EFFD, OS_ERRMODUL↩ E_QMAN)
- #define OS_ERR_QMAN_PORTAL_IS_BUSY OS_ERROR(0x0000EFFC, OS_ERRMODULE↩ _QMAN)
- #define OS_ERR_QMAN_FQ_IS_BUSY OS_ERROR(0x0000EFFB, OS_ERRMODULE_QM↩ AN)

Watchdog related error codes

- #define OS_ERR_WDT_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_CREAT↩ ED, OS_ERRMODULE_WDT)
- #define OS_ERR_WDT_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERR↩ MODULE_WDT)
- #define OS_ERR_WDT_INVALID_TIMEOUT OS_ERROR(OS_ERR_PARAM_INVALID, O↩ S_ERRMODULE_WDT)
- #define OS_ERR_WDT_INVALID_MODE OS_ERROR(OS_ERR_MODE_INVALID, OS_ER↩ RMODULE_WDT)
- #define OS_ERR_WDT_DISABLED OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMOD↩ ULE_WDT)
- #define OS_ERR_WDT_TIMEOUT_TOO_SMALL OS_ERROR(0x0000EFFF, OS_ERRMOD↩ ULE_WDT)

SPI related error codes

- #define OS_ERR_SPI_INVALID_MODE OS_ERROR(OS_ERR_MODE_INVALID, OS_ERR↩ MODULE_SPI)
- #define OS_ERR_SPI_NO_MEMORY OS_ERROR(OS_ERR_NO_MEMORY, OS_ERRMOD↩ ULE_SPI)
- #define OS_ERR_SPI_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRM↩ ODULE_SPI)
- #define OS_ERR_SPI_INVALID_PARAM OS_ERROR(OS_ERR_PARAM_INVALID, OS_E↩ RRMODULE_SPI)
- #define OS_ERR_SPI_TX_BD_BUSY OS_ERROR(0x0000EFFF, OS_ERRMODULE_SPI)
- #define OS_ERR_SPI_TX_ERROR OS_ERROR(0x0000EFFE, OS_ERRMODULE_SPI)
- #define OS_ERR_SPI_RX_BUSY OS_ERROR(OS_ERR_BUSY, OS_ERRMODULE_SPI)
- #define OS_ERR_SPI_MULTI_MASTER_ERROR OS_ERROR(0x0000EFFD, OS_ERRMOD↩ ULE_SPI)

Ethernet related error codes

- #define OS_ERR_ETHERNET_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_UNAVA↩ ILABLE, OS_ERRMODULE_ETHERNET)

HDLC related error codes

- #define OS_ERR_HDLC_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILAB↩ LE, OS_ERRMODULE_HDLC)

CLASS related error codes

- #define OS_ERR_CLASS_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ER↩ RMODULE_CLASS)
- #define OS_ERR_CLASS_INVALID_CLASS OS_ERROR(OS_ERR_DEST_INVALID, OS_E↩ RRMODULE_CLASS)

- #define OS_ERR_CLASS_BAD_CONFIG OS_ERROR(0x0000EFFF, OS_ERRMODULE_CL↩ ASS)

Performance Monitor related error codes

- #define OS_ERR_PERF_MON_BAD_CONFIG OS_ERROR(0x0000EFFF, OS_ERRMODULE↩ _PERF_MON)

MAPLE related error codes

- #define OS_ERR_MAPLE_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILA↩ BLE, OS_ERRMODULE_MAPLE)
- #define OS_ERR_MAPLE_NO_ROOM OS_ERROR(OS_ERR_FULL, OS_ERRMODULE_M↩ APLE)
- #define OS_ERR_MAPLE_NO_MEM OS_ERROR(OS_ERR_NO_MEMORY, OS_ERRMOD↩ ULE_MAPLE)
- #define OS_ERR_MAPLE_TOO_MANY_BD OS_ERROR(OS_ERR_TOO_MUCH, OS_ERR↩ MODULE_MAPLE)

LLD related error codes

- #define OS_SUCESS_LLD_TX OS_ERROR(0x0000EFFF, OS_ERRMODULE_LLD)
- #define OS_EVENT_LLD_EVENT OS_EVENT(0x00000FFF, OS_ERRMODULE_LLD)

Virtual interrupts related error codes

- #define OS_ERR_VIRQ_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERR↩ MODULE_VIRQ)
- #define OS_ERR_VIRQ_ALREADY_FREE OS_ERROR(OS_ERR_ALREADY_FREE, OS_E↩ RRMODULE_VIRQ)

Debug Print related error codes

- #define OS_ERR_DEBUG_PRINT_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_DEBUG_PRINT);
- #define OS_ERR_DEBUG_PRINT_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_DEBUG_PRINT);
- #define OS_ERR_DEBUG_PRINT_NOT_ACTIVE OS_ERROR(OS_ERR_NOT_ACTIVE, OS↩ _ERRMODULE_DEBUG_PRINT);
- #define OS_ERR_DEBUG_PRINT_LENGTH_INVALID OS_ERROR(OS_ERR_LENGTH_IN↩ VALID, OS_ERRMODULE_DEBUG_PRINT)
- #define OS_ERR_DEBUG_PRINT_MDP_ACTIVE OS_ERROR(OS_ERR_BUSY, OS_ERRM↩ ODULE_DEBUG_PRINT)
- #define OS_ERR_DEBUG_PRINT_OVERFLOW OS_ERROR(OS_ERR_OVERFLOW, OS_E↩ RRMODULE_DEBUG_PRINT)

RapidIO related error codes

- #define OS_ERR_RIO_NO_SYNC OS_ERROR(0x0000EFFF, OS_ERRMODULE_SRIO)
- #define OS_ERR_RIO_PORT_STOPPED OS_ERROR(0x0000EFFE, OS_ERRMODULE_SRIO)
- #define OS_ERR_RIO_LINK_RESPONSE_TIME_OUT OS_ERROR(0x0000EFFD, OS_ERRMODULE_SRIO)
- #define OS_ERR_RIO_LINK_UNRECOVERABLE_ERROR OS_ERROR(0x0000EFFC, OS_ERRMODULE_SRIO)

eMSG related error codes

- #define OS_ERR_EMSG_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_EMSG)

QML related error codes

- #define OS_ERR_QML_QUEUE_FULL OS_ERROR(OS_ERR_FULL, OS_ERRMODULE_QML)
- #define OS_ERR_QML_QUEUE_EMPTY OS_ERROR(OS_ERR_EMPTY, OS_ERRMODULE_QML)
- #define OS_ERR_QML_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_QML)

BML related error codes

- #define OS_ERR_BML_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_BML)
- #define OS_ERR_BML_RING_FULL OS_ERROR(OS_ERR_FULL, OS_ERRMODULE_BML)
- #define OS_ERR_BML_RING_EMPTY OS_ERROR(OS_ERR_EMPTY, OS_ERRMODULE_BML)

HSSI related error codes

- #define OS_ERR_HSSI_NO_SYNC OS_ERROR(0x0000EFFF, OS_ERRMODULE_HSSI)

2.7.2.3.2 Macro Definition Documentation

2.7.2.3.2.1 #define OS_ERR_ARCH_SPECIFIC 0x80000000

Indicated architecture specific error.

2.7.2.3.2.2 #define OS_ERROR_BITS 0x00000000

Indicates system error.

2.7.2.3.2.3 #define OS_EVENT_BITS 0x20000000

Indicates system event.

2.7.2.3.2.4 #define OS_INFORMATION_BITS 0x10000000

Indicates system information.

2.7.2.3.2.5 #define OS_ERROR(*err, mod*) ((err) | (mod) | OS_ERROR_BITS)

Constructing macro for building a module specific error code.

2.7.2.3.2.6 #define OS_EVENT(*err, mod*) ((err) | (mod) | OS_EVENT_BITS)

Constructing macro for building a module specific event code.

2.7.2.3.2.7 #define OS_INFO(*err, mod*) ((err) | (mod) | OS_INFORMATION_BITS)

Constructing macro for building a module specific information code.

2.7.2.3.2.8 #define OS_ARCH_ERROR(*err, mod, arch*) ((err) | (mod) | (arch) | OS_ERR_ARCH_SPECIFIC)

Constructing macro for building a architecture and module specific error code.

2.7.2.3.2.9 #define OS_ERRARCH_MSC815X 0x07000000

Architecture is MSC815X.

2.7.2.3.2.10 #define OS_ERRARCH_PSC9X3X 0x08000000

Architecture is PSC9X3X.

2.7.2.3.2.11 #define OS_ERRMODULE_GENERAL 0x00010000

General software module.

2.7.2.3.2.12 #define OS_ERRMODULE_HWI 0x00020000

Hardware interrupt software module.

2.7.2.3.2.13 #define OS_ERRMODULE_SWI 0x00030000

Software interrupt software module.

2.7.2.3.2.14 #define OS_ERRMODULE_TIMERS 0x00040000

Software timers software module.

2.7.2.3.2.15 #define OS_ERRMODULE_HW_TIMERS 0x00050000

Hardware timers software module.

2.7.2.3.2.16 #define OS_ERRMODULE_QUEUES 0x00060000

Queues software module.

2.7.2.3.2.17 #define OS_ERRMODULE_MESSAGES 0x00070000

Messaging software module.

2.7.2.3.2.18 #define OS_ERRMODULE_DMA 0x00080000

DMA software module.

2.7.2.3.2.19 #define OS_ERRMODULE_NET 0x00090000

Network stacks software module.

2.7.2.3.2.20 #define OS_ERRMODULE_BIO 0x000A0000

Buffered I/O (BIO) software module.

2.7.2.3.2.21 #define OS_ERRMODULE_CIO 0x000B0000

Character I/O (CIO) software module.

2.7.2.3.2.22 #define OS_ERRMODULE_SIO 0x000C0000

Synchronized I/O (SIO) software module.

2.7.2.3.2.23 #define OS_ERRMODULE_WDT 0x000D0000

Watchdog timers software module.

2.7.2.3.2.24 #define OS_ERRMODULE_TSK 0x000E0000

Tasks software module.

2.7.2.3.2.25 #define OS_ERRMODULE_EVENT 0x000F0000

Events software module.

2.7.2.3.2.26 #define OS_ERRMODULE_EVENT_QUEUE 0x00100000

Events queue software module.

2.7.2.3.2.27 #define OS_ERRMODULE_EVENT_SEMAPHORE 0x00110000

Events semaphore software module.

2.7.2.3.2.28 #define OS_ERRMODULE_LLD 0x00120000

Low Level Driver (LLD) software module.

2.7.2.3.2.29 #define OS_ERRMODULE_TICK 0x00130000

System tick software module.

2.7.2.3.2.30 #define OS_ERRMODULE_COP 0x00140000

Co-Processor (COP) software module.

2.7.2.3.2.31 #define OS_ERRMODULE_TDM 0x00200000

Time Division Multiplexing (TDM) LLD.

2.7.2.3.2.32 #define OS_ERRMODULE_ETHERNET 0x00210000

Ethernet LLD.

2.7.2.3.2.33 #define OS_ERRMODULE_UART 0x00220000

UART LLD.

2.7.2.3.2.34 #define OS_ERRMODULE_HDI 0x00230000

Host Interface (HDI) LLD.

2.7.2.3.2.35 #define OS_ERRMODULE_DSI 0x00240000

Direct Slave Interface (DSI) LLD.

2.7.2.3.2.36 #define OS_ERRMODULE_MMU 0x00250000

Memory Management Unit (MMU) LLD.

2.7.2.3.2.37 #define OS_ERRMODULE_SRIO 0x00260000

Serial RapidIO LLD.

2.7.2.3.2.38 #define OS_ERRMODULE_I2C 0x00270000

I2C LLD.

2.7.2.3.2.39 #define OS_ERRMODULE_SPI 0x00280000

Serial Peripheral Interface (SPI) LLD.

2.7.2.3.2.40 #define OS_ERRMODULE_CLOCK 0x00290000

Clocks LLD.

2.7.2.3.2.41 #define OS_ERRMODULE_CLASS 0x002A0000

Chip-Level Arbitration and Switching System LLD.

2.7.2.3.2.42 #define OS_ERRMODULE_PERF_MON 0x002B0000

Performance Monitor LLD.

2.7.2.3.2.43 #define OS_ERRMODULE_L2_CACHE 0x002C0000

L2 cache LLD.

2.7.2.3.2.44 #define OS_ERRMODULE_MAPLE 0x002D0000

Multi Accelerator Platform Engine (MAPLE) LLD.

2.7.2.3.2.45 #define OS_ERRMODULE_VIRQ 0x002E0000

Virtual Interrupts LLD.

2.7.2.3.2.46 #define OS_ERRMODULE_DEBUG_PRINT 0x002F0000

Debug Print LLD.

2.7.2.3.2.47 #define OS_ERRMODULE_EMSG 0x00300000

Enhanced Message Unit (eMSG) LLD.

2.7.2.3.2.48 #define OS_ERRMODULE_QML 0x00310000

QMan-Lite (QML) LLD.

2.7.2.3.2.49 #define OS_ERRMODULE_BML 0x00320000

BMan-Lite (BML) LLD.

2.7.2.3.2.50 #define OS_ERRMODULE_HSSI 0x00330000

High Speed Serial nterface (HSSI) LLD.

2.7.2.3.2.51 #define OS_ERRMODULE_HDLC 0x00340000

High-Level Data Link Control LLD.

2.7.2.3.2.52 #define OS_ERRMODULE_AIC 0x00350000

Antenna Interface Controller.

2.7.2.3.2.53 #define OS_ERRMODULE_CPRI 0x00360000

Antenna Interface Controller.

2.7.2.3.2.54 #define OS_ERRMODULE_BMAN 0x00370000

Buffer's manager.

2.7.2.3.2.55 #define OS_ERRMODULE_QMAN 0x00380000

Queue's manager.

2.7.2.3.2.56 #define OS_ERRMODULE_DTU 0x00390000

Debug and Trace Unit.

2.7.2.3.2.57 #define OS_ERRMODULE_QDMA 0x003A0000

QDMA Unit.

2.7.2.3.2.58 #define OS_ERRMODULE_BOOT 0x003F0000

Reserved coding space for boot ROM.

2.7.2.3.2.59 #define OS_FAIL 0x00000000

General failure.

2.7.2.3.2.60 #define OS_SUCCESS 0x00000001

General success.

2.7.2.3.2.61 #define OS_ERR_UNKNOWN 0x00000002

Unknown error.

2.7.2.3.2.62 #define OS_ERR_NO_MEMORY 0x0000FFFF

No memory available.

2.7.2.3.2.63 #define OS_ERR_ADDRESS_ERROR 0x0000FFF2

Passed address is invalid.

2.7.2.3.2.64 #define OS_ERR_TRANSFER_ERROR 0x0000FFF1

Passed configuration is invalid.

2.7.2.3.2.65 #define OS_ERR_INVALID_CONFIGURATION 0x0000FFF0

Passed configuration is invalid.

2.7.2.3.2.66 #define OS_ERR_HANDLE_INVALID 0x0000FFEF

Passed handle is invalid.

2.7.2.3.2.67 #define OS_ERR_PARAM_INVALID 0x0000FFEE

Passed parameter is invalid.

2.7.2.3.2.68 #define OS_ERR_MODE_INVALID 0x0000FFED

Passed mode is invalid.

2.7.2.3.2.69 #define OS_ERR_FUNCTION_INVALID 0x0000FFEC

Passed function is invalid.

2.7.2.3.2.70 #define OS_ERR_COMMAND_INVALID 0x0000FFEB

Passed command is invalid.

2.7.2.3.2.71 #define OS_ERR_COMMAND_UNSUPPORTED 0x0000FFEA

Passed command is unsupported.

2.7.2.3.2.72 #define OS_ERR_SPINLOCK_TAKEN 0x0000FFE9

The command could not obtain a spinlock, and failed.

2.7.2.3.2.73 #define OS_ERR_UNAVAILABLE 0x0000FFDF

Resource is unavailable.

2.7.2.3.2.74 #define OS_ERR_ALREADY_CREATED 0x0000FFDE

Resource was already created.

2.7.2.3.2.75 #define OS_ERR_NOT_CREATED 0x0000FFDD

Resource was not created.

2.7.2.3.2.76 #define OS_ERR_NOT_ACTIVE 0x0000FFDC

Resource was created, but not activated.

2.7.2.3.2.77 #define OS_ERR_ALREADY_FREE 0x0000FFDB

resource was already frees

2.7.2.3.2.78 #define OS_ERR_PRIORITY_INVALID 0x0000FFDA

Passed priority is invalid.

2.7.2.3.2.79 #define OS_ERR_SOURCE_INVALID 0x0000FFD9

Passed source is invalid.

2.7.2.3.2.80 #define OS_ERR_DEST_INVALID 0x0000FFD8

Passed destination is invalid.

2.7.2.3.2.81 #define OS_ERR_INTERVAL_UNSUPPORTED 0x0000FFD7

Passed interval is unsupported.

2.7.2.3.2.82 #define OS_ERR_EMPTY 0x0000FFD6

Resource is empty.

2.7.2.3.2.83 #define OS_ERR_FULL 0x0000FFD4

Resource is full.

2.7.2.3.2.84 #define OS_ERR_TOO_MUCH 0x0000FFD3

Attempt to process too much data.

2.7.2.3.2.85 #define OS_ERR_BUSY 0x0000FFD2

Resource is currently busy.

2.7.2.3.2.86 #define OS_ERR_NO_ROOM 0x0000FFD1

No room to store all of the data.

2.7.2.3.2.87 #define OS_ERR_TYPE_INVALID 0x0000FFD0

Indicated type of resource is invalid.

2.7.2.3.2.88 #define OS_ERR_LENGTH_INVALID 0x0000FFCF

Passed length is invalid.

2.7.2.3.2.89 #define OS_ERR_TOO_BIG 0x0000FFCE

Data is too large.

2.7.2.3.2.90 #define OS_ERR_TOO_SMALL 0x0000FFCD

Data is too small.

2.7.2.3.2.91 #define OS_ERR_VERSION_ERROR 0x0000FFCC

Version is not compatible.

2.7.2.3.2.92 #define OS_ERR_TIMEOUT 0x0000FFCB

Timeout.

2.7.2.3.2.93 #define OS_ERR_PEND_IN_INTERRUPT 0x0000FFCA

Attempt to pend while in interrupt context.

2.7.2.3.2.94 #define OS_ERR_OVERFLOW 0x0000FFC9

Overflow error.

2.7.2.3.2.95 #define OS_ERR_DELAY_IN_INTERRUPT 0x0000FFC8

Attempt to delay while in interrupt context.

2.7.2.3.2.96 #define OS_ERR_YIELD_IN_INTERRUPT 0x0000FFC7

Attempt to yield while in interrupt context.

2.7.2.3.2.97 #define OS_ERR_YIELD_TO_ILLEGAL_TASK 0x0000FFC6

Attempt to yield to an illegal task.

2.7.2.3.2.98 #define OS_ERR_YIELD_TO_SINGLE_TASK 0x0000FFC5

Attempt to yield when there are no other tasks.

2.7.2.3.2.99 #define OS_ERR_PEND_IN_LOCK 0x0000FFC4

Attempt to pend when scheduler locked.

2.7.2.3.2.100 #define OS_ERR_ALREADY_ACTIVE 0x0000FFC3

Attempt to activate an already active resource.

2.7.2.3.2.101 #define OS_ERR_DELETE_LOCKED 0x0000FFC2

Attempt to delete a locked resource.

2.7.2.3.2.102 #define OS_ERR_DELAY_IN_LOCK 0x0000FFC1

Attempt to delay from within a locked resource.

2.7.2.3.2.103 #define OS_ERR_UNDERRUN 0x0000FFC0

Underrun error.

2.7.2.3.2.104 #define OS_RETURN_NO_SUCCESS(*status*) if(status != OS_SUCCESS) RETURN_ERROR(status);

Check and return on failure.

2.7.2.3.2.105 #define OS_ERR_TSK_BLOCKING OS_ERROR(OS_ERR_NOT_ACTIVE, OS_ERRMODULE_TSK)

Task is blocking.

2.7.2.3.2.106 #define OS_ERR_TSK_ALREADY_SUSPENDED OS_ERROR(0x0000EFFE, OS_ERRMODULE_TSK)

Attempt to suspend an already suspended task.

2.7.2.3.2.107 #define OS_ERR_TSK_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_TSK)

Task not yet created.

2.7.2.3.2.108 #define OS_ERR_TSK_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_↩ CREATED, OS_ERRMODULE_TSK)

Attempt to create an already created task.

2.7.2.3.2.109 #define OS_ERR_TSK_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_TSK)

Task is unavailable.

2.7.2.3.2.110 #define OS_ERR_TSK_DELAY_IN_INTERR OS_ERROR(OS_ERR_DELAY_IN_I↩ NTERRUPT, OS_ERRMODULE_TSK)

Attempt to delay a task while in interrupt context.

2.7.2.3.2.111 #define OS_ERR_TSK_YIELD_TO_ILLEGAL_TASK OS_ERROR(OS_ERR_YIELD↩ _TO_ILLEGAL_TASK, OS_ERRMODULE_TSK)

Attempt to yield to an illegal task.

2.7.2.3.2.112 #define OS_ERR_TSK_YIELD_TO_SINGLE_TASK OS_ERROR(OS_ERR_YIELD_↩ TO_SINGLE_TASK, OS_ERRMODULE_TSK)

Attempt to yield when there is a single task in the system.

2.7.2.3.2.113 #define OS_ERR_TSK_OS_ERR_YIELD_IN_INTERR OS_ERROR(OS_ERR_YIEL↩ D_IN_INTERRUPT, OS_ERRMODULE_TSK)

Attempt to yield a task while in interrupt context.

2.7.2.3.2.114 #define OS_ERR_TSK_OS_ERR_ALREADY_ACTIVE OS_ERROR(OS_ERR_ALR↩ EADY_ACTIVE, OS_ERRMODULE_TSK)

Attempt activate an already active task.

2.7.2.3.2.115 #define OS_ERR_TSK_NOT_ACTIVE OS_ERROR(OS_ERR_NOT_ACTIVE, OS_ERRMODULE_TSK)

Attempt to query an inactive task.

2.7.2.3.2.116 #define OS_ERR_TSK_DELAY_IN_LOCK OS_ERROR(OS_ERR_DELAY_IN_LO↩ CK, OS_ERRMODULE_TSK)

Attempt to delay a task while the scheduler is locked.

2.7.2.3.2.117 #define OS_ERR_TSK_DELETE_LOCKED OS_ERROR(OS_ERR_DELETE_LOC↩ KED, OS_ERRMODULE_TSK)

Attempt to delete a locked task.

2.7.2.3.2.118 #define OS_ERR_TSK_MMU_SET_IN_INTERR OS_ERROR(0x0000EFFD, OS_ERRMODULE_TSK)

Attempt to set a task MMU context while in interrupt context.

2.7.2.3.2.119 #define OS_ERR_TSK_STACK_TOO_BIG OS_ERROR(OS_ERR_TOO_BIG, OS_ERRMODULE_TSK)

Attempt to create a task with a stack too large to be handled by the OS.

2.7.2.3.2.120 #define OS_ERR_TSK_STACK_UNALIGNED OS_ERROR(OS_ERR_LENGTH_IN↩ VALID, OS_ERRMODULE_TSK)

Attempt to create a task with a stack that is unaligned to software/hardware requirements.

2.7.2.3.2.121 #define OS_ERR_EVENT_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_EVENT)

No event is available.

2.7.2.3.2.122 #define OS_ERR_EVENT_PEND_IN_LOCK OS_ERROR(OS_ERR_PEND_IN_LO↩ CK, OS_ERRMODULE_EVENT)

Attempt to pend on an event while the scheduler is locked.

2.7.2.3.2.123 #define OS_ERR_EVENT_QUEUE_TIMEOUT OS_ERROR(OS_ERR_TIMEOUT, OS_ERRMODULE_EVENT_QUEUE)

Timeout while pending on an event queue.

2.7.2.3.2.124 #define OS_ERR_EVENT_QUEUE_UNAVAILABLE OS_ERROR(OS_ERR_UNAV↩ AILABLE, OS_ERRMODULE_EVENT_QUEUE)

No event queue is available.

2.7.2.3.2.125 #define OS_ERR_EVENT_QUEUE_PEND_INTERR OS_ERROR(OS_ERR_PEND_IN_INTERRUPT, OS_ERRMODULE_EVENT_QUEUE)

Attempt to pend on an event queue while in interrupt context.

2.7.2.3.2.126 #define OS_ERR_EVENT_QUEUE_OVERFLOW OS_ERROR(OS_ERR_OVERFLOW, OS_ERRMODULE_EVENT_QUEUE)

Overflow on an event queue.

2.7.2.3.2.127 #define OS_ERR_EVENT_QUEUE_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_CREATED, OS_ERRMODULE_EVENT_QUEUE)

Attempt to create an already created event queue.

2.7.2.3.2.128 #define OS_ERR_EVENT_QUEUE_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_EVENT_QUEUE)

Attempt to perform an operation on an invalid event queue.

2.7.2.3.2.129 #define OS_ERR_EVENT_SEMA_TIMEOUT OS_ERROR(OS_ERR_TIMEOUT, OS_ERRMODULE_EVENT_SEMAPHORE)

Timeout while pending on an event semaphore.

2.7.2.3.2.130 #define OS_ERR_EVENT_SEMA_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_EVENT_SEMAPHORE)

No event semaphore is available.

2.7.2.3.2.131 #define OS_ERR_EVENT_SEMA_PEND_INTERR OS_ERROR(OS_ERR_PEND_IN_INTERRUPT, OS_ERRMODULE_EVENT_SEMAPHORE)

Attempt to pend on an event semaphore while in interrupt context.

2.7.2.3.2.132 #define OS_ERR_EVENT_SEMA_OVERFLOW OS_ERROR(OS_ERR_OVERFLOW, OS_ERRMODULE_EVENT_SEMAPHORE)

Overflow on an event semaphore.

2.7.2.3.2.133 #define OS_ERR_EVENT_SEMA_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_CREATED, OS_ERRMODULE_EVENT_QUEUE)

Attempt to create an already created event semaphore.

2.7.2.3.2.134 #define OS_ERR_EVENT_SEMA_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_EVENT_QUEUE)

Attempt to perform an operation on an invalid event semaphore.

2.7.2.3.2.135 #define OS_ERR_HWI_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_HWI)

Hardware interrupt handle is invalid.

2.7.2.3.2.136 #define OS_ERR_HWI_MODE_INVALID OS_ERROR(OS_ERR_MODE_INVALID, OS_ERRMODULE_HWI)

Hardware interrupt mode is invalid.

2.7.2.3.2.137 #define OS_ERR_HWI_FUNCTION_INVALID OS_ERROR(OS_ERR_FUNCTION_I↩ NVALID, OS_ERRMODULE_HWI)

Hardware interrupt function (ISR) is invalid.

2.7.2.3.2.138 #define OS_ERR_HWI_PRIORITY_INVALID OS_ERROR(OS_ERR_PRIORITY_IN↩ VALID, OS_ERRMODULE_HWI)

Hardware interrupt priority is invalid.

2.7.2.3.2.139 #define OS_ERR_HWI_COMMAND_INVALID OS_ERROR(OS_ERR_COMMAND_↩ INVALID, OS_ERRMODULE_HWI)

Hardware interrupt command is invalid.

2.7.2.3.2.140 #define OS_ERR_HWI_DISPATCHER_INVALID OS_ERROR(OS_ERR_PARAM_I↩ NVALID, OS_ERRMODULE_HWI)

Hardware interrupt dispatcher is invalid.

2.7.2.3.2.141 #define OS_ERR_HWI_NO_ROOM OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_HWI)

No room to install a new dispatcher.

2.7.2.3.2.142 #define OS_ERR_SWI_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_SWI)

Software interrupt handle is invalid.

2.7.2.3.2.143 #define OS_ERR_SWI_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_SWI)

No software interrupt is available.

2.7.2.3.2.144 #define OS_ERR_SWI_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_↩ CREATED, OS_ERRMODULE_SWI)

Attempt to create an already created software interrupt.

2.7.2.3.2.145 #define OS_ERR_SWI_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_SWI)

Software interrupt handle is not created.

2.7.2.3.2.146 #define OS_ERR_SWI_NOT_ACTIVE OS_ERROR(OS_ERR_NOT_ACTIVE, OS_ERRMODULE_SWI)

Attempt to query an inactive software interrupt.

2.7.2.3.2.147 #define OS_ERR_SWI_FUNCTION_INVALID OS_ERROR(OS_ERR_FUNCTION_I↩ NVALID, OS_ERRMODULE_SWI)

Software interrupt function (ISR) is invalid.

2.7.2.3.2.148 #define OS_ERR_SWI_PRIORITY_INVALID OS_ERROR(OS_ERR_PRIORITY_IN↩ VALID, OS_ERRMODULE_SWI)

Software interrupt priority is invalid.

2.7.2.3.2.149 #define OS_ERR_TM_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_TIMERS)

Software timer handle is invalid.

2.7.2.3.2.150 #define OS_ERR_TM_MODE_INVALID OS_ERROR(OS_ERR_MODE_INVALID, OS_ERRMODULE_TIMERS)

Software timer mode is invalid.

2.7.2.3.2.151 #define OS_ERR_TM_FUNCTION_INVALID OS_ERROR(OS_ERR_FUNCTION_I↩ NVALID, OS_ERRMODULE_TIMERS)

Software timer function is invalid.

2.7.2.3.2.152 #define OS_ERR_TM_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_TIMERS)

No software timer is available.

2.7.2.3.2.153 #define OS_ERR_TM_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_C↩ REATED, OS_ERRMODULE_TIMERS)

Attempt to create an already created software timer.

2.7.2.3.2.154 #define OS_ERR_TM_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_TIMERS)

Software timer handle is not created.

2.7.2.3.2.155 #define OS_ERR_TM_NOT_ACTIVE OS_ERROR(OS_ERR_NOT_ACTIVE, OS_ERRMODULE_TIMERS)

Attempt to query an inactive software timer.

2.7.2.3.2.156 #define OS_ERR_HW_TM_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_HW_TIMERS)

Hardware timer handle is invalid.

2.7.2.3.2.157 #define OS_ERR_HW_TM_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_HW_TIMERS)

No hardware timer is available.

2.7.2.3.2.158 #define OS_ERR_HW_TM_ALREADY_CREATED OS_ERROR(OS_ERR_ALREAD↩ Y_CREATED, OS_ERRMODULE_HW_TIMERS)

Attempt to create an already created hardware timer.

2.7.2.3.2.159 #define OS_ERR_HW_TM_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREAT↩ ED, OS_ERRMODULE_HW_TIMERS)

Hardware timer handle is not created.

2.7.2.3.2.160 #define OS_ERR_HW_TM_MODE_INVALID OS_ERROR(OS_ERR_MODE_INVA↩ LID, OS_ERRMODULE_HW_TIMERS)

Hardware timer mode is invalid.

2.7.2.3.2.161 #define OS_ERR_HW_TM_PRIORITY_INVALID OS_ERROR(OS_ERR_PRIORITY↩ _INVALID, OS_ERRMODULE_HW_TIMERS)

Hardware interrupt priority is invalid.

2.7.2.3.2.162 #define OS_ERR_HW_TM_SOURCE_INVALID OS_ERROR(OS_ERR_SOURCE_I↩ NVALID, OS_ERRMODULE_HW_TIMERS)

Clock source for hardware timer is invalid.

2.7.2.3.2.163 #define OS_ERR_HW_TM_INTERVAL_UNSUPPORTED OS_ERROR(OS_ERR_IN↩ TERVAL_UNSUPPORTED, OS_ERRMODULE_HW_TIMERS)

Hardware timer interval is unsupported (out of range)

2.7.2.3.2.164 #define OS_ERR_HW_TM_ALREADY_ACTIVE OS_ERROR(OS_ERR_ALREADY↩ _ACTIVE, OS_ERRMODULE_HW_TIMERS)

hardware timer is already active

2.7.2.3.2.165 #define OS_ERR_HW_TM_NOT_ACTIVE OS_ERROR(OS_ERR_NOT_ACTIVE, OS_ERRMODULE_HW_TIMERS)

hardware timer was created but not activated

2.7.2.3.2.166 #define OS_ERR_HW_TM_OS_ERR_BUSY OS_ERROR(OS_ERR_BUSY, OS_ERRMODULE_HW_TIMERS)

hardware timer is running and can’t be modified

2.7.2.3.2.167 #define OS_ERR_Q_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_QUEUES)

Queue handle is invalid.

2.7.2.3.2.168 #define OS_ERR_Q_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_C↩ REATED, OS_ERRMODULE_QUEUES)

Attempt to create an already created queue.

2.7.2.3.2.169 #define OS_ERR_Q_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_QUEUES)

Queue handle is not created.

2.7.2.3.2.170 #define OS_ERR_Q_EMPTY OS_ERROR(OS_ERR_EMPTY, OS_ERRMODULE_QUEUES)

Queue is empty.

2.7.2.3.2.171 #define OS_ERR_Q_FULL OS_ERROR(OS_ERR_FULL , OS_ERRMODULE_QUEUES)

Queue is full.

2.7.2.3.2.172 #define OS_ERR_MSG_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_MESSAGES)

Message handle is invalid.

2.7.2.3.2.173 #define OS_ERR_MSG_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_MESSAGES)

No message handle is available.

2.7.2.3.2.174 #define OS_ERR_MSG_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_↩ CREATED, OS_ERRMODULE_MESSAGES)

Attempt to create an already created message handle.

2.7.2.3.2.175 #define OS_ERR_MSG_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_MESSAGES)

Message handle is not created.

2.7.2.3.2.176 #define OS_ERR_MSG_FUNCTION_INVALID OS_ERROR(OS_ERR_FUNCTION_↩ INVALID, OS_ERRMODULE_MESSAGES)

Message function is invalid.

2.7.2.3.2.177 #define OS_ERR_MSG_NUM_LIMIT OS_ERROR(OS_ERR_TOO_MUCH, OS_ERRMODULE_MESSAGES)

Too many message handles.

2.7.2.3.2.178 #define OS_ERR_MSG_DEST_INVALID OS_ERROR(OS_ERR_DEST_INVALID, OS_ERRMODULE_MESSAGES)

Message destination is invalid.

2.7.2.3.2.179 #define OS_ERR_MSG_SRC_INVALID OS_ERROR(OS_ERR_SOURCE_INVALID, OS_ERRMODULE_MESSAGES)

Message source is invalid.

2.7.2.3.2.180 #define OS_ERR_MSG_BUSY OS_ERROR(OS_ERR_BUSY, OS_ERRMODULE_MESSAGES)

Message mailbox/queue is busy.

2.7.2.3.2.181 #define OS_ERR_MSG_DEST_LOCKED OS_ERR_MSG_BUSY

Message is locked by the destination.

2.7.2.3.2.182 #define OS_ERR_MSG_NO_MSG_TO_GET OS_ERROR(OS_ERR_ALREADY_F↩ REE, OS_ERRMODULE_MESSAGES)

No message available.

2.7.2.3.2.183 #define OS_ERR_DMA_RESOURCE_UNAVAILABLE OS_ERROR(OS_ERR_UNA↩ VAILABLE, OS_ERRMODULE_DMA)

DMA resource is unavailable.

2.7.2.3.2.184 #define OS_ERR_DMA_RESOURCE_ALREADY_FREE OS_ERROR(OS_ERR_AL↩ READY_FREE, OS_ERRMODULE_DMA)

Attempt to free an already freed DMA resource.

2.7.2.3.2.185 #define OS_ERR_DMA_NO_ROOM_FOR_CHAIN OS_ERROR(OS_ERR_NO_ROO↩ M, OS_ERRMODULE_DMA)

No room available to create DMA chain (BD ring)

2.7.2.3.2.186 #define OS_ERR_DMA_CHAIN_FULL OS_ERROR(OS_ERR_FULL, OS_ERRMODULE_DMA)

No room in DMA chain (BD ring) for additional transfers/buffers.

2.7.2.3.2.187 #define OS_ERR_DMA_CHAIN_TYPE_INVALID OS_ERROR(OS_ERR_TYPE_IN↩ VALID, OS_ERRMODULE_DMA)

DMA chain (BD ring) type is invalid.

2.7.2.3.2.188 #define OS_ERR_DMA_CHAIN_LENGTH_INVALID OS_ERROR(OS_ERR_LENGT↩ H_INVALID, OS_ERRMODULE_DMA)

DMA chain (BD ring) length is invalid.

2.7.2.3.2.189 #define OS_ERR_DMA_CHAIN_BOUND OS_ERROR(0x0000EFFB, OS_ERRMODULE_DMA)

DMA chain (BD ring) is currently bound to a channel.

2.7.2.3.2.190 #define OS_ERR_DMA_CHANNEL_BOUND OS_ERROR(0x0000EFFA, OS_ERRMODULE_DMA)

DMA channel is currently bound to a chain (BD ring)

2.7.2.3.2.191 #define OS_ERR_DMA_CHANNEL_NOT_BOUND OS_ERROR(0x0000EFF9, OS_ERRMODULE_DMA)

DMA channel is currently not bound to a chain (BD ring)

2.7.2.3.2.192 #define OS_ERR_DMA_CHANNEL_INDEX_INVALID OS_ERROR(0x0000EFF8, OS_ERRMODULE_DMA)

DMA channel number is invalid.

2.7.2.3.2.193 #define OS_ERR_DMA_PRIORITY_TAKEN OS_ERROR(0x0000EFF7, OS_ERRMODULE_DMA)

DMA channel priority is already taken.

2.7.2.3.2.194 #define OS_ERR_DMA_DEST_NOT_ALIGNED OS_ERROR(0x0000EFF6, OS_ERRMODULE_DMA)

DMA destination address is not aligned.

2.7.2.3.2.195 #define OS_ERR_DMA_SOURCE_NOT_ALIGNED OS_ERROR(0x0000EFF5, OS_ERRMODULE_DMA)

DMA source address is not aligned.

2.7.2.3.2.196 #define OS_ERR_DMA_SIZE_NOT_ALIGNED OS_ERROR(0x0000EFF4, OS_ERRMODULE_DMA)

DMA transfer size is not aligned.

2.7.2.3.2.197 #define OS_ERR_NET_CONSUMED OS_ERROR(0x0000EFFF, OS_ERRMODULE_NET)

Network packet was already consumed.

2.7.2.3.2.198 #define OS_ERR_NET_TOO_LARGE OS_ERROR(OS_ERR_TOO_BIG, OS_ERRMODULE_NET)

Network packet is too large.

2.7.2.3.2.199 #define OS_ERR_NET_TOO_SMALL OS_ERROR(OS_ERR_TOO_SMALL, OS_ERRMODULE_NET)

Network packet is too small.

2.7.2.3.2.200 #define OS_ERR_SOCKET_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILAB↩ LE, OS_ERRMODULE_NET)

Socket is unavailable.

2.7.2.3.2.201 #define OS_ERR_RTP_LENGTH_ERROR OS_ERROR(OS_ERR_LENGTH_INVA↩ LID, OS_ERRMODULE_NET)

RTP length is erroneous.

2.7.2.3.2.202 #define OS_ERR_RTP_VERSION_ERROR OS_ERROR(OS_ERR_VERSION_ER↩ ROR, OS_ERRMODULE_NET)

RTP version is erroneous.

2.7.2.3.2.203 #define OS_ERR_RTCP_VERSION_ERROR OS_ERROR(OS_ERR_VERSION_ER↩ ROR, OS_ERRMODULE_NET)

RTCP length is erroneous.

2.7.2.3.2.204 #define OS_ERR_RTCP_TYPE_INVALID OS_ERROR(OS_ERR_TYPE_INVALID, OS_ERRMODULE_NET)

RTCP type is invalid.

2.7.2.3.2.205 #define OS_ERR_RTCP_LENGTH_ERROR OS_ERROR(OS_ERR_LENGTH_INV↩ ALID, OS_ERRMODULE_NET)

RTCP length is erroneous.

2.7.2.3.2.206 #define OS_ERR_NET_UNREACHABLE OS_ERROR(0x0000EFFC, OS_ERRMODULE_NET)

Network destination is unreachable.

2.7.2.3.2.207 #define OS_ERR_NET_SP_MISMATCH OS_ERROR(0x0000EFFB, OS_ERRMODULE_NET)

IPSec security policy mismatch.

2.7.2.3.2.208 #define OS_ERR_NO_BUFFERS OS_ERROR(OS_ERR_NO_MEMORY, OS_ERRMODULE_GENERAL)

No available buffers in buffer pool.

2.7.2.3.2.209 #define OS_ERR_NO_FRAMES OS_ERROR(OS_ERR_NO_MEMORY, OS_ERRMODULE_GENERAL)

No available frames in frame pool.

2.7.2.3.2.210 #define OS_ERR_ARP_TABLE_FULL OS_ERROR(OS_ERR_FULL, OS_ERRMODULE_NET)

ARP table is full - can’t add another ARP entry.

2.7.2.3.2.211 #define OS_ERR_NET_UNKNOWN_PROTOCOL OS_ERROR(0x0000EFFA, OS_ERRMODULE_NET)

Network protocol is unknown.

2.7.2.3.2.212 #define OS_ERR_NET_ND_TABLE_FULL OS_ERROR(OS_ERR_FULL, OS_ERRMODULE_NET)

Neighbor discovery table is full - can’t add another ND entry.

2.7.2.3.2.213 #define OS_ERR_MMU_VIRT_OVERLAPPED OS_ERROR(0x0000FFFF, OS_ERRMODULE_MMU)

Overlap of virtual addresses between MMU descriptors.

2.7.2.3.2.214 #define OS_ERR_MMU_PHYS_OVERLAPPED OS_ERROR(0x0000FFFE, OS_ERRMODULE_MMU)

Overlap of physical addresses between MMU descriptors.

2.7.2.3.2.215 #define OS_ERR_MMU_WRONG_ALIGNMENT OS_ERROR(0x0000FFFD, OS_ERRMODULE_MMU)

MMU addresses are not aligned.

2.7.2.3.2.216 #define OS_ERR_MMU_WRONG_SIZE OS_ERROR(0x0000FFFC, OS_ERRMODULE_MMU)

MMU descriptor size is not legal.

2.7.2.3.2.217 #define OS_ERR_MMU_WRONG_DESC OS_ERROR(0x0000FFFB, OS_ERRMODULE_MMU)

MMU descriptor is not legal.

2.7.2.3.2.218 #define OS_ERR_MMU_WRONG_BOUNDARY OS_ERROR(0x0000FFFA, OS_ERRMODULE_MMU)

MMU descriptor is not within boundary restrictions.

2.7.2.3.2.219 #define OS_ERR_MMU_MULTIPLE_HIT OS_ERROR(0x0000FF0F, OS_ERRMODULE_MMU)

MMU hit on multiple descriptors.

2.7.2.3.2.220 #define OS_ERR_MMU_NON_MAPPED OS_ERROR(0x0000FF1F, OS_ERRMODULE_MMU)

Access to address that is not mapped by the hardware.

2.7.2.3.2.221 #define OS_ERR_MMU_PRIVILEGE OS_ERROR(0x0000FF2F, OS_ERRMODULE_MMU)

Access with wrong privilege level.

2.7.2.3.2.222 #define OS_ERR_MMU_NOT_ALIGNED OS_ERROR(0x0000FF3F, OS_ERRMODULE_MMU)

Access size not aligned to address.

2.7.2.3.2.223 #define OS_ERR_MMU_THESAME_ADDR OS_ERROR(0x0000FF4F, OS_ERRMODULE_MMU)

Accessing the same address on more than one bus.

2.7.2.3.2.224 #define OS_ERR_MMU_SEG_MISS OS_ERROR(0x0000FF5F, OS_ERRMODULE_MMU)

Access to address that is not mapped in any MMU descriptor.

2.7.2.3.2.225 #define OS_ERR_MMU_EDC OS_ERROR(0x0000FF6F, OS_ERRMODULE_MMU)

Access caused an EDC error.

2.7.2.3.2.226 #define OS_ERR_MMU_DSPE_ERR OS_ERROR(0x0000FF7F, OS_ERRMODULE_MMU)

Data MMU programming error.

2.7.2.3.2.227 #define OS_ERR_MMU_PSPE_ERR OS_ERROR(0x0000FF8F, OS_ERRMODULE_MMU)

Program MMU programming error.

2.7.2.3.2.228 #define OS_ERR_MMU_DPV_ERR OS_ERROR(0x0000FF9F, OS_ERRMODULE_MMU)

Data Permission Violation.

2.7.2.3.2.229 #define OS_ERR_MMU_DCCV_ERR OS_ERROR(0x0000FFAF, OS_ERRMODULE_MMU)

Data Cache Command Violation.

2.7.2.3.2.230 #define OS_ERR_MMU_DAVW_ERR OS_ERROR(0x0000FFBF, OS_ERRMODULE_MMU)

Data Access Violation Width.

2.7.2.3.2.231 #define OS_ERR_MMU_DAVD_ERR OS_ERROR(0x0000FFCF, OS_ERRMODULE_MMU)

Data Access Violation Direction.

2.7.2.3.2.232 #define OS_ERR_MMU_DECC_ERR OS_ERROR(0x0000FFDF, OS_ERRMODULE_MMU)

Data ECC Error.

2.7.2.3.2.233 #define OS_ERR_MMU_DRE_ERR OS_ERROR(0x0000FFEF, OS_ERRMODULE_MMU)

Data Read Error.

2.7.2.3.2.234 #define OS_ERR_MMU_DNC_ERR OS_ERROR(0x0000F0FF, OS_ERRMODULE_MMU)

NC hit.

2.7.2.3.2.235 #define OS_ERR_MMU_DNCC_ERR OS_ERROR(0x0000F1FF, OS_ERRMODULE_MMU)

Noncacheable Cache Command.

2.7.2.3.2.236 #define OS_ERR_MMU_DSOV_ERR OS_ERROR(0x0000F2FF, OS_ERRMODULE_MMU)

Stack Overrun Violation.

2.7.2.3.2.237 #define OS_ERR_MMU_DSAS_ERR OS_ERROR(0x0000F3FF, OS_ERRMODULE_MMU)

Semaphore Access Size Error.

2.7.2.3.2.238 #define OS_ERR_MMU_DPAS_ERR OS_ERROR(0x0000F4FF, OS_ERRMODULE_MMU)

Peripheral Access Size Error.

2.7.2.3.2.239 #define OS_ERR_MMU_PECC_ERR OS_ERROR(0x0000F5FF, OS_ERRMODULE_MMU)

Program ECC.

2.7.2.3.2.240 #define OS_ERR_MMU_PFE_ERR OS_ERROR(0x0000F6FF, OS_ERRMODULE_MMU)

Program Fetch Error.

2.7.2.3.2.241 #define OS_ERR_MMU_PNC_ERR OS_ERROR(0x0000F7FF, OS_ERRMODULE_MMU)

NC hit.

2.7.2.3.2.242 #define OS_ERROR_MEM_TYPE_ATTR_NOT_METCHING OS_ERROR(0x0000F8FF, OS_ERRMODULE_MMU)

subsection MMU attributes doesn’t match heap type

2.7.2.3.2.243 #define OS_EVENT_BIO_DEQUEUE OS_EVENT(OS_ERR_MODE_INVALID, OS_ERRMODULE_BIO)

Error in dequeueing from BIO queue (frame or buffer)

2.7.2.3.2.244 #define OS_ERR_COP_QUEUE_SIZE_MISMATCH OS_ERROR(OS_ERR_LENGT↩ H_INVALID, OS_ERRMODULE_COP)

Queue size not large enough for COP.

2.7.2.3.2.245 #define OS_ERR_L2_CACHE_INVALID_FUNCTION OS_ERROR(OS_ERR_FUNCTION_INVALID, OS_ERRMODULE_L2_CACHE)

L2 cache function is invalid.

2.7.2.3.2.246 #define OS_ERR_L2_CACHE_INVALID OS_ERROR(OS_ERR_HANDLE_INVALID, OS_ERRMODULE_L2_CACHE)

L2 cache resource (partition/prefetch-channel) is invalid.

2.7.2.3.2.247 #define OS_ERR_L2_CACHE_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_L2_CACHE)

L2 cache resource (partition/prefetch-channel) is unavailable.

2.7.2.3.2.248 #define OS_ERR_L2_CACHE_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_CREATED, OS_ERRMODULE_L2_CACHE)

L2 cache resource (partition/prefetch-channel) is already created.

2.7.2.3.2.249 #define OS_ERR_L2_CACHE_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_L2_CACHE)

L2 cache resource (partition/prefetch-channel) not created yet.

2.7.2.3.2.250 #define OS_ERR_L2_CACHE_WRONG_ALIGNMENT OS_ERROR(0x0000FFFF, OS_ERRMODULE_L2_CACHE)

L2 cache alignment error.

2.7.2.3.2.251 #define OS_ERR_L2_CACHE_WRONG_SIZE OS_ERROR(0x0000FFFE, OS_ERRMODULE_L2_CACHE)

L2 cache size error.

2.7.2.3.2.252 #define OS_ERR_L2_CACHE_NOT_CACHEABLE OS_ERROR(0x0000FFFD, OS_ERRMODULE_L2_CACHE)

Attempt to perform cache operation on a non-cacheable memory region.

2.7.2.3.2.253 #define OS_ERR_L2_CACHE_PARTITION_FULL OS_ERROR(0x0000FFFC, OS_ERRMODULE_L2_CACHE)

Attempt to perform cache operation on a non-cacheable memory region.

2.7.2.3.2.254 #define OS_ERR_L2_CACHE_CME_LACK_OF_SPACE OS_ERROR(0x0000FFFB, OS_ERRMODULE_L2_CACHE)

Maintenance instruction failed insertion into CME channel due to lack of space.

2.7.2.3.2.255 #define OS_ERR_L2_CACHE_CME_DISABLE_IN_CONTROL_REGS OS_ERRO↩ R(0x0000FFFA, OS_ERRMODULE_L2_CACHE)

Maintenance instruction failed insertion into CME channel because CME is disabled by control registers.

2.7.2.3.2.256 #define OS_ERR_L2_CACHE_CME_INVALID_COMMAND OS_ERROR(0x0000FFFA, OS_ERRMODULE_L2_CACHE)

Maintenance instruction failed insertion into CME channel because it is invalid.

2.7.2.3.2.257 #define OS_ERR_L2_CACHE_CME_IS_BUSY OS_ERROR(0x0000FFF9, OS_ERRMODULE_L2_CACHE)

Maintenance instruction failed because CME is busy allocating channel to previous job.

2.7.2.3.2.258 #define OS_ERR_L2_CACHE_CME_INVALID_PARAMS OS_ERROR(0x0000FFF8, OS_ERRMODULE_L2_CACHE)

Maintenance instruction failed because CME is busy allocating channel to previous job.

2.7.2.3.2.259 #define OS_ERR_DTU_TRIAD_COUNTERS_ARE_ALREADY_ALLOCATED OS_ER↩ ROR(0x0000EFFF, OS_ERRMODULE_DTU)

DTU triad counters are already allocated.

2.7.2.3.2.260 #define OS_ERR_I2C_INVALID_PARAM OS_ERROR(0x0000EFFF, OS_ERRMODULE_I2C)

I2C parameter is invalid.

2.7.2.3.2.261 #define OS_ERR_I2C_LOST_ARBITRATION OS_ERROR(0x0000EFFE, OS_ERRMODULE_I2C)

Lost arbitration on I2C bus.

2.7.2.3.2.262 #define OS_ERR_I2C_9TH_CLOCK_TIMEOUT OS_ERROR(0x0000EFFD, OS_ERRMODULE_I2C)

Timed out waiting for acknowledge on 9th clock cycle.

2.7.2.3.2.263 #define OS_ERR_I2C_BUS_STUCK OS_ERROR(0x0000EFFC, OS_ERRMODULE_I2C)

I2C bus is stuck.

2.7.2.3.2.264 #define OS_ERR_I2C_UNAVAILABLE OS_ERROR(0x0000EFFB, OS_ERRMODULE_I2C)

I2C module is unavailable.

2.7.2.3.2.265 #define OS_ERR_I2C_BUS_BUSY OS_ERROR(0x0000EFFA, OS_ERRMODULE_I2C)

I2C bus is busy.

2.7.2.3.2.266 #define OS_ERR_QMAN_INVALID_PARAM OS_ERROR(0x0000EFFF, OS_ERRMODULE_QMAN)

I2C parameter is invalid.

2.7.2.3.2.267 #define OS_ERR_QMAN_INVALID_STATE OS_ERROR(0x0000EFFE, OS_ERRMODULE_QMAN)

The Frame queue is in invalid state.

2.7.2.3.2.268 #define OS_ERR_QMAN_QUEUE_IS_EMPTY OS_ERROR(0x0000EFFD, OS_ERRMODULE_QMAN)

The requested queue is empty cannot dequeue.

2.7.2.3.2.269 #define OS_ERR_QMAN_PORTAL_IS_BUSY OS_ERROR(0x0000EFFC, OS_ERRMODULE_QMAN)

The Qman portal is currently busy.

2.7.2.3.2.270 #define OS_ERR_QMAN_FQ_IS_BUSY OS_ERROR(0x0000EFFB, OS_ERRMODULE_QMAN)

The Qman portal is currently busy.

2.7.2.3.2.271 #define OS_ERR_WDT_ALREADY_CREATED OS_ERROR(OS_ERR_ALREADY_↩ CREATED, OS_ERRMODULE_WDT)

Watchdog timer is already created.

2.7.2.3.2.272 #define OS_ERR_WDT_NOT_CREATED OS_ERROR(OS_ERR_NOT_CREATED, OS_ERRMODULE_WDT)

Watchdog timer is not created.

2.7.2.3.2.273 #define OS_ERR_WDT_INVALID_TIMEOUT OS_ERROR(OS_ERR_PARAM_INV↩ ALID, OS_ERRMODULE_WDT)

Watchdog timer timeout is invalid.

2.7.2.3.2.274 #define OS_ERR_WDT_INVALID_MODE OS_ERROR(OS_ERR_MODE_INVALID, OS_ERRMODULE_WDT)

Watchdog timer mode is invalid.

2.7.2.3.2.275 #define OS_ERR_WDT_DISABLED OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_WDT)

Watchdog timer is disabled.

2.7.2.3.2.276 #define OS_ERR_WDT_TIMEOUT_TOO_SMALL OS_ERROR(0x0000EFFF, OS_ERRMODULE_WDT)

Watchdog timer timeout is too short.

2.7.2.3.2.277 #define OS_ERR_SPI_INVALID_MODE OS_ERROR(OS_ERR_MODE_INVALID, OS_ERRMODULE_SPI)

SPI mode is invalid.

2.7.2.3.2.278 #define OS_ERR_SPI_NO_MEMORY OS_ERROR(OS_ERR_NO_MEMORY, OS_ERRMODULE_SPI)

No memory for SPI.

2.7.2.3.2.279 #define OS_ERR_SPI_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_SPI)

SPI module is unavailable.

2.7.2.3.2.280 #define OS_ERR_SPI_INVALID_PARAM OS_ERROR(OS_ERR_PARAM_INVALI↩ D, OS_ERRMODULE_SPI)

SPI parameter is invalid.

2.7.2.3.2.281 #define OS_ERR_SPI_TX_BD_BUSY OS_ERROR(0x0000EFFF, OS_ERRMODULE_SPI)

SPI transmit BD is busy.

2.7.2.3.2.282 #define OS_ERR_SPI_TX_ERROR OS_ERROR(0x0000EFFE, OS_ERRMODULE_SPI)

SPI transmit error.

2.7.2.3.2.283 #define OS_ERR_SPI_RX_BUSY OS_ERROR(OS_ERR_BUSY, OS_ERRMODULE_SPI)

SPI receive BD is busy.

2.7.2.3.2.284 #define OS_ERR_SPI_MULTI_MASTER_ERROR OS_ERROR(0x0000EFFD, OS_ERRMODULE_SPI)

SPI multiple master error.

2.7.2.3.2.285 #define OS_ERR_ETHERNET_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_↩ UNAVAILABLE, OS_ERRMODULE_ETHERNET)

Ethernet channel is unavailable.

2.7.2.3.2.286 #define OS_ERR_HDLC_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_UNAV↩ AILABLE, OS_ERRMODULE_HDLC)

HDLC channel is unavailable.

2.7.2.3.2.287 #define OS_ERR_CLASS_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_CLASS)

CLASS module is unavailable.

2.7.2.3.2.288 #define OS_ERR_CLASS_INVALID_CLASS OS_ERROR(OS_ERR_DEST_INVAL↩ ID, OS_ERRMODULE_CLASS)

CLASS module is invalid.

2.7.2.3.2.289 #define OS_ERR_CLASS_BAD_CONFIG OS_ERROR(0x0000EFFF, OS_ERRMODULE_CLASS)

Attempt to configure erroneous CLASS configuration.

2.7.2.3.2.290 #define OS_ERR_PERF_MON_BAD_CONFIG OS_ERROR(0x0000EFFF, OS_ERRMODULE_PERF_MON)

Attempt to configure erroneous performance monitor configuration.

2.7.2.3.2.291 #define OS_ERR_MAPLE_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_UNA↩ VAILABLE, OS_ERRMODULE_MAPLE)

MAPLE channel is unavailable.

2.7.2.3.2.292 #define OS_ERR_MAPLE_NO_ROOM OS_ERROR(OS_ERR_FULL, OS_ERRMODULE_MAPLE)

No room on MAPLE BD ring for all requested BD.

2.7.2.3.2.293 #define OS_ERR_MAPLE_NO_MEM OS_ERROR(OS_ERR_NO_MEMORY, OS_ERRMODULE_MAPLE)

No memory available in MAPLE Parameter RAM.

2.7.2.3.2.294 #define OS_ERR_MAPLE_TOO_MANY_BD OS_ERROR(OS_ERR_TOO_MUCH, OS_ERRMODULE_MAPLE)

Attempt to send more BD that available on BD ring.

2.7.2.3.2.295 #define OS_SUCESS_LLD_TX OS_ERROR(0x0000EFFF, OS_ERRMODULE_LLD)

LLD successfully transmitted the data - used internally.

2.7.2.3.2.296 #define OS_EVENT_LLD_EVENT OS_EVENT(0x00000FFF, OS_ERRMODULE_LLD)

LLD general event.

2.7.2.3.2.297 #define OS_ERR_VIRQ_UNAVAILABLE OS_ERROR(OS_ERR_UNAVAILABLE, OS_ERRMODULE_VIRQ)

No virtual interrupt available.

2.7.2.3.2.298 #define OS_ERR_VIRQ_ALREADY_FREE OS_ERROR(OS_ERR_ALREADY_FR↩ EE, OS_ERRMODULE_VIRQ)

Attempt to free an already freed virtual interrupt resource.

2.7.2.3.2.299 #define OS_ERR_DEBUG_PRINT_UNAVAILABLE OS_ERROR(OS_ERR_UNAVA↩ ILABLE, OS_ERRMODULE_DEBUG_PRINT);

Debug print module unavailable.

2.7.2.3.2.300 #define OS_ERR_DEBUG_PRINT_NOT_CREATED OS_ERROR(OS_ERR_NOT_C↩ REATED, OS_ERRMODULE_DEBUG_PRINT);

Debug print resource not created.

2.7.2.3.2.301 #define OS_ERR_DEBUG_PRINT_NOT_ACTIVE OS_ERROR(OS_ERR_NOT_AC↩ TIVE, OS_ERRMODULE_DEBUG_PRINT);

Debug print resource not active.

2.7.2.3.2.302 #define OS_ERR_DEBUG_PRINT_LENGTH_INVALID OS_ERROR(OS_ERR_LEN↩ GTH_INVALID, OS_ERRMODULE_DEBUG_PRINT)

Debug print length is erroneous.

2.7.2.3.2.303 #define OS_ERR_DEBUG_PRINT_MDP_ACTIVE OS_ERROR(OS_ERR_BUSY, OS_ERRMODULE_DEBUG_PRINT)

Attempt to perform debug print activity while mass debug print (mdp) is in progress.

2.7.2.3.2.304 #define OS_ERR_DEBUG_PRINT_OVERFLOW OS_ERROR(OS_ERR_OVERFLO↩ W, OS_ERRMODULE_DEBUG_PRINT)

Debug print overflow.

2.7.2.3.2.305 #define OS_ERR_RIO_NO_SYNC OS_ERROR(0x0000EFFF, OS_ERRMODULE_SRIO)

No synchronization on RapidIO bus.

2.7.2.3.2.306 #define OS_ERR_RIO_PORT_STOPPED OS_ERROR(0x0000EFFE, OS_ERRMODULE_SRIO)

RapidIO port is stopped.

2.7.2.3.2.307 #define OS_ERR_RIO_LINK_RESPONSE_TIME_OUT OS_ERROR(0x0000EFFD, OS_ERRMODULE_SRIO)

RapidIO link timed out.

2.7.2.3.2.308 #define OS_ERR_RIO_LINK_UNRECOVERABLE_ERROR OS_ERROR(0x0000EFFC, OS_ERRMODULE_SRIO)

Unrecoverable error detected by RapidIO controller.

2.7.2.3.2.309 #define OS_ERR_EMSG_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_UNA↩ VAILABLE, OS_ERRMODULE_EMSG)

eMSG channel is unavailable

2.7.2.3.2.310 #define OS_ERR_QML_QUEUE_FULL OS_ERROR(OS_ERR_FULL, OS_ERRMODULE_QML)

QMan-Lite queue is full.

2.7.2.3.2.311 #define OS_ERR_QML_QUEUE_EMPTY OS_ERROR(OS_ERR_EMPTY, OS_ERRMODULE_QML)

QMan-Lite queue is empty.

2.7.2.3.2.312 #define OS_ERR_QML_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_UNAV↩ AILABLE, OS_ERRMODULE_QML)

QMan-Lite channel is unavailable.

2.7.2.3.2.313 #define OS_ERR_BML_CHANNEL_UNAVAILABLE OS_ERROR(OS_ERR_UNAV↩ AILABLE, OS_ERRMODULE_BML)

BMan-Lite channel is unavailable.

2.7.2.3.2.314 #define OS_ERR_BML_RING_FULL OS_ERROR(OS_ERR_FULL, OS_ERRMODULE_BML)

BMan-Lite ring is full.

2.7.2.3.2.315 #define OS_ERR_BML_RING_EMPTY OS_ERROR(OS_ERR_EMPTY, OS_ERRMODULE_BML)

BMan-Lite ring is empty.

2.7.2.3.2.316 #define OS_ERR_HSSI_NO_SYNC OS_ERROR(0x0000EFFF, OS_ERRMODULE_HSSI)

No synchronization on HSSI.

2.7.2.3.2.317 #define OS_EVENT_BUSY OS_EVENT(OS_ERR_BUSY, OS_ERRMODULE_GE↩ NERAL)

General busy event.

2.7.2.3.3 Function Documentation

2.7.2.3.3.1 void osReportEvent (uint32_t *event,* const char * *file,* int *line,* const char * *remarks*)

Reports a system event

Parameters

in	*event*	- The event to report
in	*file*	- The file that is reporting this event
in	*line*	- The line in the file that is reporting this event
in	*remarks*	- Additional remarks to report, may be set to NULL

2.7.2.3.3.2 void REPORT_EVENT (uint32_t *_err*)

Reports a system event by calling osReportEvent() - may be disabled, based on compilation flags

Parameters

in	*_err*	- The event to report

Warning

The function signature is for documentation purposes only

2.7.2.3.3.3 void REPORT_EVENT_MESSAGE (uint32_t *_err,* const char * *message*)

Reports a system event by calling osReportEvent() - may be disabled, based on compilation flags

Parameters

in	*_err*	- The event to report
in	*message*	- The additional remarks to report

Warning

The function signature is for documentation purposes only

2.7.2.3.3.4 void REPORT_ERROR (uint32_t *_err*)

Reports a system error by calling osReportEvent() - may be disabled, based on compilation flags

Parameters

in	*_err*	- The event to report

Warning

The function signature is for documentation purposes only

2.7.2.3.3.5 void REPORT_ERROR_MESSAGE (uint32_t *_err*, const char * *message*)

Reports a system error by calling osReportEvent() - may be disabled, based on compilation flags

Parameters

in	*_err*	- The event to report
in	*message*	- The additional remarks to report

Warning

The function signature is for documentation purposes only

2.7.2.3.3.6 void RETURN_ERROR (uint32_t *_err*)

Reports a system error by calling osReportEvent() and returns from the calling function with the error code - may be disabled, based on compilation flags

Parameters

in	*_err*	- The event to report and return

Warning

The function signature is for documentation purposes only

2.7.2.3.3.7 void RETURN_ERROR_MESSAGE (uint32_t *_err,* const char * *message*)

Reports a system error by calling osReportEvent() and returns from the calling function with the error code - may be disabled, based on compilation flags

Parameters

in	*_err*	- The event to report
in	*message*	- The additional remarks to report

Warning

The function signature is for documentation purposes only

2.7.2.4 Accessor API

2.7.2.4.1 Overview

Macros for accessing registers and memory.

These macros are written in such a manner to make them portable across platforms and architectures. Any address that is passed should be passed as dereferenced pointer

Warning

Although the macros for accessing memory and registers are, in general, implemented in the same manner; for future compatibility use the proper version

These macros are written in such a manner to make them portable across platforms and architectures. Any address that is passed should be passed as dereferenced pointer

Warning

Although the macros for accessing memory and registers are, in general, implemented in the same manner; for future compatibility use the proper version

Modules

- SC39XX Accessor API

Functions

- void READ_DATA8 (uint8_t *data, uint8_t arg)
- void READ_DATA16 (uint16_t *data, uint16_t arg)
- void READ_DATA32 (uint32_t *data, uint32_t arg)
- void READ_DATA64 (uint64_t *data, uint64_t arg)
- uint8_t GET_DATA8 (uint8_t arg)
- uint16_t GET_DATA16 (uint13_t arg)
- uint32_t GET_DATA32 (uint32_t arg)
- uint64_t GET_DATA64 (uint64_t arg)
- void WRITE_UINT8 (uint8_t arg, uint8_t data)
- void WRITE_UINT16 (uint16_t arg, uint16_t data)
- void WRITE_UINT32 (uint32_t arg, uint32_t data)
- void WRITE_UINT64 (uint64_t arg, uint64_t data)
- void READ_MEM_UINT8 (uint8_t *data, uint8_t arg)
- void READ_MEM_UINT16 (uint16_t *data, uint16_t arg)
- void READ_MEM_UINT32 (uint32_t *data, uint32_t arg)
- uint8_t GET_MEM_UINT8 (uint8_t arg)
- uint16_t GET_MEM_UINT16 (uint16_t arg)
- uint32_t GET_MEM_UINT32 (uint32_t arg)
- void WRITE_MEM_UINT8 (uint8_t arg, uint8_t data)
- void WRITE_MEM_UINT16 (uint16_t arg, uint16_t data)
- void WRITE_MEM_UINT32 (uint32_t arg, uint32_t data)
- void CLEAR_UINT8 (uint8_t addr, uint8_t bitmask)
- void SET_UINT8 (uint8_t addr, uint8_t bitmask)
- void CLEAR_UINT16 (uint16_t addr, uint16_t bitmask)
- void SET_UINT8 (uint16_t addr, uint16_t bitmask)
- void CLEAR_UINT32 (uint32_t addr, uint32_t bitmask)
- void SET_UINT8 (uint32_t addr, uint32_t bitmask)

2.7.2.4.2 Function Documentation

2.7.2.4.2.1 void READ_DATA8 (uint8_t * *data*, uint8_t *arg*)

Read a uint8_t register and store the value

This function performs: data = (uint8_t)(arg)

Parameters

out	*data*	- The location to store the read data in
in	*arg*	- The address to read from

Warning

The function signature is for documentation purposes only

2.7.2.4.2.2 void READ_DATA16 (uint16_t * *data*, uint16_t *arg*)

Read a uint16_t register and store the value

This function performs: data = (uint16_t)(arg)

Parameters

out	*data*	- The location to store the read data in
in	*arg*	- The address to read from

Warning

The function signature is for documentation purposes only

2.7.2.4.2.3 void READ_DATA32 (uint32_t * *data*, uint32_t *arg*)

Read a uint32_t register and store the value

This function performs: data = (uint32_t)(arg)

Parameters

out	*data*	- The location to store the read data in
in	*arg*	- The address to read from

Warning

The function signature is for documentation purposes only

2.7.2.4.2.4 void READ_DATA64 (uint64_t * *data*, uint64_t *arg*)

Read a uint64_t register and store the value

This function performs: data = (uint64_t)(arg)

Parameters

out	*data*	- The location to store the read data in
in	*arg*	- The address to read from

Warning

The function signature is for documentation purposes only

2.7.2.4.2.5 uint8_t GET_DATA8 (uint8_t *arg*)

Read a uint8_t register and return the value

This function performs: (uint8_t)(arg)

Parameters

in	*arg*	- The address to read from

Returns

value in register

Warning

The function signature is for documentation purposes only

2.7.2.4.2.6 uint16_t GET_DATA16 (uint13_t *arg*)

Read a uint16_t register and return the value

This function performs: (uint16_t)(arg)

Parameters

in	*arg*	- The address to read from

Returns

value in register

Warning

The function signature is for documentation purposes only

2.7.2.4.2.7 uint32_t GET_DATA32 (uint32_t *arg*)

Read a uint32_t register and return the value

This function performs: (uint32_t)(arg)

Parameters

in	*arg*	- The address to read from

Returns

value in register

Warning

The function signature is for documentation purposes only

2.7.2.4.2.8 uint64_t GET_DATA64 (uint64_t *arg*)

Read a uint54_t register and return the value

This function performs: (uint64_t)(arg)

Parameters

in	*arg*	- The address to read from

Returns

value in register

Warning

The function signature is for documentation purposes only

2.7.2.4.2.9 void WRITE_UINT8 (uint8_t *arg*, uint8_t *data*)

Write a uint8_t register

This function performs: arg = (uint8_t)(data)

Parameters

in	*arg*	- The address to write to
in	*data*	- The data value to write

Warning

The function signature is for documentation purposes only

2.7.2.4.2.10 void WRITE_UINT16 (uint16_t *arg*, uint16_t *data*)

Write a uint16_t register

This function performs: arg = (uint16_t)(data)

Parameters

in	*arg*	- The address to write to
in	*data*	- The data value to write

Warning

The function signature is for documentation purposes only

2.7.2.4.2.11 void WRITE_UINT32 (uint32_t *arg*, uint32_t *data*)

Write a uint32_t register

This function performs: arg = (uint32_t)(data)

Parameters

in	*arg*	- The address to write to
in	*data*	- The data value to write

Warning

The function signature is for documentation purposes only

2.7.2.4.2.12 void WRITE_UINT64 (uint64_t *arg*, uint64_t *data*)

Write a uint64_t register

This function performs: arg = (uint64_t)(data)

Parameters

in	*arg*	- The address to write to
in	*data*	- The data value to write

Warning

The function signature is for documentation purposes only

2.7.2.4.2.13 void READ_MEM_UINT8 (uint8_t * *data*, uint8_t *arg*)

Read a uint8_t memory location and store the value

This function performs: data = (uint8_t)(arg)

Parameters

out	*data*	- The location to store the read data in
in	*arg*	- The address to read from

Warning

The function signature is for documentation purposes only

2.7.2.4.2.14 void READ_MEM_UINT16 (uint16_t * *data*, uint16_t *arg*)

Read a uint16_t memory location and store the value

This function performs: data = (uint16_t)(arg)

Parameters

out	*data*	- The location to store the read data in
in	*arg*	- The address to read from

Warning

The function signature is for documentation purposes only

2.7.2.4.2.15 void READ_MEM_UINT32 (uint32_t * *data*, uint32_t *arg*)

Read a uint32_t memory location and store the value

This function performs: data = (uint32_t)(arg)

Parameters

out	*data*	- The location to store the read data in
in	*arg*	- The address to read from

Warning

The function signature is for documentation purposes only

2.7.2.4.2.16 uint8_t GET_MEM_UINT8 (uint8_t *arg*)

Read a uint8_t memory location and return the value

This function performs: (uint8_t)(arg)

Parameters

in	*arg*	- The address to read from

Returns

value in memory

Warning

The function signature is for documentation purposes only

2.7.2.4.2.17 uint16_t GET_MEM_UINT16 (uint16_t *arg*)

Read a uint16_t memory location and return the value

This function performs: (uint16_t)(arg)

Parameters

in	*arg*	- The address to read from

Returns

value in memory

Warning

The function signature is for documentation purposes only

2.7.2.4.2.18 uint32_t GET_MEM_UINT32 (uint32_t *arg*)

Read a uint32_t memory location and return the value

This function performs: (uint32_t)(arg)

Parameters

in	*arg*	- The address to read from

Returns

value in memory

Warning

The function signature is for documentation purposes only

2.7.2.4.2.19 void WRITE_MEM_UINT8 (uint8_t *arg*, uint8_t *data*)

Write a uint8_t memory location

This function performs: arg = (uint8_t)(data)

Parameters

in	*arg*	- The address to write to
in	*data*	- The data value to write

Warning

The function signature is for documentation purposes only

2.7.2.4.2.20 void WRITE_MEM_UINT16 (uint16_t *arg*, uint16_t *data*)

Write a uint16_t memory location

This function performs: arg = (uint16_t)(data)

Parameters

in	*arg*	- The address to write to
in	*data*	- The data value to write

Warning

The function signature is for documentation purposes only

2.7.2.4.2.21 void WRITE_MEM_UINT32 (uint32_t *arg*, uint32_t *data*)

Write a uint32_t memory location

This function performs: arg = (uint32_t)(data)

Parameters

in	*arg*	- The address to write to
in	*data*	- The data value to write

Warning

The function signature is for documentation purposes only

2.7.2.4.2.22 void CLEAR_UINT8 (uint8_t *addr,* uint8_t *bitmask*)

Clear a bitmask on a uint8_t register

Parameters

in	*addr*	- The address to clear
in	*bitmask*	- The bits to clear - set 1 in each bit location that should be cleared

Warning

The function signature is for documentation purposes only

2.7.2.4.2.23 void SET_UINT8 (uint8_t *addr,* uint8_t *bitmask*)

SET a bitmask on a uint8_t register

Parameters

in	*addr*	- The address to clear
in	*bitmask*	- The bits to set - set 1 in each bit location that should be set

Warning

The function signature is for documentation purposes only

2.7.2.4.2.24 void CLEAR_UINT16 (uint16_t *addr,* uint16_t *bitmask*)

Clear a bitmask on a uint16_t register

Parameters

in	*addr*	- The address to clear
in	*bitmask*	- The bits to clear - set 1 in each bit location that should be cleared

Warning

The function signature is for documentation purposes only

2.7.2.4.2.25 void SET_UINT8 (uint16_t *addr,* uint16_t *bitmask*)

SET a bitmask on a uint16_t register

Parameters

in	*addr*	- The address to clear
in	*bitmask*	- The bits to set - set 1 in each bit location that should be set

Warning

The function signature is for documentation purposes only

2.7.2.4.2.26 void CLEAR_UINT32 (uint32_t *addr,* uint32_t *bitmask*)

Clear a bitmask on a uint32_t register

Parameters

in	*addr*	- The address to clear
in	*bitmask*	- The bits to clear - set 1 in each bit location that should be cleared

Warning

The function signature is for documentation purposes only

2.7.2.4.2.27 void SET_UINT8 (uint32_t *addr,* uint32_t *bitmask*)

SET a bitmask on a uint32_t register

Parameters

in	*addr*	- The address to clear
in	*bitmask*	- The bits to set - set 1 in each bit location that should be set

Warning

The function signature is for documentation purposes only

2.7.2.4.3 SC39XX Accessor API

`Macros for accessing registers and memory (SC39XX specific).`

These macros are written in such a manner to make them portable across platforms and architectures. Any address that is passed should be passed as dereferenced pointer

2.7.2.5 General System Functions

2.7.2.5.1 Overview

`Kernel initialization and general system functions.`

Data Structures

- struct os_version_t
- struct os_core_info_t

Macros

- #define STACK_MINIMAL_ALIGNMENT 7
- #define HAVE_OS_ERROR_DEBUG
- #define OS_COMPILER_ASSERT(expression) cw_assert(expression)
- #define OS_ASSERT osErrorDebug()
- #define OS_ASSERT_COND(expression) { if (!(expression)) osErrorDebug(); }
- #define OS_ASSERT_COMPILER_COND(expression)
- #define osGetDeviceID() (g_dev_id + 0)
- #define osGetSocMasterCore() (g_core_info.soc_master_core + 0)
- #define osGetSocCoreID() (g_core_info.soc_core_num + 0)
- #define osGetMpicCoreID() (g_core_info.soc_core_num + OS_MPIC_CORES_ID_OFFSET)
- #define osGetMpicMasterCore() (g_core_info.soc_master_core + OS_MPIC_CORES_ID_OFFS↩ ET)
- #define osGetClusterMasterCore() (g_core_info.cluster_master_core + 0)
- #define osGetClusterCoreID() (g_core_info.cluster_core_num + 0)
- #define osGetClusterNum() (g_core_info.cluster_num + 0)
- #define osGetMasterCore() osGetSocMasterCore()
- #define osGetCoreID() osGetSocCoreID()

Functions

- void SYS_NOP ()
- void COPY_ALIGNED_2 (uint16_t *x, uint16_t *y)
- void osTestDebug ()
- os_status osInitialize ()
- os_status osStart (os_background_task_function background_task)
- void osGetVersionInfo (os_version_t *version_info)
- void osErrorDebug ()
- uint32_t osGetSystemClock ()
- uint8_t osGetNumberOfCores ()
- void osWait (uint32_t num) __attribute__((noinline))
- int osMulticoreSupport ()

Variables

- os_core_info_t g_core_info

2.7.2.5.2 Data Structure Documentation

2.7.2.5.2.1 struct os_version_t

OS Version Information Structure.

Data Fields

- uint16_t major
- uint16_t minor
- uint16_t custom
- uint16_t patch

2.7.2.5.2.1.1 Field Documentation

2.7.2.5.2.1.2 uint16_t os_version_t::major

Major version number.

2.7.2.5.2.1.3 uint16_t os_version_t::minor

Minor version number.

2.7.2.5.2.1.4 uint16_t os_version_t::custom

Custom build number.

2.7.2.5.2.1.5 uint16_t os_version_t::patch

Patch number.

2.7.2.5.2.2 struct os_core_info_t

Core Identity Information Structure.

Data Fields

- uint8_t soc_core_num
- uint8_t cluster_core_num
- uint8_t cluster_num
- uint8_t soc_master_core
- uint8_t cluster_master_core

2.7.2.5.2.2.1 Field Documentation

2.7.2.5.2.2.2 uint8_t os_core_info_t::soc_core_num

Core number within SoC - hardware defined.

2.7.2.5.2.2.3 uint8_t os_core_info_t::cluster_core_num

Core number within cluster - hardware defined.

2.7.2.5.2.2.4 uint8_t os_core_info_t::cluster_num

Cluster number within SoC - hardware defined.

2.7.2.5.2.2.5 uint8_t os_core_info_t::soc_master_core

SoC master core number - software defined.

2.7.2.5.2.2.6 uint8_t os_core_info_t::cluster_master_core

Cluster master core number - software defined.

2.7.2.5.3 Macro Definition Documentation

2.7.2.5.3.1 #define STACK_MINIMAL_ALIGNMENT 7

Top of stack minimal alignment.

2.7.2.5.3.2 #define HAVE_OS_ERROR_DEBUG

If defined OS runs in OS error debug mode.

2.7.2.5.3.3 #define OS_COMPILER_ASSERT(*expression*) cw_assert(expression)

Compiler assertion, for performance.

The support is restricted to establishing range (minimum, maximum values) and modulo (or alignment) for user variables. The condition can establish order relations (>,>=,<,<=,==) between C variables and constants, or modulo properties (var&0x01..1 == 0 OR varconst == 0). If

2.7.2.5.3.4 #define OS_ASSERT osErrorDebug()

Assertion macro, for debugging - calls osErrorDebug()

2.7.2.5.3.5 #define OS_ASSERT_COND(*expression*) { if (!(expression)) osErrorDebug(); }

Assertion if condition is false, for debugging - calls osErrorDebug()

2.7.2.5.3.6 #define OS_ASSERT_COMPILER_COND(*expression*)

Value:

```
OS_ASSERT_COND(expression) \
    OS_COMPILER_ASSERT(expression)
```

Compiler and conditional assertions, for performance and debugging.

The support is restricted to establishing range (minimum, maximum values) and modulo (or alignment) for user variables. The condition can establish order relations (>,>=,<,<=,==) between C variables and constants, or modulo properties (var&0x01..1 == 0 OR varconst == 0). If

2.7.2.5.3.7 #define osGetDeviceID( ) (g_dev_id + 0)

Gets the device ID.

Returns

The device ID

2.7.2.5.3.8 #define osGetSocMasterCore( ) (g_core_info.soc_master_core + 0)

Retrieves the ID of the SoC master core.

Returns

The ID of the master core

2.7.2.5.3.9 #define osGetSocCoreID( ) (g_core_info.soc_core_num + 0)

Retrieves the ID of the core within the SoC.

Returns

The ID of the core

2.7.2.5.3.10 #define osGetMpicCoreID( ) (g_core_info.soc_core_num + OS_MPIC_CORES_ID_OFFSET)

Retrieves the ID of the core for MPIC use.

Returns

The ID of the core

2.7.2.5.3.11 #define osGetMpicMasterCore( ) (g_core_info.soc_master_core + OS_MPIC_CORES_ID_OFFSET)

Retrieves the ID of the master core for MPIC use.

Returns

The ID of the master core

2.7.2.5.3.12 #define osGetClusterMasterCore( ) (g_core_info.cluster_master_core + 0)

Retrieves the ID of the core within the cluster.

Returns

The ID of the master core

2.7.2.5.3.13 #define osGetClusterCoreID( ) (g_core_info.cluster_core_num + 0)

Retrieves the ID of the core within the cluster.

Returns

The ID of the master core

2.7.2.5.3.14 #define osGetClusterNum( ) (g_core_info.cluster_num + 0)

Retrieves the cluster number within the SoC.

Returns

The ID of cluster

2.7.2.5.3.15 #define osGetMasterCore( ) osGetSocMasterCore()

Retrieves the ID of the master core.

Returns

The ID of the master core

Warning

Backward compatible use - deprecation alert

2.7.2.5.3.16 #define osGetCoreID( ) osGetSocCoreID()

Retrieves the ID of calling master core.

Returns

The ID of the calling core

Warning

Backward compatible use - deprecation alert

2.7.2.5.4 Function Documentation

2.7.2.5.4.1 void SYS_NOP ( )

OS nop function

Warning

The function signature is for documentation purposes only

2.7.2.5.4.2 void COPY_ALIGNED_2 (uint16_t * *x*, uint16_t * *y*)

Copies 4 bytes from y to x using 2 writes - 2 bytes each time.

Parameters

in	*x*	- Destination address.
in	*y*	- Source address.

Warning

The function signature is for documentation purposes only

2.7.2.5.4.3 void osTestDebug ( )

Function used for Auto-test process for test passed/failed indication.

Warning

The function signature is for documentation purposes only

2.7.2.5.4.4 os_status osInitialize ( )

Initializes the OS.

This function initializes the various modules of the OS, using the parameters taken from the configuration file.

Return values

OS_SUCCESS	- kernel and drivers initialized properly.

Warning

Must be called before osStart().

2.7.2.5.4.5 os_status osStart (os_background_task_function *background_task*)

Starts the OS operation.

This function receives a pointer to a background task and switches to that task. If the background task runs infinitely (as it should), this function never returns.

This function also activates the system tick, if it is required.

Parameters

in	*background_task*	- The background task to run.

Returns

Error status, encoded in os_error.h, for any reason; as this function does not, normally, return

Warning

osInitialize() must be called before osStart().

```
Do not use local variables here, because this function does
not return, and its stack will not be cleared.
```

2.7.2.5.4.6 void osGetVersionInfo (os_version_t * *version_info*)

Returns the OS version information in the structure passed in.

Parameters

out	*version_info*	- Pointer to a version information structure.

2.7.2.5.4.7 void osErrorDebug ( )

Breaks execution (used for debug).

2.7.2.5.4.8 uint32_t osGetSystemClock ( )

Gets the system clock value.

The returned value is the same value that was given in the configuration structure when the OS was initialized.

Returns

System clock (MHz)

2.7.2.5.4.9 uint8_t osGetNumberOfCores ( )

Retrieves the number of cores in the system.

Returns

The number of cores in the system

2.7.2.5.4.10 void osWait (uint32_t *num*)

Macro for looping waiting in loop.

The macro will not get optimized by the compiler, at any level of optimization, and will loop by performing a simple for loop

Parameters

num	- number of times to loop

2.7.2.5.4.11 int osMulticoreSupport ( )

Returns the value of OS_MULTICORE

Returns

1 if OS_MULTICORE == 1 or 0 otherwise

2.7.2.5.5 Variable Documentation

2.7.2.5.5.1 os_core_info_t g_core_info

Information structure including information regarding *this* core; Users MAY NOT access this variable directly!!!

2.7.2.6 System Information

2.7.2.6.1 Overview

```
System information structures and function.
```

Data Structures

- struct os_task_info_t
- struct os_hwi_info_t
- struct os_swi_info_t
- struct os_sys_info_t

Enumerations

- enum os_context_info_t { OS_CONTEXT_HWI = 1, OS_CONTEXT_SWI, OS_CONTEXT_TASK, OS_CONTEXT_NOT_CREATED }

Functions

- os_status osSysInfoGet (os_sys_info_t *sys_info)

2.7.2.6.2 Data Structure Documentation

2.7.2.6.2.1 struct os_task_info_t

task information

Data Fields

- uint32_t * stack_pointer
- uint32_t * top_of_stack
- uint32_t * task_function
- os_task_priority priority
- os_task_handle task_handle

2.7.2.6.2.1.1 Field Documentation

2.7.2.6.2.1.2 uint32_t* os_task_info_t::stack_pointer

Current SP in the task.

2.7.2.6.2.1.3 uint32_t* os_task_info_t::top_of_stack

Top of task's stack.

2.7.2.6.2.1.4 uint32_t* os_task_info_t::task_function

Function ran in this task.

2.7.2.6.2.1.5 os_task_priority os_task_info_t::priority

Priority of task.

2.7.2.6.2.1.6 os_task_handle os_task_info_t::task_handle

Task ID.

2.7.2.6.2.2 struct os_hwi_info_t

HW interrupt information.

Data Fields

- uint32_t vector

2.7.2.6.2.2.1 Field Documentation

2.7.2.6.2.2.2 uint32_t os_hwi_info_t::vector

Interrupt Vector.

2.7.2.6.2.3 struct os_swi_info_t

SW interrupt information.

Data Fields

- uint32_t swi_num

2.7.2.6.2.3.1 Field Documentation

2.7.2.6.2.3.2 uint32_t os_swi_info_t::swi_num

SWI Number.

2.7.2.6.2.4 struct os_sys_info_t

OS current system information.

Data Fields

- os_task_info_t active_task_info
- os_swi_info_t active_swi_info
- os_hwi_info_t active_hwi_info
- os_context_info_t os_context_info

2.7.2.6.2.4.1 Field Documentation

2.7.2.6.2.4.2 os_task_info_t os_sys_info_t::active_task_info

Currentely active task info.

2.7.2.6.2.4.3 os_swi_info_t os_sys_info_t::active_swi_info

Currentely active SWI info.

2.7.2.6.2.4.4 os_hwi_info_t os_sys_info_t::active_hwi_info

Not supported.

2.7.2.6.2.4.5 os_context_info_t os_sys_info_t::os_context_info

Current context.

2.7.2.6.3 Enumeration Type Documentation

2.7.2.6.3.1 enum os_context_info_t

OS contexts.

Enumerator

OS_CONTEXT_HWI Current context is hardware interrupt.
OS_CONTEXT_SWI Current context is software interrupt.
OS_CONTEXT_TASK Current context is task.
OS_CONTEXT_NOT_CREATED Only valid during initialization.

2.7.2.6.4 Function Documentation

2.7.2.6.4.1 os_status osSysInfoGet (os_sys_info_t * *sys_info*)

Returns current OS system information.

Parameters

out	*sys_info*	- structure to fill OS info in.

Return values

OS_SUCCESS	

Warning

None.

2.7.3 Debug Utilities

2.7.3.1 Overview

Modules

- CLASS Profiling API

- Logging API

2.7.3.2 CLASS Profiling API

2.7.3.2.1 Overview

```
CLASS debug and profiling Monitor API.
```

Modules

- SC39XX DTU API
- B4860 Debug Print API

2.7.3.2.2 SC39XX DTU API

2.7.3.2.2.1 Overview

Macros

- #define NUMBER_OF_CORE_COMMAND_REGISTERS 4
- #define NUMBER_OF_CORE_COMMAND_DATA_REGISTERS 4
- #define NUMBER_OF_PROFILING_COUNTER_VALUE_REGISTERS_A 3
- #define NUMBER_OF_PROFILING_SNAPSHOT_REGISTERS_A 3
- #define NUMBER_OF_PROFILING_COUNTER_VALUE_REGISTERS_B 3
- #define NUMBER_OF_PROFILING_SNAPSHOT_REGISTERS_B 3
- #define NUMBER_OF_INDIRECT_EVENT_COND_TRASITION_CONF_REGISTERS 4
- #define OS_DTU_MODE(cev, triad, counter) ((cev << 16) | (triad << 8) | counter)

Enumerations

- enum b486x_dtu_triad { DTU_TRIAD_A = 0, DTU_TRIAD_B = 1 }
- enum b486x_dtu_triad_counter { DTU_TRIAD_COUNTER_0 = 0, DTU_TRIAD_COUNTER_1 = 1, DTU_TRIAD_COUNTER_2 = 2 }
- enum b486x_dtu_cev_triad_a {
  DTU_TRIAD_A_CEV_NULL = 0, DTU_CEV_CE1 = 0, DTU_CEV_CBP1 = 2,
  DTU_CEV_CCH = 5, DTU_CEV_CCS1 = 6, DTU_CEV_ED1 = 10,
  DTU_CEV_EW1 = 11, DTU_CEV_EIN1 = 12, DTU_CEV_IAS = 20,
  DTU_CEV_IPF = 21, DTU_CEV_DGCI = 25, DTU_CEV_DLSR = 26,
  DTU_CEV_DLPF = 27, DTU_CEV_DLH1 = 28, DTU_CEV_DLSP = 29,
  DTU_CEV_DSAS = 35, DTU_CEV_BL12 = 43, DTU_CEV_KED1 = 50,
  DTU_CEV_KAOT = 51, DTU_CEV_KEXT = 52 }
- enum b486x_dtu_cev_triad_b {
  DTU_TRIAD_B_CEV_NULL = 0, DTU_CEV_CE2 = 0, DTU_CEV_CBP2 = 2,
  DTU_CEV_CLP = 3, DTU_CEV_CRW = 4, DTU_CEV_CCS2 = 6,
  DTU_CEV_ED2 = 10, DTU_CEV_IL12 = 20, DTU_CEV_IAH = 21,
  DTU_CEV_DGAH = 25, DTU_CEV_DL12 = 26, DTU_CEV_DLH2 = 28,
  DTU_CEV_DSGE = 35, DTU_CEV_KED2 = 50, DTU_CEV_KTOT = 51,
  DTU_CEV_KDIN = 52, DTU_CEV_KNSP = 53 }

- enum b486x_dtu_profiling_mode {
  DTU_TRAP_INSTRUCTION = OS_DTU_MODE(DTU_CEV_CE1, DTU_TRIAD_A, DTU_↩
  TRIAD_COUNTER_0), DTU_CRITICAL_INTERRUPT = OS_DTU_MODE(DTU_CEV_CE1,
  DTU_TRIAD_A, DTU_TRIAD_COUNTER_1), DTU_NON_CRITICAL_INTERRUPT = OS_↩
  DTU_MODE(DTU_CEV_CE1, DTU_TRIAD_A, DTU_TRIAD_COUNTER_2),
  DTU_MMU_EXCEPTIONS = OS_DTU_MODE(DTU_CEV_CE2, DTU_TRIAD_B, DTU_↩
  TRIAD_COUNTER_0), DTU_DEBUG_EXCEPTIONS = OS_DTU_MODE(DTU_CEV_CE2,
  DTU_TRIAD_B, DTU_TRIAD_COUNTER_1), DTU_OTHER_EXCEPTIONS = OS_DTU_M↩
  ODE(DTU_CEV_CE2, DTU_TRIAD_B, DTU_TRIAD_COUNTER_2),
  DTU_TOTAL_VLES = OS_DTU_MODE(DTU_CEV_CBP1, DTU_TRIAD_A, DTU_TRIA↩
  D_COUNTER_0), DTU_COF_VLES = OS_DTU_MODE(DTU_CEV_CBP1, DTU_TRIAD_A,
  DTU_TRIAD_COUNTER_1), DTU_COF_CORRECTLY_PREDICTED = OS_DTU_MODE(D↩
  TU_CEV_CBP1, DTU_TRIAD_A, DTU_TRIAD_COUNTER_2),
  DTU_ALL_BTB_COF = OS_DTU_MODE(DTU_CEV_CBP2, DTU_TRIAD_B, DTU_TRIA↩
  D_COUNTER_0), DTU_BTB_COF_WRONGLY_PREDICTED_NOT_IN_BTB = OS_DTU_↩
  MODE(DTU_CEV_CBP2, DTU_TRIAD_B, DTU_TRIAD_COUNTER_1), DTU_BTB_COF↩
  _WRONGLY_PREDICTED_IN_BTB = OS_DTU_MODE(DTU_CEV_CBP2, DTU_TRIAD_B,
  DTU_TRIAD_COUNTER_2),
  DTU_TOTAL_HW_EOL_COF = OS_DTU_MODE(DTU_CEV_CLP, DTU_TRIAD_B, DTU↩
  _TRIAD_COUNTER_0), DTU_WRONGLY_PREDICTED_BUFFERED_EOL_COF = OS_D↩
  TU_MODE(DTU_CEV_CLP, DTU_TRIAD_B, DTU_TRIAD_COUNTER_1), DTU_OTHER_↩
  WRONGLY_PREDICTED_EOL_COF = OS_DTU_MODE(DTU_CEV_CLP, DTU_TRIAD_B,
  DTU_TRIAD_COUNTER_2),
  DTU_APPLICATION_CYCLES = OS_DTU_MODE(DTU_CEV_CCH, DTU_TRIAD_A, DT↩
  U_TRIAD_COUNTER_0), DTU_BUBBLES = OS_DTU_MODE(DTU_CEV_CCH, DTU_TRI↩
  AD_A, DTU_TRIAD_COUNTER_2), DTU_NO_BUBBLES = OS_DTU_MODE(DTU_CEV_↩
  CCS1, DTU_TRIAD_A, DTU_TRIAD_COUNTER_0),
  DTU_COF = OS_DTU_MODE(DTU_CEV_CCS1, DTU_TRIAD_A, DTU_TRIAD_COUNTE↩
  R_1), DTU_INTERLOCK = OS_DTU_MODE(DTU_CEV_CCS1, DTU_TRIAD_A, DTU_TRI↩
  AD_COUNTER_2), DTU_DATA_MEMORY_HOLDS_AND_FREEZE = OS_DTU_MODE(D↩
  TU_CEV_CCS2, DTU_TRIAD_B, DTU_TRIAD_COUNTER_0),
  DTU_PROGRAM_STARVATION = OS_DTU_MODE(DTU_CEV_CCS2, DTU_TRIAD_B, D↩
  TU_TRIAD_COUNTER_1), DTU_REWIND_CYCLES = OS_DTU_MODE(DTU_CEV_CCS2,
  DTU_TRIAD_B, DTU_TRIAD_COUNTER_2), DTU_DATA_REWIND_EVENTS = OS_DTU↩
  _MODE(DTU_CEV_CRW, DTU_TRIAD_B, DTU_TRIAD_COUNTER_0),
  DTU_PROGRAM_REWIND_EVENTS = OS_DTU_MODE(DTU_CEV_CRW, DTU_TRIAD_B,
  DTU_TRIAD_COUNTER_1), DTU_REWIND_EVENTS = OS_DTU_MODE(DTU_CEV_CRW,
  DTU_TRIAD_B, DTU_TRIAD_COUNTER_2), DTU_IEU_Q0 = OS_DTU_MODE(DTU_CE↩
  V_ED1, DTU_TRIAD_A, DTU_TRIAD_COUNTER_0),
  DTU_IEU_Q1 = OS_DTU_MODE(DTU_CEV_ED1, DTU_TRIAD_A, DTU_TRIAD_COUN↩
  TER_1), DTU_RELOAD_COUNTER_0_RZ = OS_DTU_MODE(DTU_CEV_ED1, DTU_T↩
  RIAD_A, DTU_TRIAD_COUNTER_2), DTU_PCDA0 = OS_DTU_MODE(DTU_CEV_EW1,
  DTU_TRIAD_A, DTU_TRIAD_COUNTER_1),
  DTU_PCDA2 = OS_DTU_MODE(DTU_CEV_EW1, DTU_TRIAD_A, DTU_TRIAD_COUNT↩
  ER_2), DTU_DEBUGEV_0 = OS_DTU_MODE(DTU_CEV_EIN1, DTU_TRIAD_A, DTU_TR↩

IAD_COUNTER_0), DTU_DEBUGEV_2 = OS_DTU_MODE(DTU_CEV_EIN1, DTU_TRIA↩
D_A, DTU_TRIAD_COUNTER_1),
DTU_DEBUGEV_3 = OS_DTU_MODE(DTU_CEV_EIN1, DTU_TRIAD_A, DTU_TRIAD↩
_COUNTER_2), DTU_PROGRAM_ACCESS_L1_HITS = OS_DTU_MODE(DTU_CEV_IAS,
DTU_TRIAD_A, DTU_TRIAD_COUNTER_0), DTU_PROGRAM_ACCESS_L1_PREFATCH↩
_HITS = OS_DTU_MODE(DTU_CEV_IAS, DTU_TRIAD_A, DTU_TRIAD_COUNTER_1),
DTU_PROGRAM_ACCESS_L1_MISS = OS_DTU_MODE(DTU_CEV_IAS, DTU_TRIAD_A,
DTU_TRIAD_COUNTER_2), DTU_L2_PROGRAM_ACCESS_HITS_SHARED = OS_DT↩
U_MODE(DTU_CEV_IL12, DTU_TRIAD_B, DTU_TRIAD_COUNTER_0), DTU_L2_PRO↩
GRAM_ACCESS_HITS_MODIFIED = OS_DTU_MODE(DTU_CEV_IL12, DTU_TRIAD_B,
DTU_TRIAD_COUNTER_1),
DTU_L2_PROGRAM_ACCESS_HITS = OS_DTU_MODE(DTU_CEV_IL12, DTU_TRIAD_B,
DTU_TRIAD_COUNTER_2), DTU_CME_PF_ACCESSES_L1_HIT = OS_DTU_MODE(DTU↩
_CEV_IPF, DTU_TRIAD_A, DTU_TRIAD_COUNTER_0), DTU_CME_PF_ACCESSES_L1_↩
MISS = OS_DTU_MODE(DTU_CEV_IPF, DTU_TRIAD_A, DTU_TRIAD_COUNTER_2),
DTU_PROG_CACHEABLE_HOLD_CONTENTION_CYCLES = OS_DTU_MODE(DTU_CE↩
V_IAH, DTU_TRIAD_B, DTU_TRIAD_COUNTER_0), DTU_PROG_NON_CACHEABLE_↩
HOLDS_MISCELLANEOUS = OS_DTU_MODE(DTU_CEV_IAH, DTU_TRIAD_B, DTU_T↩
RIAD_COUNTER_1), DTU_PROG_REWIND_EVENTS = OS_DTU_MODE(DTU_CEV_IAH,
DTU_TRIAD_B, DTU_TRIAD_COUNTER_2),
DTU_DATA_CACHE_HOLDS = OS_DTU_MODE(DTU_CEV_DGAH, DTU_TRIAD_B, DT↩
U_TRIAD_COUNTER_0), DTU_FATA_STORE_FREEZE = OS_DTU_MODE(DTU_CEV_D↩
GAH, DTU_TRIAD_B, DTU_TRIAD_COUNTER_1), DTU_DATA_REWIND_EVENTS_HL =
OS_DTU_MODE(DTU_CEV_DGAH, DTU_TRIAD_B, DTU_TRIAD_COUNTER_2),
DTU_L1_DATA_INVALIDATE_FROM_L2 = OS_DTU_MODE(DTU_CEV_DGCI, DTU_↩
TRIAD_A, DTU_TRIAD_COUNTER_0), DTU_CME_INVALIDATION_ACCESSES = OS↩
_DTU_MODE(DTU_CEV_DGCI, DTU_TRIAD_A, DTU_TRIAD_COUNTER_1), DTU_L1↩
_DATA_INVALIDATE_FROM_INTER_CONFLICTS = OS_DTU_MODE(DTU_CEV_DGCI,
DTU_TRIAD_A, DTU_TRIAD_COUNTER_2),
DTU_DATA_ACCESS_L1_HITS = OS_DTU_MODE(DTU_CEV_DLSR, DTU_TRIAD_A, D↩
TU_TRIAD_COUNTER_0), DTU_DATA_ACCESS_L1_PF_HITS = OS_DTU_MODE(DTU_↩
CEV_DLSR, DTU_TRIAD_A, DTU_TRIAD_COUNTER_1), DTU_DATA_ACCESS_L1_MISS
= OS_DTU_MODE(DTU_CEV_DLSR, DTU_TRIAD_A, DTU_TRIAD_COUNTER_2),
DTU_L2_DATA_ACCESS_HITS_SHARED = OS_DTU_MODE(DTU_CEV_DL12, DTU_TR↩
IAD_B, DTU_TRIAD_COUNTER_0), DTU_L2_DATA_ACCESS_HITS_MODIFIED = OS_D↩
TU_MODE(DTU_CEV_DL12, DTU_TRIAD_B, DTU_TRIAD_COUNTER_1), DTU_L2_DA↩
TA_ACCESS_MISS = OS_DTU_MODE(DTU_CEV_DL12, DTU_TRIAD_B, DTU_TRIAD_↩
COUNTER_2),
DTU_CME_PF_ACCESS_L1_HIT = OS_DTU_MODE(DTU_CEV_DLPF, DTU_TRIAD_A,
DTU_TRIAD_COUNTER_0), DTU_CME_CACHE_PF_ACCESS_L1_HIT = OS_DTU_MOD↩
E(DTU_CEV_DLPF, DTU_TRIAD_A, DTU_TRIAD_COUNTER_1), DTU_CME_L1_MISS =
OS_DTU_MODE(DTU_CEV_DLPF, DTU_TRIAD_A, DTU_TRIAD_COUNTER_2),
DTU_DATA_LOAD_CACHEABLE_HOLD_CYCLES = OS_DTU_MODE(DTU_CEV_DLH1,
DTU_TRIAD_A, DTU_TRIAD_COUNTER_0), DTU_DATA_LOAD_NON_CACHEABLE_H↩
OLD_CYCLES = OS_DTU_MODE(DTU_CEV_DLH1, DTU_TRIAD_A, DTU_TRIAD_COU↩

NTER_1), DTU_DATA_CACHE_CONTENTION_CYCLES = OS_DTU_MODE(DTU_CEV_↩
DLH1, DTU_TRIAD_A, DTU_TRIAD_COUNTER_2),
DTU_FULL_FETCH_QUEUE_HOLD_CYCLES = OS_DTU_MODE(DTU_CEV_DLH2, D↩
TU_TRIAD_B, DTU_TRIAD_COUNTER_0), DTU_MISCELLANEOUS_HOLD_CYCLES =
OS_DTU_MODE(DTU_CEV_DLH2, DTU_TRIAD_B, DTU_TRIAD_COUNTER_1), DTU_A↩
DRRESS_QUEUE_LOAD_AFTER_STORE_HAZARD = OS_DTU_MODE(DTU_CEV_DLH2,
DTU_TRIAD_B, DTU_TRIAD_COUNTER_2),
DTU_TOTAL_MEMORY_LOADS = OS_DTU_MODE(DTU_CEV_DLSP, DTU_TRIAD_A,
DTU_TRIAD_COUNTER_0), DTU_NON_ALIGNED_4K_LOAD = OS_DTU_MODE(DTU_↩
CEV_DLSP, DTU_TRIAD_A, DTU_TRIAD_COUNTER_1), DTU_LOAD_REWIND_EVENTS
= OS_DTU_MODE(DTU_CEV_DLSP, DTU_TRIAD_A, DTU_TRIAD_COUNTER_2),
DTU_TOTAL_STORE_ACCESSES = OS_DTU_MODE(DTU_CEV_DSAS, DTU_TRIAD_↩
A, DTU_TRIAD_COUNTER_0), DTU_SGB_MAGES = OS_DTU_MODE(DTU_CEV_DSAS,
DTU_TRIAD_A, DTU_TRIAD_COUNTER_1), DTU_SGB_WRITE_OUT_HOT = OS_DTU_↩
MODE(DTU_CEV_DSAS, DTU_TRIAD_A, DTU_TRIAD_COUNTER_2),
DTU_SGB_FREEZE_CYCLES = OS_DTU_MODE(DTU_CEV_DSGE, DTU_TRIAD_B, DT↩
U_TRIAD_COUNTER_0), DTU_SGB_READ_MODIFY_WRITE_ACCESSES_L2 = OS_DT↩
U_MODE(DTU_CEV_DSGE, DTU_TRIAD_B, DTU_TRIAD_COUNTER_1), DTU_SGB_AC↩
CEPTER_REWIND = OS_DTU_MODE(DTU_CEV_DSGE, DTU_TRIAD_B, DTU_TRIAD_↩
COUNTER_2),
DTU_DLINK_READ_BEATS = OS_DTU_MODE(DTU_CEV_BL12, DTU_TRIAD_A, DTU_↩
TRIAD_COUNTER_0), DTU_DLINK_WRITE_BEATS = OS_DTU_MODE(DTU_CEV_BL12,
DTU_TRIAD_A, DTU_TRIAD_COUNTER_1), DTU_ILINK_BEATS = OS_DTU_MODE(DT↩
U_CEV_BL12, DTU_TRIAD_A, DTU_TRIAD_COUNTER_2),
DTU_TOTAL_L2_DEMAND_ACCESSES = OS_DTU_MODE(DTU_CEV_KED1, DTU_TRI↩
AD_A, DTU_TRIAD_COUNTER_0), DTU_L2_PROGRAM_ACCESSSES = OS_DTU_MOD↩
E(DTU_CEV_KED1, DTU_TRIAD_A, DTU_TRIAD_COUNTER_1), DTU_L2_DATA_ACC↩
ESSES = OS_DTU_MODE(DTU_CEV_KED1, DTU_TRIAD_A, DTU_TRIAD_COUNTER_2),
DTU_TOTAL_L2_HIT = OS_DTU_MODE(DTU_CEV_KED2, DTU_TRIAD_B, DTU_TR↩
IAD_COUNTER_0), DTU_L2_INSTRUCTION_MISS = OS_DTU_MODE(DTU_CEV_KED2,
DTU_TRIAD_B, DTU_TRIAD_COUNTER_1), DTU_L2_DATA_MISS = OS_DTU_MODE(D↩
TU_CEV_KED2, DTU_TRIAD_B, DTU_TRIAD_COUNTER_2),
DTU_TOTAL_L2_AOUT_REQUESTS = OS_DTU_MODE(DTU_CEV_KAOT, DTU_TRI↩
AD_A, DTU_TRIAD_COUNTER_0), DTU_L2_AOUT_REQUESTS_SENT_AS_GLOBAL =
OS_DTU_MODE(DTU_CEV_KAOT, DTU_TRIAD_A, DTU_TRIAD_COUNTER_1), DTU_↩
L2_AOUT_DATA_SIDE_REQUESTS = OS_DTU_MODE(DTU_CEV_KAOT, DTU_TRIAD_↩
A, DTU_TRIAD_COUNTER_2),
DTU_TOTAL_L2_DEMAND_ELINK_ACCESSES = OS_DTU_MODE(DTU_CEV_KTOT, D↩
TU_TRIAD_B, DTU_TRIAD_COUNTER_0), DTU_L2_SNOOP_REQUESTS = OS_DTU_M↩
ODE(DTU_CEV_KTOT, DTU_TRIAD_B, DTU_TRIAD_COUNTER_1), DTU_TOTAL_L2_↩
ACCESSES_FROM_ALL_SOURCES = OS_DTU_MODE(DTU_CEV_KTOT, DTU_TRIAD_B,
DTU_TRIAD_COUNTER_2),
DTU_L2_TOTAL_STASHES = OS_DTU_MODE(DTU_CEV_KEXT, DTU_TRIAD_A, DTU_↩
TRIAD_COUNTER_0), DTU_L2_SNOOP_REQUESTS_EXTERNAL_ACCESSES = OS_DT↩
U_MODE(DTU_CEV_KEXT, DTU_TRIAD_A, DTU_TRIAD_COUNTER_1), DTU_L2_STA↩

SH_REQUESTS_DEGARDED_TO_SNOOP = OS_DTU_MODE(DTU_CEV_KEXT, DTU_T↩ RIAD_A, DTU_TRIAD_COUNTER_2), DTU_L2_RELOAD_REQUESTS_FROM_CORENET = OS_DTU_MODE(DTU_CEV_KDIN, DTU_TRIAD_B, DTU_TRIAD_COUNTER_0), DTU_L2_STORE_ALLOCATE = OS_DTU_↩ MODE(DTU_CEV_KDIN, DTU_TRIAD_B, DTU_TRIAD_COUNTER_1), DTU_L2_SNOOP↩ _HIT = OS_DTU_MODE(DTU_CEV_KNSP, DTU_TRIAD_B, DTU_TRIAD_COUNTER_0), DTU_L2_SNOOP_CAUSING_MINT = OS_DTU_MODE(DTU_CEV_KNSP, DTU_TRIAD_B, DTU_TRIAD_COUNTER_1), DTU_L2_SNOOP_CAUSING_SINT = OS_DTU_MODE(DTU_↩ CEV_KNSP, DTU_TRIAD_B, DTU_TRIAD_COUNTER_2) }

Functions

- uint32_t osDtuReadCount (b486x_dtu_profiling_mode profiling_mode)
- os_status osDtuInitProfiler (b486x_dtu_cev_triad_a cev_triad_a, b486x_dtu_cev_triad_b cev_↩ triad_b)
- void osDtuDisableProfiler ()
- void osDtuStartProfiling ()
- void osDtuStopProfiling ()

2.7.3.2.2.2 Macro Definition Documentation

2.7.3.2.2.2.1 #define NUMBER_OF_CORE_COMMAND_REGISTERS 4

DTU memory map definitions.

number of Core Command Registers 3 through 0

2.7.3.2.2.2.2 #define NUMBER_OF_CORE_COMMAND_DATA_REGISTERS 4

number of core Command Data register

2.7.3.2.2.2.3 #define NUMBER_OF_PROFILING_COUNTER_VALUE_REGISTERS_A 3

number of Profiling Counter Value Register A

2.7.3.2.2.2.4 #define NUMBER_OF_PROFILING_SNAPSHOT_REGISTERS_A 3

number of Profiling Counter Snapshot Registers A

2.7.3.2.2.2.5 #define NUMBER_OF_PROFILING_COUNTER_VALUE_REGISTERS_B 3

number of Profiling Counter Value Register B

2.7.3.2.2.2.6 #define NUMBER_OF_PROFILING_SNAPSHOT_REGISTERS_B 3

number of Profiling Counter Snapshot Registers B

2.7.3.2.2.2.7 #define NUMBER_OF_INDIRECT_EVENT_COND_TRASITION_CONF_REGISTERS 4

number of Indirect Event Conditional Transition Configuration Register

2.7.3.2.2.8 #define OS_DTU_MODE(*cev, triad, counter*) ((cev << 16) | (triad << 8) | counter)

DTU profiling mode - index for DTU modes of operation.

2.7.3.2.2.3 Enumeration Type Documentation

2.7.3.2.2.3.1 enum b486x_dtu_triad

triad - index for DTU triads

Enumerator

DTU_TRIAD_A index for triad A
DTU_TRIAD_B index for triad B

2.7.3.2.2.3.2 enum b486x_dtu_triad_counter

triad counter - index for DTU triads counter

Enumerator

DTU_TRIAD_COUNTER_0 index for triad counter 0
DTU_TRIAD_COUNTER_1 index for triad counter 1
DTU_TRIAD_COUNTER_2 index for triad counter 2

2.7.3.2.2.3.3 enum b486x_dtu_cev_triad_a

DTU CEV triad A value - index for DTU CEV group events.

Enumerator

DTU_TRIAD_A_CEV_NULL NULL.
DTU_CEV_CE1 Exceptions 1.
DTU_CEV_CBP1 Branch prediction 1.
DTU_CEV_CCH Cycles high level.
DTU_CEV_CCS1 Stall cycles 1.
DTU_CEV_ED1 Debug events 1.
DTU_CEV_EW1 Watchpoints 1.
DTU_CEV_EIN1 DEBUGEV instructions 1.
DTU_CEV_IAS L1 Instruction cache access sorting.
DTU_CEV_IPF L1 instruction cache pre-fetch.
DTU_CEV_DGCI Data Coherency - invalidation events.
DTU_CEV_DLSR L1 data cache loadaccess sorting.
DTU_CEV_DLPF L1 data cache pre-fetch.
DTU_CEV_DLH1 Data load holds 1.
DTU_CEV_DLSP Data load special cases.

DTU_CEV_DSAS Data Store Accesses Sorting.
DTU_CEV_BL12 L1-L2 bus load.
DTU_CEV_KED1 L2 demand (Elink) accesses - 1.
DTU_CEV_KAOT L2 AOUT traffic.
DTU_CEV_KEXT L2 external accesses.

2.7.3.2.2.3.4 enum b486x_dtu_cev_triad_b

DTU triad B CEV value - index for DTU CEV group events.

Enumerator

DTU_TRIAD_B_CEV_NULL NULL.
DTU_CEV_CE2 Exceptions 2.
DTU_CEV_CBP2 Branch prediction 2.
DTU_CEV_CLP HW Loop.
DTU_CEV_CRW Rewind events.
DTU_CEV_CCS2 Stall cycles 2.
DTU_CEV_ED2 Debug events 2.
DTU_CEV_IL12 Instruction L1 to L2 cacheable access sorting.
DTU_CEV_IAH Instruction access holds.
DTU_CEV_DGAH Data accesses - general hold/freeze.
DTU_CEV_DL12 Data L1 to L2 cacheable access sorting.
DTU_CEV_DLH2 Data load holds 2.
DTU_CEV_DSGE SGB events.
DTU_CEV_KED2 L2 demand (Elink) accesses - 2.
DTU_CEV_KTOT L2 total traffic.
DTU_CEV_KDIN L2 DIN.
DTU_CEV_KNSP L2 snooping.

2.7.3.2.2.3.5 enum b486x_dtu_profiling_mode

Enumerator

DTU_TRAP_INSTRUCTION Trap instructions.
DTU_CRITICAL_INTERRUPT Critical interrupts.
DTU_NON_CRITICAL_INTERRUPT Non-critical interrupts.
DTU_MMU_EXCEPTIONS MMU exceptions.
DTU_DEBUG_EXCEPTIONS Debug exceptions.
DTU_OTHER_EXCEPTIONS Other exceptions.
DTU_TOTAL_VLES Total VLES.
DTU_COF_VLES COF VLES (w/o hardware loops)
DTU_COF_CORRECTLY_PREDICTED COF correctly predicted (w/o hardware loops)
DTU_ALL_BTB_COF All BTB-table COF.
DTU_BTB_COF_WRONGLY_PREDICTED_NOT_IN_BTB BTB-able COFs, wrongly predicted, not in the BTB.

DTU_BTB_COF_WRONGLY_PREDICTED_IN_BTB BTB-able COFs, wrongly predicted, in the BTB.
DTU_TOTAL_HW_EOL_COF Total hardware end-of-loop COF.
DTU_WRONGLY_PREDICTED_BUFFERED_EOL_COF incorrectly predicted buffered end-of-loop COF
DTU_OTHER_WRONGLY_PREDICTED_EOL_COF Other incorrectly predicted end-of-loop COF (not in the BTB, in the BTB wrongly predicted, loop re-learn)
DTU_APPLICATION_CYCLES Application cycles.
DTU_BUBBLES Bubbles.
DTU_NO_BUBBLES No bubbles (once per VLES)
DTU_COF COF (pipe flush cycles)
DTU_INTERLOCK Interlocks (RSU) w/o holds.
DTU_DATA_MEMORY_HOLDS_AND_FREEZE Data memory holds and freezes.
DTU_PROGRAM_STARVATION Program starvation.
DTU_REWIND_CYCLES Rewind cycles (prog & data)
DTU_DATA_REWIND_EVENTS Data rewind events.
DTU_PROGRAM_REWIND_EVENTS Program rewind events.
DTU_REWIND_EVENTS DTU rewind events.
DTU_IEU_Q0 IEU Q0.
DTU_IEU_Q1 IEU Q1.
DTU_RELOAD_COUNTER_0_RZ Reload counter 0 RZ.
DTU_PCDA0 PCDA0.
DTU_PCDA2 PCDA2.
DTU_DEBUGEV_0 DEBUGEV.0.
DTU_DEBUGEV_2 DEBUGEV.2.
DTU_DEBUGEV_3 DEBUGEV.3.
DTU_PROGRAM_ACCESS_L1_HITS Program access L1 hits.
DTU_PROGRAM_ACCESS_L1_PREFATCH_HITS Program access L1 pre-fetch hits.
DTU_PROGRAM_ACCESS_L1_MISS Program access L1 miss.
DTU_L2_PROGRAM_ACCESS_HITS_SHARED L2 Program access hits (shared/exclusive)
DTU_L2_PROGRAM_ACCESS_HITS_MODIFIED L2 Program access hits (modified)
DTU_L2_PROGRAM_ACCESS_HITS L2 Program access hits.
DTU_CME_PF_ACCESSES_L1_HIT CME/granular PF accesses - L1 hit (including pre-fetch hit)

DTU_CME_PF_ACCESSES_L1_MISS CME granular - L1 miss.
DTU_PROG_CACHEABLE_HOLD_CONTENTION_CYCLES Program cacheable hold + contention cycles.
DTU_PROG_NON_CACHEABLE_HOLDS_MISCELLANEOUS Program non-cacheable holds + miscellaneous.
DTU_PROG_REWIND_EVENTS Program rewind events.
DTU_DATA_CACHE_HOLDS Data cache holds.
DTU_FATA_STORE_FREEZE Data store freeze.
DTU_DATA_REWIND_EVENTS_HL Data hold rewind events.
DTU_L1_DATA_INVALIDATE_FROM_L2 L1 data invalidate from L2.
DTU_CME_INVALIDATION_ACCESSES CME/granular invalidation accesses.

DTU_L1_DATA_INVALIDATE_FROM_INTER_CONFLICTS L1 data invalidation from internal conflicts.
DTU_DATA_ACCESS_L1_HITS Data access L1 hits.
DTU_DATA_ACCESS_L1_PF_HITS Data access L1 pre-fetch hits.
DTU_DATA_ACCESS_L1_MISS Data access L1 miss.
DTU_L2_DATA_ACCESS_HITS_SHARED L2 Data access hits (shared/exclusive)
DTU_L2_DATA_ACCESS_HITS_MODIFIED L2 Data access hits (modified)
DTU_L2_DATA_ACCESS_MISS L2 Data access miss.
DTU_CME_PF_ACCESS_L1_HIT CME/granular PF accesses - L1 hit (including pre-fetch hit)
DTU_CME_CACHE_PF_ACCESS_L1_HIT CME/granular cache pre-fetch accesses that were dropped.
DTU_CME_L1_MISS CME granular - L1 miss.
DTU_DATA_LOAD_CACHEABLE_HOLD_CYCLES Data load cacheable hold cycles.
DTU_DATA_LOAD_NON_CACHEABLE_HOLD_CYCLES Data loads non-cacheable hold cycles.
DTU_DATA_CACHE_CONTENTION_CYCLES Data cache contention cycles.
DTU_FULL_FETCH_QUEUE_HOLD_CYCLES Full fetch queue hold cycles.
DTU_MISCELLANEOUS_HOLD_CYCLES miscellaneous hold cycles (barriers, special commands)
DTU_ADRRESS_QUEUE_LOAD_AFTER_STORE_HAZARD Address queue load after store hazard cycles.
DTU_TOTAL_MEMORY_LOADS Total memory loads.
DTU_NON_ALIGNED_4K_LOAD Non-aligned 4K load.
DTU_LOAD_REWIND_EVENTS load rewind events
DTU_TOTAL_STORE_ACCESSES Total store accesses.
DTU_SGB_MAGES SGB merges.
DTU_SGB_WRITE_OUT_HOT SGB write out hot (DLINK to L1)
DTU_SGB_FREEZE_CYCLES SGB freeze cycles.
DTU_SGB_READ_MODIFY_WRITE_ACCESSES_L2 SGB read-modify-write accesses to L2.
DTU_SGB_ACCEPTER_REWIND SGB accepter rewind events.
DTU_DLINK_READ_BEATS DLINK read beats.
DTU_DLINK_WRITE_BEATS DLINK write beats.
DTU_ILINK_BEATS ILINK beats.
DTU_TOTAL_L2_DEMAND_ACCESSES Total L2 demand accesses.
DTU_L2_PROGRAM_ACCESSSES L2 program accesses.
DTU_L2_DATA_ACCESSES L2 data accesses.
DTU_TOTAL_L2_HIT Total L2 hit.
DTU_L2_INSTRUCTION_MISS L2 instruction miss.
DTU_L2_DATA_MISS L2 data miss.
DTU_TOTAL_L2_AOUT_REQUESTS Total L2 AOUT requests.
DTU_L2_AOUT_REQUESTS_SENT_AS_GLOBAL L2 AOUT requests sent as global (M=1)
DTU_L2_AOUT_DATA_SIDE_REQUESTS L2 AOUT data side requests (including barriers)
DTU_TOTAL_L2_DEMAND_ELINK_ACCESSES Total L2 demand (ELINK) accesses.
DTU_L2_SNOOP_REQUESTS L2 Snoop requests.
DTU_TOTAL_L2_ACCESSES_FROM_ALL_SOURCES Tot. L2 accesses from all sources

DTU_L2_TOTAL_STASHES L2 Total stashes.
DTU_L2_SNOOP_REQUESTS_EXTERNAL_ACCESSES L2 Snoop requests external accesses.
DTU_L2_STASH_REQUESTS_DEGARDED_TO_SNOOP L2 Stash requests degraded to snoop.

DTU_L2_RELOAD_REQUESTS_FROM_CORENET L2 reload requests from CoreNet.
DTU_L2_STORE_ALLOCATE L2 Store allocate.
DTU_L2_SNOOP_HIT L2 Snoop hit.
DTU_L2_SNOOP_CAUSING_MINT L2 snoops causing MINT.
DTU_L2_SNOOP_CAUSING_SINT L2 snoops causing SINT.

2.7.3.2.2.4 Function Documentation

2.7.3.2.2.4.1 uint32_t osDtuReadCount (b486x_dtu_profiling_mode *profiling_mode*)

Reads counter value

Parameters

in	*profiling_mode*	- gives an indication for which counter hold the benchmark data.

Returns

The counter value.

2.7.3.2.2.4.2 os_status osDtuInitProfiler (b486x_dtu_cev_triad_a *cev_triad_a*, b486x_dtu_cev_triad_b *cev_triad_b*)

initialized the triads counters.

Parameters

in	*cev_triad_a*	- index for DTU CEV group events to trigger the triad A counters.
in	*cev_triad_b*	- index for DTU CEV group events to trigger the triad B counters.

Returns

OS_SUCCESS.

2.7.3.2.2.4.3 void osDtuDisableProfiler ( )

Disable the triads counters.

2.7.3.2.2.4.4 void osDtuStartProfiling ( )

Triggers DTU counters to start counting.

2.7.3.2.2.4.5 void osDtuStopProfiling ( )

Triggers DTU counters to stop the count.

2.7.3.2.3 B4860 Debug Print API

2.7.3.2.3.1 Overview

Modules

- B4860 Debug Print Initialization API
- B4860 Debug Print runtime API

2.7.3.2.3.2 B4860 Debug Print Initialization API

2.7.3.2.3.2.1 Overview

```
Debug Print setup.
```

Data Structures

- struct b486x_debug_print_init_params_t

Enumerations

- enum vtb_size_t
- enum vtb_segments_t

Functions

- os_status b486xDebugPrintInitialize (b486x_debug_print_init_params_t *init_params)

2.7.3.2.3.2.2 Data Structure Documentation

2.7.3.2.3.2.3 struct b486x_debug_print_init_params_t

Debug print initialization parameters.

Data Fields

- os_phys_ptr dcsr_phys_addr
- os_virt_ptr dcsr_virt_addr
- os_virt_ptr vtb_start_addr
- os_mem_type vtb_mem_location

- vtb_size_t vtb_size
- vtb_segments_t num_of_segments
- uint32_t flags
- os_virt_ptr soc_het_ctrl
- bool mngmnt_cores [OS_SOC_MAX_NUM_OF_CORES]

2.7.3.2.3.2.4 Field Documentation

2.7.3.2.3.2.5 os_phys_ptr b486x_debug_print_init_params_t::dcsr_phys_addr

DCSR Physical address, shall reflect the definition in LAW.

2.7.3.2.3.2.6 os_virt_ptr b486x_debug_print_init_params_t::dcsr_virt_addr

DCSR virtual address.

2.7.3.2.3.2.7 os_virt_ptr b486x_debug_print_init_params_t::vtb_start_addr

VTB virtual address, if NULL the driver will allocate it.

2.7.3.2.3.2.8 os_mem_type b486x_debug_print_init_params_t::vtb_mem_location

Memory where VTBs are allocated.

2.7.3.2.3.2.9 vtb_size_t b486x_debug_print_init_params_t::vtb_size

Log2 of the VTB size to be used for Debug Prints (VTB_size = 1 << vtb_size)

2.7.3.2.3.2.10 vtb_segments_t b486x_debug_print_init_params_t::num_of_segments

Log2 of the VTB number of segments.

2.7.3.2.3.2.11 uint32_t b486x_debug_print_init_params_t::flags

If any core is disabled.

2.7.3.2.3.2.12 os_virt_ptr b486x_debug_print_init_params_t::soc_het_ctrl

virtual pointer to os_het_control_t shared space

2.7.3.2.3.2.13 bool b486x_debug_print_init_params_t::mngmnt_cores[OS_SOC_MAX_NUM_↩ OF_CORES]

This determines which core is allowed to react to debug print watermark interrupts.

2.7.3.2.3.2.14 Enumeration Type Documentation

2.7.3.2.3.2.15 enum vtb_size_t

Virtual Trace Buffer Size.

2.7.3.2.3.2.16 enum vtb_segments_t

Virtual Trace Buffer - Number of segments.

2.7.3.2.3.2.17 Function Documentation

2.7.3.2.3.2.18 os_status b486xDebugPrintInitialize (b486x_debug_print_init_params_t * *init_params*)

Configures QE for basic ETH access

Parameters

init_params	- debug print initialization params

Returns

OS_SUCCESS if valid

Warning

New Mode: None Old Mode: After calling this function: o interrupts are disabled on the UEC used for channels 0,NUM_OF_CORES o UTFTT register in UEC memory map is modified

2.7.3.2.3.3 B4860 Debug Print runtime API

2.7.3.2.3.3.1 Overview

`Debug Print functions.`

Enumerations

- enum b486x_debug_print_trigger_level_t

Functions

- os_status b486xDebugPrintFast (uint32_t trigger_level, uint16_t local_address, uint32_t data)
- os_status b486xDebugPrintMass (uint32_t trigger_level, uint16_t local_address, uint32_t *data, uint16_t address_increment_offset, uint16_t count)

2.7.3.2.3.3.2 Enumeration Type Documentation

2.7.3.2.3.3.3 enum b486x_debug_print_trigger_level_t

Debug Print Trigger Level Enumeration.

2.7.3.2.3.3.4 Function Documentation

2.7.3.2.3.3.5 os_status b486xDebugPrintFast (uint32_t *trigger_level*, uint16_t *local_address*, uint32_t *data*)

Writes a single DP into the VTB buffer

Parameters

in	*trigger_level*	- DP trigger level, left-shifted with 16 bits
in	*local_address*	- DP address
in	*data*	- DP value

Warning

If function being called in a HWI or SWI ISR, the application MUST evaluate msc815xDebug↩ PrintRunning() and validate it returns FALSE prior to calling.

Returns

OS_SUCCESS

2.7.3.2.3.3.6 os_status b486xDebugPrintMass (uint32_t *trigger_level*, uint16_t *local_address*, uint32_t * *data*, uint16_t *address_increment_offset*, uint16_t *count*)

Writes a mass DP into the VTB buffer

Parameters

in	*trigger_level*	- DP trigger level
in	*local_address*	- DP address
in	*data*	- pointer to the start address of the data to be included in MDP
in	*address_↩ increment_↩ offset*	- DP address increment offset
in	*count*	- number of simple DPs contained in the mass DP

Return values

OS_SUCCESS	- MDP was successfully inserted
OS_ERR_DEBUG_PRI↩ NT_NOT_ACTIVE	- QE is not in ACTIVE state
OS_ERR_DEBUG_PRI↩ NT_LENGTH_INVALID	- More than 2048 WORDS were attempted to be written in an MDP
OS_ERR_DEBUG_PRI↩ NT_MDP_ACTIVE	- Another MDP is currently being written
OS_ERR_DEBUG_PRI↩ NT_OVERFLOW	- An overflow would occur if the MDP would be written

Warning

Interrupts must be enabled when calling this function

2.7.3.3 Logging API

2.7.3.3.1 Overview

```
Debug hook API.

Logging API
```

Debug Hooks are callback functions that are called by the OS runtime and initialization code at predetermined locations in the code.

Modules

- Logging runtime API

Macros

- #define osDebugHookDelete(hook_num) osDebugHookCreate(hook_num, NULL)

Functions

- os_status osDebugHookCreate (os_debug_hooks_enum hook_num, debug_hook_function func)

Debug Hook Options

- enum os_debug_hooks_enum {
  OS_DEBUG_TASK_CREATE, OS_DEBUG_TASK_SWITCH, OS_DEBUG_PROG_MMU,
  OS_DEBUG_DATA_MMU, OS_DEBUG_IPC_BASIC_SEND, OS_DEBUG_IPC_BASIC_RE↩

CEIVE,
OS_DEBUG_LAST }

2.7.3.3.2 Macro Definition Documentation

2.7.3.3.2.1 #define osDebugHookDelete(*hook_num*) osDebugHookCreate(hook_num, NULL)

Unregister a debug hook callback function

Parameters

in	*hook_num*	- The event to unregister from - Must be one of Debug Hook Options

Return values

OS_SUCCESS	- the hook was unregistered successfully
OS_ERR_TOO_MUCH	- hook_num is larger than OS_DEBUG_LAST

2.7.3.3.3 Enumeration Type Documentation

2.7.3.3.3.1 enum os_debug_hooks_enum

Debug Hooks configuration options.

Enumerator

OS_DEBUG_TASK_CREATE Called when task is created. Arguments are: os_task_handle and pointer to os_task_init_param_t

OS_DEBUG_TASK_SWITCH Called when there is a task switch. Arguments are: g_running_task->task_handle and g_highest_task->task_handle

OS_DEBUG_PROG_MMU Called on program MMU exception. Arguments are: pointer to os_sys_info_t and pointer to struct os_mmu_error

OS_DEBUG_DATA_MMU Called on data MMU exception. Arguments are: pointer to os_sys_info_t and pointer to struct os_mmu_error

OS_DEBUG_IPC_BASIC_SEND Called on IPC message send Arguments are: pointer to os_het_ipc_channel_t, pointer to local copy of current BD (message address in this BD is physical); Caution: this local copy is not valid after exiting the debug hook.

OS_DEBUG_IPC_BASIC_RECEIVE Called on IPC message receive Arguments are: pointer to os_het_ipc_channel_t, pointer to local copy of current BD (message address in this BD is physical); Caution: this local copy is not valid after exiting the debug hook.

OS_DEBUG_LAST Number of supported debug hooks - MUST reain last.

2.7.3.3.4 Function Documentation

2.7.3.3.4.1 os_status osDebugHookCreate (os_debug_hooks_enum *hook_num,* debug_hook_function *func*)

Register a debug hook callback function

Parameters

in	*hook_num*	- The event to register to - Must be one of Debug Hook Options
in	*func*	- The callback function to register.

Return values

OS_SUCCESS	- the hook was registered successfully
OS_ERR_TOO_MUCH	- hook_num is larger than OS_DEBUG_LAST
OS_ERR_ALREADY_CREATED	- the hook was previously registered

2.7.3.3.5 Logging runtime API

2.7.3.3.5.1 Overview

Macros

- #define LOG_ENTER_EVENT 0x01000000
- #define LOG_EXIT_EVENT 0x02000000
- #define GET_FUNCTION_CODE(num) (num>>12)

Functions

- void osLog (uint32_t op, uint32_t val)
- void osLogRegisterHandler (uint32_t type_of_command, void(*handler)(uint32_t op, uint32_t val, uint32_t time[2]))
- void osLogImmediate (uint32_t op, uint32_t val, uint32_t time[2])

Types of opcodes

- #define OS_LOG_HWI 0x1000
- #define OS_LOG_SWI 0x2000
- #define OS_LOG_SPINLOCK 0x3000
- #define **OS_LOG_HWI_LATENCY** 0x4000
- #define OS_LOG_TASK 0x5000
- #define OS_LOG_CLASS_UTILIZATION 0x6000
- #define OS_LOG_USER_DEFINED_EVENT 0x8000
- #define LOG_OPCODE_TO_STACK 0x01000000
- #define LOG_OPCODE_FROM_STACK 0x02000000
- #define LOG_COMMAND_TYPE_MASK 0xFF00

Opcodes supported for osLog

- #define LOG_HWI_ENTER (0x1 | OS_LOG_HWI | LOG_OPCODE_TO_STACK)
- #define LOG_HWI_EXIT (0x2 | OS_LOG_HWI | LOG_OPCODE_FROM_STACK)
- #define LOG_HWI_CREATE (0x3 | OS_LOG_HWI)
- #define LOG_HWI_SET_PRIORITY (0x4 | OS_LOG_HWI)
- #define LOG_SWI_CREATE (0x1 | OS_LOG_SWI)
- #define LOG_SWI_ENTER (0x2 | OS_LOG_SWI | LOG_OPCODE_TO_STACK)
- #define LOG_SWI_EXIT (0x3 | OS_LOG_SWI | LOG_OPCODE_FROM_STACK)
- #define LOG_SWI_ACTIVATE (0x4 | OS_LOG_SWI)
- #define LOG_SWI_SET_PRIORITY (0x5 | OS_LOG_SWI)
- #define LOG_SPINLOCK_GET (0x1 | OS_LOG_SPINLOCK)
- #define LOG_SPINLOCK_RELEASE (0x2 | OS_LOG_SPINLOCK)
- #define LOG_SPINLOCK_IRQ_GET (0x3 | OS_LOG_SPINLOCK)
- #define LOG_SPINLOCK_IRQ_RELEASE (0x4 | OS_LOG_SPINLOCK)
- #define LOG_CLASS_M2_UTILIZATION (0x1 | OS_LOG_CLASS_UTILIZATION)
- #define LOG_CLASS_M3_UTILIZATION (0x2 | OS_LOG_CLASS_UTILIZATION)
- #define LOG_CLASS_DDR_UTILIZATION (0x3 | OS_LOG_CLASS_UTILIZATION)
- #define LOG_CLASS_SKYBLUE_UTILIZATION (0x4 | OS_LOG_CLASS_UTILIZATION)
- #define LOG_CLASS_SYSTEM_TO_M2_UTILIZATION (0x5 | OS_LOG_CLASS_UTILIZA↩ TION)
- #define LOG_CLASS_CORES_TO_M3_DDR_UTILIZATION (0x6 | OS_LOG_CLASS_UTIL↩ IZATION)
- #define LOG_CLASS_QE_UTILIZATION (0x7 | OS_LOG_CLASS_UTILIZATION)
- #define LOG_CLASS_SRIO_UTILIZATION (0x8 | OS_LOG_CLASS_UTILIZATION)
- #define LOG_CLASS_TDM_UTILIZATION (0x9 | OS_LOG_CLASS_UTILIZATION)
- #define LOG_CLASS_DMA_PORT0_UTILIZATION (0xA | OS_LOG_CLASS_UTILIZATION)
- #define LOG_CLASS_DMA_PORT1_UTILIZATION (0xB | OS_LOG_CLASS_UTILIZATION)

Opcodes not supported for osLog

- #define LOG_HWI_ENABLE (0x1 | OS_LOG_HWI_LATENCY)
- #define LOG_HWI_DISABLE (0x2 | OS_LOG_HWI_LATENCY)
- #define LOG_SWIFT_HWI_DISABLE (0x3 | OS_LOG_HWI_LATENCY)
- #define LOG_SWIFT_HWI_ENABLE (0x4 | OS_LOG_HWI_LATENCY)
- #define LOG_TASK_CREATE (0x1 | OS_LOG_TASK)
- #define LOG_TASK_SWITCH (0x2 | OS_LOG_TASK)
- #define LOG_TASK_ACTIVATE (0x3 | OS_LOG_TASK)
- #define LOG_TASK_SET_PRIORITY (0x4 | OS_LOG_TASK)
- #define LOG_TASK_PEND_QUEUE (0x5 | OS_LOG_TASK)
- #define LOG_TASK_PEND_SEM (0x6 | OS_LOG_TASK)
- #define LOG_TASK_DELETE (0x7 | OS_LOG_TASK)
- #define LOG_TASK_POST_QUEUE (0x8 | OS_LOG_TASK)
- #define LOG_TASK_POST_SEM (0x9 | OS_LOG_TASK)
- #define LOG_TASK_SUSPEND (0xA | OS_LOG_TASK)
- #define LOG_TASK_DELAY (0xB | OS_LOG_TASK)
- #define LOG_TASK_YIELD (0xC | OS_LOG_TASK)

2.7.3.3.5.2 Macro Definition Documentation

2.7.3.3.5.2.1 #define LOG_ENTER_EVENT 0x01000000

start flag for opening LOG event

2.7.3.3.5.2.2 #define LOG_EXIT_EVENT 0x02000000

ending flag for LOG event

2.7.3.3.5.2.3 #define OS_LOG_HWI 0x1000

HWI events.

2.7.3.3.5.2.4 #define OS_LOG_SWI 0x2000

SWI events.

2.7.3.3.5.2.5 #define OS_LOG_SPINLOCK 0x3000

Spinlock events.

2.7.3.3.5.2.6 #define OS_LOG_TASK 0x5000

Task events.

2.7.3.3.5.2.7 #define OS_LOG_CLASS_UTILIZATION 0x6000

CLASS profiling.

2.7.3.3.5.2.8 #define OS_LOG_USER_DEFINED_EVENT 0x8000

User-defined events.

2.7.3.3.5.2.9 #define LOG_OPCODE_TO_STACK 0x01000000

opcode to stack

2.7.3.3.5.2.10 #define LOG_OPCODE_FROM_STACK 0x02000000

opcode from stack

2.7.3.3.5.2.11 #define LOG_COMMAND_TYPE_MASK 0xFF00

opcode type log mask

2.7.3.3.5.2.12 #define LOG_HWI_ENTER (0x1 | OS_LOG_HWI | LOG_OPCODE_TO_STACK)

HWI enter event.

2.7.3.3.5.2.13 #define LOG_HWI_EXIT (0x2 | OS_LOG_HWI | LOG_OPCODE_FROM_STACK)

HWI exit event.

2.7.3.3.5.2.14 #define LOG_HWI_CREATE (0x3 | OS_LOG_HWI)

HWI create event.

2.7.3.3.5.2.15 #define LOG_HWI_SET_PRIORITY (0x4 | OS_LOG_HWI)

HWI priority set event.

2.7.3.3.5.2.16 #define LOG_SWI_CREATE (0x1 | OS_LOG_SWI)

SWI create event.

2.7.3.3.5.2.17 #define LOG_SWI_ENTER (0x2 | OS_LOG_SWI | LOG_OPCODE_TO_STACK)

SWI enter event.

2.7.3.3.5.2.18 #define LOG_SWI_EXIT (0x3 | OS_LOG_SWI | LOG_OPCODE_FROM_STACK)

SWI exit event.

2.7.3.3.5.2.19 #define LOG_SWI_ACTIVATE (0x4 | OS_LOG_SWI)

SWI activate event.

2.7.3.3.5.2.20 #define LOG_SWI_SET_PRIORITY (0x5 | OS_LOG_SWI)

SWI priority set event.

2.7.3.3.5.2.21 #define LOG_SPINLOCK_GET (0x1 | OS_LOG_SPINLOCK)

Spinlock get event.

2.7.3.3.5.2.22 #define LOG_SPINLOCK_RELEASE (0x2 | OS_LOG_SPINLOCK)

Spinlock release event.

2.7.3.3.5.2.23 #define LOG_SPINLOCK_IRQ_GET (0x3 | OS_LOG_SPINLOCK)

Spinlock irq disable get event.

2.7.3.3.5.2.24 #define LOG_SPINLOCK_IRQ_RELEASE (0x4 | OS_LOG_SPINLOCK)

Spinlock irq disable release event.

2.7.3.3.5.2.25 #define LOG_CLASS_M2_UTILIZATION (0x1 | OS_LOG_CLASS_UTILIZATION)

class M2 utilization

2.7.3.3.5.2.26 #define LOG_CLASS_M3_UTILIZATION (0x2 | OS_LOG_CLASS_UTILIZATION)

class M3 utilization

2.7.3.3.5.2.27 #define LOG_CLASS_DDR_UTILIZATION (0x3 | OS_LOG_CLASS_UTILIZATION)

class DDR utilization

2.7.3.3.5.2.28 #define LOG_CLASS_SKYBLUE_UTILIZATION (0x4 | OS_LOG_CLASS_UTILIZATION)

class SKYBLUE utilization

2.7.3.3.5.2.29 #define LOG_CLASS_SYSTEM_TO_M2_UTILIZATION (0x5 | OS_LOG_CLASS_UTILIZATION)

class system to M2 utilization

2.7.3.3.5.2.30 #define LOG_CLASS_CORES_TO_M3_DDR_UTILIZATION (0x6 | OS_LOG_CLASS_UTILIZATION)

class cores to M3 utilization

2.7.3.3.5.2.31 #define LOG_CLASS_QE_UTILIZATION (0x7 | OS_LOG_CLASS_UTILIZATION)

class QE utilization

2.7.3.3.5.2.32 #define LOG_CLASS_SRIO_UTILIZATION (0x8 | OS_LOG_CLASS_UTILIZATION)

class SRIO utilization

2.7.3.3.5.2.33 #define LOG_CLASS_TDM_UTILIZATION (0x9 | OS_LOG_CLASS_UTILIZATION)

class TDM utilization

2.7.3.3.5.2.34 #define LOG_CLASS_DMA_PORT0_UTILIZATION (0xA | OS_LOG_CLASS_UTILIZATION)

class DMA port 0 utilization

2.7.3.3.5.2.35 #define LOG_CLASS_DMA_PORT1_UTILIZATION (0xB | OS_LOG_CLASS_UTILIZATION)

class DMA port 1 utilization

2.7.3.3.5.2.36 #define LOG_HWI_ENABLE (0x1 | OS_LOG_HWI_LATENCY)

harware interrupt enable

2.7.3.3.5.2.37 #define LOG_HWI_DISABLE (0x2 | OS_LOG_HWI_LATENCY)

harware interrupt disable

2.7.3.3.5.2.38 #define LOG_SWIFT_HWI_DISABLE (0x3 | OS_LOG_HWI_LATENCY)

swift harware interrupt enable

2.7.3.3.5.2.39 #define LOG_SWIFT_HWI_ENABLE (0x4 | OS_LOG_HWI_LATENCY)

swift harware interrupt disable

2.7.3.3.5.2.40 #define LOG_TASK_CREATE (0x1 | OS_LOG_TASK)

TASK create event.

2.7.3.3.5.2.41 #define LOG_TASK_SWITCH (0x2 | OS_LOG_TASK)

TASK switch event.

2.7.3.3.5.2.42 #define LOG_TASK_ACTIVATE (0x3 | OS_LOG_TASK)

TASK activate event.

2.7.3.3.5.2.43 #define LOG_TASK_SET_PRIORITY (0x4 | OS_LOG_TASK)

TASK priority set event.

2.7.3.3.5.2.44 #define LOG_TASK_PEND_QUEUE (0x5 | OS_LOG_TASK)

TASK queue pend.

2.7.3.3.5.2.45 #define LOG_TASK_PEND_SEM (0x6 | OS_LOG_TASK)

TASK sem pend.

2.7.3.3.5.2.46 #define LOG_TASK_DELETE (0x7 | OS_LOG_TASK)

TASK switch event.

2.7.3.3.5.2.47 #define LOG_TASK_POST_QUEUE (0x8 | OS_LOG_TASK)

TASK queue post.

2.7.3.3.5.2.48 #define LOG_TASK_POST_SEM (0x9 | OS_LOG_TASK)

TASK sem pend.

2.7.3.3.5.2.49 #define LOG_TASK_SUSPEND (0xA | OS_LOG_TASK)

TASK suspend.

2.7.3.3.5.2.50 #define LOG_TASK_DELAY (0xB | OS_LOG_TASK)

TASK delay.

2.7.3.3.5.2.51 #define LOG_TASK_YIELD (0xC | OS_LOG_TASK)

TASK yield.

2.7.3.3.5.2.52 #define GET_FUNCTION_CODE(*num*) (num>>12)

get function code from log

2.7.3.3.5.3 Function Documentation

2.7.3.3.5.3.1 void osLog (uint32_t *op*, uint32_t *val*)

Log events to the internal logging module.

Parameters

in	*op*	- Event opcode. Opcodes supported: • define LOG_HWI_ENTER (0x1 | OS_LOG_HWI | LOG_OPC↩ ODE_TO_STACK) • define LOG_HWI_EXIT (0x2 | OS_LOG_HWI | LOG_OPCO↩ DE_FROM_STACK) • define LOG_HWI_CREATE (0x3 | OS_LOG_HWI) • define LOG_HWI_SET_PRIORITY (0x4 | OS_LOG_HWI) • define LOG_SWI_CREATE (0x1 | OS_LOG_SWI) • define LOG_SWI_ENTER (0x2 | OS_LOG_SWI | LOG_OPC↩ ODE_TO_STACK) • define LOG_SWI_EXIT (0x3 | OS_LOG_SWI | LOG_OPCO↩ DE_FROM_STACK) • define LOG_SWI_ACTIVATE (0x4 | OS_LOG_SWI) • define LOG_SWI_SET_PRIORITY (0x5 | OS_LOG_SWI) • define LOG_SPINLOCK_GET (0x1 | OS_LOG_SPINLOCK) • define LOG_SPINLOCK_RELEASE (0x2 | OS_LOG_SPINL↩ OCK) • define LOG_SPINLOCK_IRQ_GET (0x3 | OS_LOG_SPINLO↩ CK) • define LOG_SPINLOCK_IRQ_RELEASE (0x4 | OS_LOG_SP↩ INLOCK) • define LOG_CLASS_TRGT_NUM(LOG_CLASS,TRGT_NU↩ M) • define LOG_CLASS_INIT_NUM(LOG_CLASS,INIT_NUM)

in	*val*	- Event-related value; This value is specific for each type of event.

2.7.3.3.5.3.2 void osLogRegisterHandler (uint32_t *type_of_command,* void(∗)(uint32_t op, uint32_t val, uint32_t time[2]) *handler*)

Define user callback function for os events.

This function sets callback for specific event. User may use it in order to handle all the events of this type. If the user wish also to log these events he should call osLogImmediate() function for each event with proper arguments.

Parameters

in	*type_of_↩ command*	- Event opcode. Supported values: • OS_LOG_HWI • OS_LOG_SWI • OS_LOG_SPINLOCK
in	*handler*	- pointer to user callback, that will be called on each event.

2.7.3.3.5.3.3 void osLogImmediate (uint32_t *op,* uint32_t *val,* uint32_t *time[2]*)

The same as osLog(), but does not call user-defined callback and does not check if this event was enabled.

Usally called by user in event handler function to enable logging of this event.

Parameters

in	*op*	- Event opcode.
in	*val*	- Event-related value. This value is specific for each type of event.
in	*time*	- Time value for log.

2.8 Queues API

2.8.1 Overview

Queues setup and control.

Typedefs

- typedef void ∗ os_queue_handle

Enumerations

- enum os_queue_status { OS_QUEUE_EMPTY = 0, OS_QUEUE_MIDDLE, OS_QUEUE_FULL }
- enum os_queue_shared { OS_QUEUE_PRIVATE = 0, OS_QUEUE_SHARED }

Functions

- os_status osQueueCreate (os_queue_handle que_handle, uint16_t queue_size)
- os_status osQueueCreateMultiple (os_queue_handle que_handle, uint16_t queue_size, uint16_t num)
- os_status osQueueDelete (os_queue_handle que_handle)
- os_status osQueueDeleteUnsafe (os_queue_handle que_handle)
- os_status osQueueReset (os_queue_handle que_handle)
- os_status osQueueResetUnsafe (os_queue_handle que_handle)
- os_status osQueueEnqueue (os_queue_handle que_handle, uint32_t value)
- os_status osQueueUnsafeEnqueue (os_queue_handle que_handle, uint32_t value)
- os_status osQueueHeadEnqueue (os_queue_handle que_handle, uint32_t value)
- os_status osQueueUnsafeHeadEnqueue (os_queue_handle que_handle, uint32_t value)
- os_status osQueueDequeue (os_queue_handle que_handle, uint32_t *value)
- os_status osQueueUnsafeDequeue (os_queue_handle que_handle, uint32_t *value)
- os_status osQueuePeek (os_queue_handle que_handle, uint32_t *value)
- os_status osQueueTailDequeue (os_queue_handle que_handle, uint32_t *value)
- os_status osQueueUnsafeTailDequeue (os_queue_handle que_handle, uint32_t *value)
- os_queue_status osQueueStatus (os_queue_handle que_handle)
- os_queue_status osQueueUnsafeStatus (os_queue_handle que_handle)
- os_status osQueueFind (os_queue_handle *que_handle, bool shared)
- os_status osQueueEnqueueMultiple (os_queue_handle que_handle, uint32_t *value)
- os_status osQueueUnsafeEnqueueMultiple (os_queue_handle que_handle, uint32_t *value)
- os_status osQueueDequeueMultiple (os_queue_handle que_handle, uint32_t *value)
- os_status osQueueUnsafeDequeueMultiple (os_queue_handle que_handle, uint32_t *value)
- os_queue_shared osQueueShared (os_queue_handle que_handle)
- os_status osQueueSize (os_queue_handle que_handle, uint32_t *size)
- os_status osQueueMultipleSize (os_queue_handle que_handle, uint32_t *size)
- os_status osQueueMultipleNum (os_queue_handle que_handle, uint32_t *num)

2.8.2 Typedef Documentation

2.8.2.1 typedef void* os_queue_handle

Queue handle - represents a specific queue object.

The queue handle is returned from osQueueFind(), and should be passed as a parameter to all the queue functions.

2.8.3 Enumeration Type Documentation

2.8.3.1 enum os_queue_status

Queue status values.

Enumerator

OS_QUEUE_EMPTY Queue is empty.
OS_QUEUE_MIDDLE Queue is neither empty, nor full.
OS_QUEUE_FULL Queue is full.

2.8.3.2 enum os_queue_shared

Queue shared values.

Enumerator

OS_QUEUE_PRIVATE Queue is private, interfaced by a single core.
OS_QUEUE_SHARED Queue is shared, may be interfaced by multiple cores.

2.8.4 Function Documentation

2.8.4.1 os_status osQueueCreate (os_queue_handle *que_handle,* uint16_t *queue_size*)

Prepares a queue for operation.

Parameters

in	*que_handle*	- The queue handle.
in	*queue_size*	- Maximum number of elements allowed in the queue.

Return values

OS_SUCCESS	- The queue was successfully created.
OS_ERR_NO_MEMORY	- Not enough memory.
OS_ERR_Q_ALREADY_CREATED	- The queue is already created.

Warning

This function allocates some memory for the queue operation. You should use it only in your application's initialization code. Creating a specific queue twice will cause memory leaks.

2.8.4.2 os_status osQueueCreateMultiple (os_queue_handle *que_handle,* uint16_t *queue_size,* uint16_t *num*)

Prepares a multiple queue for operation.

Parameters

in	*que_handle*	- The queue handle.
in	*queue_size*	- Maximum number of elements allowed in the queue.
in	*num*	- Size of each element in 32 bits.

Return values

OS_SUCCESS	- The queue was successfully created.
OS_ERR_NO_MEMORY	- Not enough memory.
OS_ERR_Q_ALREADY_CREATED	- The queue is already created.

Warning

This function allocates some memory for the queue operation. You should use it only in your application's initialization code. Creating a specific queue twice will cause memory leaks.

2.8.4.3 os_status osQueueDelete (os_queue_handle *que_handle*)

Deletes the given queue from the system.

Parameters

in	*que_handle*	- The queue handle.

Return values

OS_SUCCESS	- The queue was successfully deleted.
OS_ERR_Q_NOT_CREATED	- The queue was not created, or it was already deleted.

2.8.4.4 os_status osQueueDeleteUnsafe (os_queue_handle *que_handle*)

Deletes the given queue from the system (unsafe version).

Parameters

in	*que_handle*	- The queue handle.

Return values

OS_SUCCESS	- The queue was successfully deleted.
OS_ERR_Q_NOT_CREATED	- The queue was not created, or it was already deleted.

Warning

Use this function when you can be sure that the queue object cannot be accessed from any other source.

2.8.4.5 os_status osQueueReset (os_queue_handle *que_handle*)

empty the given queue.

Parameters

in	*que_handle*	- The queue handle.

Return values

OS_SUCCESS	- The queue was successfully reset.

2.8.4.6 os_status osQueueResetUnsafe (os_queue_handle *que_handle*)

empty the given queue (unsafe version).

Parameters

in	*que_handle*	- The queue handle.

Return values

OS_SUCCESS	- The queue was successfully reset.

Warning

Use this function when you can be sure that the queue object cannot be accessed from any other source.

2.8.4.7 os_status osQueueEnqueue (os_queue_handle *que_handle,* uint32_t *value*)

Inserts a value to the queue.

Parameters

in	*que_handle*	- The queue handle.
in	*value*	- The value to insert to the queue.

Return values

OS_SUCCESS	- The value was successfully inserted.
OS_ERR_Q_FULL	- The queue is full; operation failed.
OS_ERR_Q_NOT_CRE↩ ATED	- The queue was not created, or it was already deleted.

2.8.4.8 os_status osQueueUnsafeEnqueue (os_queue_handle *que_handle,* uint32_t *value*)

Inserts a value to the queue (unsafe version).

Parameters

in	*que_handle*	- The queue handle.
in	*value*	- The value to insert to the queue.

Return values

OS_SUCCESS	- The value was successfully inserted.
OS_ERR_Q_FULL	- The queue is full; operation failed.
OS_ERR_Q_NOT_CRE↩ ATED	- The queue was not created, or it was already deleted.

Warning

Use this function when you can be sure that the queue object cannot be accessed from any other source.

2.8.4.9 os_status osQueueHeadEnqueue (os_queue_handle *que_handle,* uint32_t *value*)

Inserts a value to the front of queue.

Parameters

in	*que_handle*	- The queue handle.
in	*value*	- The value to insert to the queue.

Return values

OS_SUCCESS	- The value was successfully inserted.
OS_ERR_Q_FULL	- The queue is full; operation failed.
OS_ERR_Q_NOT_CRE↩ ATED	- The queue was not created, or it was already deleted.

2.8.4.10 os_status osQueueUnsafeHeadEnqueue (os_queue_handle *que_handle,* uint32_t *value*)

Inserts a value to the front of queue (unsafe version).

Parameters

in	*que_handle*	- The queue handle.
in	*value*	- The value to insert to the queue.

Return values

OS_SUCCESS	- The value was successfully inserted.
OS_ERR_Q_FULL	- The queue is full; operation failed.
OS_ERR_Q_NOT_CRE↩ ATED	- The queue was not created, or it was already deleted.

Warning

Use this function when you can be sure that the queue object cannot be accessed from any other source.

2.8.4.11 os_status osQueueDequeue (os_queue_handle *que_handle,* uint32_t ∗ *value*)

Removes the first value from the queue.

Parameters

in	*que_handle*	- The queue handle.
out	*value*	- Receives the first value in the queue.

Return values

OS_SUCCESS	- The first value was successfully removed.
OS_ERR_Q_EMPTY	- The queue is empty; operation failed.
OS_ERR_Q_NOT_CRE↩ ATED	- The queue was not created, or it was already deleted.

2.8.4.12 os_status osQueueUnsafeDequeue (os_queue_handle *que_handle,* uint32_t * *value*)

Removes the first value from the queue (unsafe version).

Parameters

in	*que_handle*	- The queue handle.
out	*value*	- Receives the first value in the queue.

Return values

OS_SUCCESS	- The first value was successfully removed.
OS_ERR_Q_EMPTY	- The queue is empty; operation failed.
OS_ERR_Q_NOT_CRE↩ ATED	- The queue was not created, or it was already deleted.

Warning

Use this function when you can be sure that the queue object cannot be accessed from any other source.

2.8.4.13 os_status osQueuePeek (os_queue_handle *que_handle,* uint32_t * *value*)

Queries the first value in the queue without removing it.

Parameters

in	*que_handle*	- The queue handle.
out	*value*	- Receives the first value in the queue.

Return values

OS_SUCCESS	- The first value was successfully removed.
OS_ERR_Q_EMPTY	- The queue is empty; operation failed.
OS_ERR_Q_NOT_CREATED	- The queue was not created, or it was already deleted.

2.8.4.14 os_status osQueueTailDequeue (os_queue_handle *que_handle,* uint32_t * *value*)

Removes the last value from the queue.

Parameters

in	*que_handle*	- The queue handle.
out	*value*	- Receives the last value in the queue.

Return values

OS_SUCCESS	- The first value was successfully removed.
OS_ERR_Q_EMPTY	- The queue is empty; operation failed.
OS_ERR_Q_NOT_CREATED	- The queue was not created, or it was already deleted.

2.8.4.15 os_status osQueueUnsafeTailDequeue (os_queue_handle *que_handle,* uint32_t * *value*)

Removes the last value from the queue (unsafe version).

Parameters

in	*que_handle*	- The queue handle.
out	*value*	- Receives the last value in the queue.

Return values

OS_SUCCESS	- The first value was successfully removed.
OS_ERR_Q_EMPTY	- The queue is empty; operation failed.
OS_ERR_Q_NOT_CRE↩ ATED	- The queue was not created, or it was already deleted.

Warning

Use this function when you can be sure that the queue object cannot be accessed from any other source.

2.8.4.16 os_queue_status osQueueStatus (os_queue_handle *que_handle*)

Queries the status of the queue (empty, full, etc.).

Parameters

in	*que_handle*	- The queue handle.

Returns

The queue status (OS_QUEUE_EMPTY, OS_QUEUE_MIDDLE, OS_QUEUE_FULL); if the queue was not created, OS_QUEUE_EMPTY will be returned.

2.8.4.17 os_queue_status osQueueUnsafeStatus (os_queue_handle *que_handle*)

Queries the status of the queue (unsafe version). Use this function when you can be sure that the queue object cannot be accessed from any other source.

Parameters

in	*que_handle*	- The queue handle.

Returns

The queue status (OS_QUEUE_EMPTY, OS_QUEUE_MIDDLE, OS_QUEUE_FULL); if the queue was not created, OS_QUEUE_EMPTY will be returned.

Warning

Use this function when you can be sure that the queue object cannot be accessed from any other source.

2.8.4.18 os_status osQueueFind (os_queue_handle * *que_handle,* bool *shared*)

Finds the first available queue number.

Parameters

out	*que_handle*	- Receives the available queue number, or NULL if no queue is available.
in	*shared*	- Set to TRUE to find an available shared queue, or FALSE for a regular queue.

Return values

OS_SUCCESS	- Found an available queue.
OS_ERR_Q_UNAVAIL↩ ABLE	- No queue is available.

2.8.4.19 os_status osQueueEnqueueMultiple (os_queue_handle *que_handle,* uint32_t * *value*)

Inserts a value to the queue.

Parameters

in	*que_handle*	- The queue handle.
out	*value*	- Pointer to an array of values to insert to queue

Return values

OS_SUCCESS	- The value was successfully inserted.
OS_ERR_Q_FULL	- The queue is full; operation failed.
OS_ERR_Q_NOT_CRE↩ ATED	- The queue was not created, or it was already deleted.

2.8.4.20 os_status osQueueUnsafeEnqueueMultiple (os_queue_handle *que_handle,* uint32_t * *value*)

Inserts a multiple value to the queue (unsafe version).

Parameters

in	*que_handle*	- The queue handle.
out	*value*	- Pointer to an array of values to insert to queue

Return values

OS_SUCCESS	- The value was successfully inserted.
OS_ERR_Q_FULL	- The queue is full; operation failed.
OS_ERR_Q_NOT_CRE↩ ATED	- The queue was not created, or it was already deleted.

Warning

Use this function when you can be sure that the queue object cannot be accessed from any other source.

2.8.4.21 os_status osQueueDequeueMultiple (os_queue_handle *que_handle,* uint32_t ∗ *value*)

Removes the first value from the queue.

Parameters

in	*que_handle*	- The queue handle.
out	*value*	- Receives the first value in the queue.

Return values

OS_SUCCESS	- The first value was successfully removed.
OS_ERR_Q_EMPTY	- The queue is empty; operation failed.
OS_ERR_Q_NOT_CRE↩ ATED	- The queue was not created, or it was already deleted.

2.8.4.22 os_status osQueueUnsafeDequeueMultiple (os_queue_handle *que_handle,* uint32_t ∗ *value*)

Removes the first value from the queue (unsafe version).

Parameters

in	*que_handle*	- The queue handle.
out	*value*	- Receives the first value in the queue.

Return values

OS_SUCCESS	- The first value was successfully removed.
OS_ERR_Q_EMPTY	- The queue is empty; operation failed.
OS_ERR_Q_NOT_CRE↩ ATED	- The queue was not created, or it was already deleted.

Warning

Use this function when you can be sure that the queue object cannot be accessed from any other source.

2.8.4.23 os_queue_shared osQueueShared (os_queue_handle *que_handle*)

Queries the status of the queue (shared, private)

Parameters

in	*que_handle*	- The queue handle.

Returns

The queue status (OS_QUEUE_PRIVATE, OS_QUEUE_SHARED).

2.8.4.24 os_status osQueueSize (os_queue_handle *que_handle,* uint32_t ∗ *size*)

Returns the maximum number of elements allowed in the queue

Parameters

in	*que_handle*	- The queue handle.
out	*size*	- The size of the queue.

Return values

OS_SUCCESS	- The size was returned successfully.
OS_ERR_Q_EMPTY	- The queue was not created.

2.8.4.25 os_status osQueueMultipleSize (os_queue_handle *que_handle,* uint32_t * *size*)

Returns the maximum number of elements allowed in the multiple queue

Parameters

in	*que_handle*	- The queue handle.
out	*size*	- The size of the queue.

Return values

OS_SUCCESS	- The size was returned successfully.
OS_ERR_Q_EMPTY	- The queue was not created.

2.8.4.26 os_status osQueueMultipleNum (os_queue_handle *que_handle,* uint32_t * *num*)

Returns the size of each element in the multiple queue in 32 bits

Parameters

in	*que_handle*	- The queue handle.
out	*num*	- The size of the queue.

Return values

OS_SUCCESS	- The size was returned successfully.
OS_ERR_Q_EMPTY	- The queue was not created.

2.9 Frames API

2.9.1 Overview

```
SmartDSP OS implementation of data frames.
```

Frames and frame pools are used (primarily) with the BIO module to perform I/O operations through frame/cell interfaces.

Data Structures

- struct os_frames_pool_t
- struct os_frame_t

Macros

- #define FRAME_NONE (-1)
- #define GET_MAX_BUFFER_SIZE(frame) (frame->data_part->block_size)
- #define osFrameBufferNew(frame) ((uint8_t *)osMemBlockSyncGet((frame)->data_part))
- #define osFrameBufferFree(frame, data) osMemBlockSyncFree((frame)->data_part, (void *)(data))
- #define osFrameInfoGet(frame) ((frame)->info)
- #define osFrameInfoSet(frame, owner_info) { (frame)->info = (uint32_t)(owner_info); }
- #define osFrameRefCntInc(frame) { frame->ref_count++;}
- #define osFrameRefCntDec(frame) { frame->ref_count--;}
- #define osFrameRefCntGet(frame) (frame->ref_count)
- #define osFrameRefCntSet(frame, count) { OS_ASSERT_COND((count) > 0); frame->ref_count = (count); }
- #define osFrameDataPartGet(frame) ((os_mem_part_t*)frame->data_part)
- #define osFrameLengthGet(frame) ((frame)->ctrl.buffers[0].data_length - (frame)->ctrl.buffers[0].offset)
- #define osFrameLengthSet(frame, length) { (frame)->ctrl.buffers[0].data_length = (uint32_t)((length) + (frame)->ctrl.buffers[0].offset); }
- #define osFrameOffsetGet(frame) osFrameBufferOffsetGet((frame), 0)
- #define osFrameOffsetBuffer(frame, buffer)
- #define osFrameTotalBuffers(frame) ((frame)->total)
- #define osFrameExpansionGet(frame) ((void *)&((frame)->ctrl.buffers[((frame)->total == 1) ? 1 : ((frame)->total + 1)]))
- #define osFrameExpansionSize(frame) ((frame)->expansion)
- #define osFrameSingleBufferGet(frame, length) osFrameBufferGet((frame), (length), NULL)
- #define osFrameSingleBufferSet(frame, data, length) osFrameBufferSet((frame), (data), (length), 0)
- #define osFramePoolNumBuffersGet(frame_pool) ((frame_pool)->buffers_per_frame)

Functions

- os_frames_pool_t * osFramePoolCreate (uint16_t total_frames, uint8_t buffers_per_frame, uint16_t frame_expansion, uint32_t frame_offset, os_mem_type mem_type)
- os_frame_t * osFrameGet (os_frames_pool_t *frames_pool, os_mem_part_t *data_mem_part)
- void osFrameRelease (os_frame_t *frame)
- int32_t osFrameBufferOffsetGet (os_frame_t *frame, uint8_t index)
- INLINE uint8_t * osFrameBufferOffsetSet (os_frame_t *frame, int8_t index, int32_t header_length)
- uint8_t osFrameUsedBuffers (os_frame_t *frame)
- uint8_t * osFrameBufferGet (os_frame_t *frame, uint32_t *length, int8_t *index)
- uint8_t * osFrameDataStart (os_frame_t *frame)
- uint8_t * osFrameBufferSet (os_frame_t *frame, uint8_t *data, uint32_t length, int8_t index)
- int8_t osFrameBufferAppend (os_frame_t *frame, uint8_t *data, uint32_t length)
- int8_t osFrameBufferPrepend (os_frame_t *frame, uint8_t *data, uint32_t length)
- uint8_t * osFrameHeadBufferRemove (os_frame_t *frame)

- uint8_t * osFrameTailBufferRemove (os_frame_t *frame)
- INLINE uint8_t * osFrameHeaderPush (os_frame_t *frame, uint32_t header_length)
- INLINE uint8_t * osFrameHeaderPop (os_frame_t *frame, uint32_t header_length)
- INLINE uint8_t * osFrameBufferSetWithOffset (os_frame_t *frame, uint8_t *data, uint32_t length, uint32_t offset)
- os_status osFrameTailRemove (os_frame_t *frame, uint32_t length)
- os_status osFrameTailAppend (os_frame_t *frame, uint32_t length)
- void osFrameDuplicate (os_frame_t *src_frame, os_frame_t *dst_frame)
- INLINE void osFrameCopyToMem (os_frame_t *frame, uint8_t *mem)
- INLINE os_status osFrameLldHeaderVerify (os_frame_t *frame, uint32_t header_length)

2.9.2 Data Structure Documentation

2.9.2.1 struct os_frames_pool_t

Frames pool.

The frames pool is required for dynamic allocation and release of frames. The application should not create static instances of this structure, but use osFramePoolCreate() to get a pointer to an initialized frames pool.

The frames pool creates a bank of frames with identical attributes (mainly, number of buffers per frame). These frames should be accessed only through osFrameGet() and osFrameFree().

Warning

The application must not modify the contents of this structure.

Data Fields

- os_mem_part_t * frames_part
- uint8_t buffers_per_frame
- uint8_t frame_expansion
- uint32_t init_frame_offset
- void *(* f_MemBlockGet)(os_mem_part_t *mem_part)
- void(* f_MemBlockFree)(os_mem_part_t *mem_part, void *addr)

2.9.2.1.1 Field Documentation

2.9.2.1.1.1 os_mem_part_t* os_frames_pool_t::frames_part

Pointer to a memory partition.

2.9.2.1.1.2 uint8_t os_frames_pool_t::buffers_per_frame

Total number of buffers per frame.

2.9.2.1.1.3 uint8_t os_frames_pool_t::frame_expansion

The expansion size of each frame.

2.9.2.1.1.4 uint32_t os_frames_pool_t::init_frame_offset

Number of bytes reserved for LLD upon frame creation.

2.9.2.1.1.5 void∗(∗ os_frames_pool_t::f_MemBlockGet)(os_mem_part_t ∗mem_part)

Frame allocation function pointer.

2.9.2.1.1.6 void(∗ os_frames_pool_t::f_MemBlockFree)(os_mem_part_t ∗mem_part, void ∗addr)

Frame deletion function pointer.

2.9.2.2 struct os_frame_t

Data frame structure.

The frame structure is used for holding frame-oriented data (an alternative notion may be "cell" or "packet"). A frame may contain a variable number of buffers for holding the data, but this number must be set before using the frame (when the frames pool is created).

The frame structure is much faster and efficient when using single-buffer frames. But, you can create frame pools with different number of buffers. If you are using ONLY single-buffer frames in your application, it is recommended to define FRAME_SINGLE_BUFFER_ONLY before including smartdsp_os.h. This would let your application use a faster implementation of the frames functions. There is no error checking in this case, so it is recommended to define it only after the application is stable.

A frame can contain a 32-bits value that represents the user's information (usually an index or a pointer to some other user's information structure). You can access this information using osFrameInfoSet() and osFrameInfoGet().

In addition, the frame structure can be expanded to any size, in case additional information (besides the buffers) is required. The expansion size is defined when the frames pool is created. You can call osFrameExpansionGet() to get a pointer to the expanded area, and osFrameExpansionSize() to get the size of the expanded area.

Because of the frame structure's dynamic nature, it should be accessed only through a frames pool.

Warning

> The size of the frame structure is dynamic, and depends on the number of data buffers in the frame and the expansion size. You must not use a straight-forward array of frames, and not even a single frame instance. Instead, you should create a frames pool for allocating and deleting frames.

NOTE: The frame structure is extended according to the maximum number of buffers in the frame and the expansion area of the frame. Do not use sizeof(os_frame_t) in your calculations!

Data Fields

- os_frames_pool_t * frames_pool
- uint32_t info
- uint16_t total
- uint8_t expansion
- uint8_t ref_count
- os_mem_part_t * data_part
- void * lld_channel_handle
- uint32_t total_length
- int32_t total_offset
- uint8_t used
- uint8_t first
- uint8_t last
- struct {
    uint32_t total_length
    int32_t total_offset
    uint8_t used
    uint8_t first
    uint8_t last
  } mbf

- uint32_t data_length
- int32_t offset
- uint8_t * data
- struct {
    uint32_t data_length
    int32_t offset
    uint8_t * data
  } buffers [1]

2.9.2.2.1 Field Documentation

2.9.2.2.1.1 os_mem_part_t* os_frame_t::data_part

Memory partition for data buffer.

2.9.2.2.1.2 void* os_frame_t::lld_channel_handle

LLD handler for buffers management.

2.9.2.2.1.3 os_frames_pool_t* os_frame_t::frames_pool

Backward reference to frame pool.

2.9.2.2.1.4 uint32_t os_frame_t::info

Owner's custom information.

2.9.2.2.1.5 uint16_t os_frame_t::total

Total number of buffers in frame.

2.9.2.2.1.6 uint8_t os_frame_t::expansion

Size (in bytes) of the optional frame's expansion.

2.9.2.2.1.7 uint8_t os_frame_t::ref_count

Reference counter.

2.9.2.2.1.8 uint32_t os_frame_t::total_length

Total length of data, in bytes (includes data and offset).

2.9.2.2.1.9 int32_t os_frame_t::total_offset

Total length of reserved header size.

2.9.2.2.1.10 uint8_t os_frame_t::used

Number of used buffers in frame.

2.9.2.2.1.11 uint8_t os_frame_t::first

Index of first buffer in frame.

2.9.2.2.1.12 uint8_t os_frame_t::last

Index of last buffer in frame.

2.9.2.2.1.13 struct { ... } os_frame_t::mbf

Multi-buffer frame management.

2.9.2.2.1.14 uint32_t os_frame_t::data_length

Length of data block, in bytes.

2.9.2.2.1.15 int32_t os_frame_t::offset

Reserved header size.
When negative, space is taken from init_frame_offset

2.9.2.2.1.16 uint8_t* os_frame_t::data

Pointer to the data block.

2.9.2.2.1.17 struct { ... } os_frame_t::buffers[1]

Dynamic array of data buffers.

2.9.3 Macro Definition Documentation

2.9.3.1 #define FRAME_NONE (-1)

Error code for some frame functions.

2.9.3.2 #define GET_MAX_BUFFER_SIZE(*frame*) (frame->data_part->block_size)

Returns the maximal capacity of the frame's buffers

Parameters

in	*frame*	- Pointer to the frame

Returns

Size of frame's buffers (in bytes)

2.9.3.3 #define osFrameBufferNew(*frame*) ((uint8_t *)osMemBlockSyncGet((frame)->data_part))

Allocates a new data buffer from the frame's buffers pool.

Parameters

in	*frame*	- Pointer to the frame.

Returns

Pointer to the allocated buffer; NULL if no buffer is available.

2.9.3.4 #define osFrameBufferFree(*frame, data*) osMemBlockSyncFree((frame)->data_part, (void *)(data))

Releases a frame's buffer back to the frame's buffers pool.

Parameters

in	*frame*	- Pointer to the frame.
in	*data*	- The data buffer to delete.

2.9.3.5 #define osFrameInfoGet(*frame*) ((frame)->info)

Gets the user's information from the frame.

Parameters

in	*frame*	- Pointer to the frame.

Returns

The user's information (uint32_t).

2.9.3.6 #define osFrameInfoSet(*frame, owner_info*) { (frame)->info = (uint32_t)(owner_info); }

Sets the user's information to the frame.

Parameters

in	*frame*	- Pointer to the frame.
in	*owner_info*	- User's information (uint32_t).

2.9.3.7 #define osFrameRefCntInc(*frame*) { frame->ref_count++;}

Increment the frame's reference counter.

Parameters

in	*frame*	- Pointer to the frame.

2.9.3.8 #define osFrameRefCntDec(*frame*) { frame->ref_count--;}

Decrement the frame's reference counter.

Parameters

in	*frame*	- Pointer to the frame.

2.9.3.9 #define osFrameRefCntGet(*frame*) (frame->ref_count)

Gets the frame's reference counter.

Parameters

in	*frame*	- Pointer to the frame.

2.9.3.10 #define osFrameRefCntSet(*frame, count*) { OS_ASSERT_COND((count) > 0); frame->ref_count = (count); }

Gets the frame's reference counter.

Parameters

in	*frame*	- Pointer to the frame
in	*count*	- New reference count value (higher than 0)

Warning

Setting the reference count directly is not the recommended way; you should use osFrameRefCnt↩ Inc() or osFrameRefCntDec() in the general use-case

2.9.3.11 #define osFrameDataPartGet(*frame*) ((os_mem_part_t*)frame->data_part)

Gets memory partition of frame's buffers

Parameters

in	*frame*	- Pointer to the frame.

Returns

Pointer to the memory partition of frame's buffers

2.9.3.12 #define osFrameLengthGet(*frame*) ((frame)->ctrl.buffers[0].data_length - (frame)->ctrl.buffers[0].offset)

Gets the total length of the frame's data.

Parameters

in	*frame*	- Pointer to the frame.

Returns

The total length of the frame's data (in bytes).

2.9.3.13 #define osFrameLengthSet(*frame, length*) { (frame)->ctrl.↩ buffers[0].data_length = (uint32_t)((length) + (frame)->ctrl.buffers[0].offset); }

Sets the total length of the frame's data.

Parameters

in	*frame*	- Pointer to the frame.
in	*length*	- The total length of the frame's data (uint32_t).

Warning

Refrain from using this with multi-buffer frames, because it does not perform checks on the number of buffers after the change.

2.9.3.14 #define osFrameOffsetGet(*frame*) osFrameBufferOffsetGet((frame), 0)

Gets the size of the frame's first buffer reserved header space

Parameters

in	*frame*	- Pointer to the frame.

Returns

The size of the frame's first buffer reserved header (in bytes).

2.9.3.15 #define osFrameOffsetBuffer(*frame, buffer*)

Value:

```
(uint8_t *)((buffer) + ((frame)->total == 1) ? (frame)->ctrl.buffers[0].offset : \
                                                (frame)->ctrl.buffers[(frame)->ctrl.mbf.first+1].offset)
      )
```

Offsets a given buffer with the frame's first offset (reserved header size). You can use this macro to get the correct position to fill a new buffer (returned by osFrameBufferNew()) before adding it as the first (or only) buffer in the frame. Note the example below:

```
uint8_t *buffer = osFrameBufferNew(frame);
uint8_t *data1 = osFrameOffsetBuffer(frame, buffer);
uint8_t *data2 = (uint8_t *)(buffer + osFrameOffsetGet(frame));
OS_ASSERT_COND(data1 == data2);
```

Parameters

in	*frame*	- Pointer to the frame.
in	*buffer*	- Pointer to a buffer.

Returns

A pointer to the data position in the buffer (after the offset).

2.9.3.16 #define osFrameTotalBuffers(*frame*) ((frame)->total)

Gets the total number of buffers (used or unused) in the frame.

Parameters

in	*frame*	- Pointer to the frame.

Returns

The total number of buffers in the frame.

2.9.3.17 #define osFrameExpansionGet(*frame*) ((void *)&((frame)->ctrl.buffers[((frame)->total == 1) ? 1 : ((frame)->total + 1)]))

Gets a pointer to the expansion area of the frame.

Parameters

in	*frame*	- Pointer to the frame.

Returns

A pointer to the expansion area of the frame.

Warning

This function does not verify that the frame contains an expansion area - it is the responsibility of the user. You can check the expansion size with osFrameExpansionSize().

2.9.3.18 #define osFrameExpansionSize(*frame*) ((frame)->expansion)

Gets the size of the expansion area of the frame.

Parameters

in	*frame*	- Pointer to the frame.

Returns

The size (in bytes) of the expansion area of the frame.

2.9.3.19 #define osFrameSingleBufferGet(*frame, length*) osFrameBufferGet((frame), (length), NULL)

Macro for calling osFrameBufferGet() on a single-buffer frame.

Parameters

in	*frame*	- Pointer to the frame.
out	*length*	- Receives the length of the buffer (in bytes).

Returns

The requested buffer; NULL if not valid.

2.9.3.20 #define osFrameSingleBufferSet(*frame, data, length*) osFrameBufferSet((frame), (data), (length), 0)

Macro for calling osFrameBufferSet() on a single-buffer frame.

Parameters

in	*frame*	- Pointer to the frame.
in	*data*	- The data buffer to set.
in	*length*	- The length of the data buffer (in bytes).

Returns

The previous data buffer. If this is a valid buffer (not NULL), it must be freed using osFrameBuffer↩ Free().

2.9.3.21 #define osFramePoolNumBuffersGet(*frame_pool*) ((frame_pool)->buffers_↩ per_frame)

Gets the max number of buffers per frame in the frame pool.

Parameters

in	*frame_pool*	- Pointer to the frame pool.

Returns

Maximal number of buffers per frame.

2.9.4 Function Documentation

2.9.4.1 os_frames_pool_t* osFramePoolCreate (uint16_t *total_frames*, uint8_t *buffers_per_frame*, uint16_t *frame_expansion*, uint32_t *frame_offset*, os_mem_type *mem_type*)

Creates a pool of frames with the given attributes.

This function creates a bank of frames with identical attributes (number of buffers per frame, and expansion size). The frames must be accessed only through osFrameGet() and osFrameFree().

Parameters

in	*total_frames*	- Required number of frames in the pool.
in	*buffers_per_frame*	- Maximum number of buffers per frame.
in	*frame_expansion*	- Expansion size for the frames (in bytes).
in	*frame_offset*	- Reserved header size for the frames (in bytes).
in	*mem_type*	- Memory type (local, shared, etc.) for the pool (NOT the frames' data buffers).

Returns

Pointer to the initialized frames pool; NULL if not enough memory is available.

2.9.4.2 os_frame_t* osFrameGet (os_frames_pool_t * *frames_pool*, os_mem_part_t * *data_mem_part*)

Allocates an empty frame from a frames pool.

Parameters

in	*frames_pool*	- Pointer to an initialized frames pool.
in	*data_mem_part*	- Pointer to the buffers pool for the frame (an initialized memory partition).

Returns

Pointer to the allocated frame; NULL if no frame is available.

2.9.4.3 void osFrameRelease (os_frame_t * *frame*)

Releases a frame back to the frames pool.

This function will decrement the frame's reference counter. If the counter reaches 0, the frame and all valid buffers of the frame are released.

Parameters

in	*frame*	- Pointer to the frame to delete.

2.9.4.4 int32_t osFrameBufferOffsetGet (os_frame_t * *frame,* uint8_t *index*)

Gets the size of the frame's reserved header space (offset) in the specified buffer. When working with single-buffer frames the reserved header space (offset) of the first and only buffer will be returned.

Parameters

in	*frame*	- Pointer to the frame.
in	*index*	- The index of the buffer for the requested offset.

Returns

The size of the frame's reserved header (in bytes).

2.9.4.5 INLINE uint8_t* osFrameBufferOffsetSet (os_frame_t * *frame,* int8_t *index,* int32_t *header_length*)

Sets the size of the frame's specified buffer reserved header space

Parameters

in	*frame*	- Pointer to the frame.
in	*index*	- Index of the buffer which should be modified.
in	*header_length*	- The length (in bytes) to set as a header (offset).

Returns

A pointer to the new data position in the specified buffer

2.9.4.6 uint8_t osFrameUsedBuffers (os_frame_t * *frame*)

Gets the number of used (valid) buffers in the frame.

Parameters

in	*frame*	- Pointer to the frame.

Returns

The number of used (valid) buffers in the frame.

2.9.4.7 uint8_t* osFrameBufferGet (os_frame_t * *frame*, uint32_t * *length*, int8_t * *index*)

Gets a data buffer from the frame, according to the given index.

Parameters

in	*frame*	- Pointer to the frame.
out	*length*	- Receives the length of the buffer (in bytes).
out	*index*	- Pointer to the index of the requested buffer. When the function returns, the pointer will receive the index of the next available buffer (FRAME_NONE if there are no more valid buffers).

Returns

The requested buffer; NULL if index is not valid.

Warning

Do not free the buffer that is returned by this function. You can only free buffers that are returned by osFrameBufferSet(), osFrameHeadBufferRemove() or osFrameTailBufferRemove().

2.9.4.8 uint8_t* osFrameDataStart (os_frame_t * *frame*)

Returns a pointer to the first data byte (in the first buffer).

The first data buffer may contain a reserved header space (offset) before the actual data. To reserve a header size, you should specify the frame offset when creating the frames pool. You can then push and pop headers to that reserved size using osFrameHeaderPush() and osFrameHeaderPop().

Parameters

in	*frame*	- Pointer to the frame.

Returns

A pointer to the first data byte in the first buffer.

2.9.4.9 uint8_t* osFrameBufferSet (os_frame_t * *frame*, uint8_t * *data*, uint32_t *length*, int8_t *index*)

Replaces an existing data buffer at the specified index.

For single-buffer frames, this function can be used even if the frame is empty. For multi-buffer frames, you should use the osFrameBufferAppend() to add the first buffer to the frame. If the index already contains a data buffer, the old buffer will be returned to the user.

Parameters

in	*frame*	- Pointer to the frame.
in	*data*	- The data buffer to set.
in	*length*	- The length of the data buffer (in bytes).
in	*index*	- The buffer's index in the frame.

Returns

The previous data buffer at the given index. If this is a valid buffer (not NULL), it must be freed using osFrameBufferFree().

Warning

This function does not modify the offset, if you want to modify the offset call OsFrameOffsetSet() after calling this function.

2.9.4.10 int8_t osFrameBufferAppend (os_frame_t * *frame*, uint8_t * *data*, uint32_t *length*)

Adds a new data buffer after the last buffer of the frame.

Parameters

in	*frame*	- Pointer to the frame.
in	*data*	- The data buffer.
in	*length*	- The length of the data (in bytes).

Returns

Number of used buffers in the frame after the operation; FRAME_NONE if the buffer cannot be added.

2.9.4.11 int8_t osFrameBufferPrepend (os_frame_t * *frame*, uint8_t * *data*, uint32_t *length*)

Adds a new data buffer before the first buffer of the frame.

Parameters

in	*frame*	- Pointer to the frame.
in	*data*	- The data buffer.
in	*length*	- The length of the data (in bytes).

Returns

Number of used buffers in the frame after the operation; FRAME_NONE if the buffer cannot be added.

2.9.4.12 uint8_t* osFrameHeadBufferRemove (os_frame_t * *frame*)

Removes the first data buffer from the frame.

Parameters

in	*frame*	- Pointer to the frame.

Returns

The first data buffer in the frame; NULL if no buffers exist. This buffer must be freed using osFrameBufferFree().

2.9.4.13 uint8_t* osFrameTailBufferRemove (os_frame_t * *frame*)

Removes the last data buffer from the frame.

Parameters

in	*frame*	- Pointer to the frame.

Returns

The last data buffer in the frame; NULL if no buffers exist. This buffer must be freed using osFrameBufferFree().

2.9.4.14 INLINE uint8_t* osFrameHeaderPush (os_frame_t * *frame*, uint32_t *header_length*)

Adds a given size to the frame header in the first buffer.

To reserve a header size (in the first buffer only), you should specify the frame offset when creating the frames pool. You can push headers to that reserved size, but you cannot exceed it.

Parameters

in	*frame*	- Pointer to the frame.
in	*header_length*	- The length (in bytes) to add as a header.

Returns

A pointer to the new position of the first data byte.

2.9.4.15 INLINE uint8_t* osFrameHeaderPop (os_frame_t * *frame*, uint32_t *header_length*)

Subtracts a given size from the frame header in the first buffer.

To reserve a header size (in the first buffer only), you should specify the frame offset when creating the frames pool. You can pop headers from that first buffer, but you must not exceed the length of the buffer.

Parameters

in	*frame*	- Pointer to the frame.

in	*header_length*	- The length (in bytes) to remove from the header.

Returns

A pointer to the new position of the first data byte.

2.9.4.16 INLINE uint8_t* osFrameBufferSetWithOffset (os_frame_t * *frame*, uint8_t * *data*, uint32_t *length*, uint32_t *offset*)

Replaces the (only) data buffer and sets its data offset.

This function is available for single-buffer frames only. If the frame already contains a data buffer, the old buffer will be returned to the user.

Parameters

in	*frame*	- Pointer to the frame.
in	*data*	- The data buffer to set.
in	*length*	- The length of the data buffer (in bytes).
in	*offset*	- The data offset (header space) for the frame.

Returns

The previous data buffer. If this is a valid buffer (not NULL), it must be freed using osFrameBuffer↩ Free().

Warning

This frame is available for single-buffer frames only. Use this function only if the buffers_per_frame in osFramePoolCreate equals 1.

2.9.4.17 os_status osFrameTailRemove (os_frame_t * *frame*, uint32_t *length*)

Removes data from the end of frame.

Parameters

in	*frame*	- Pointer to the frame.
in	*length*	- Number bytes to chop

Returns

The last data buffer in the frame; NULL if no buffers exist. This buffer must be freed using osFrameBufferFree().

2.9.4.18 os_status osFrameTailAppend (os_frame_t * *frame*, uint32_t *length*)

Appends data to the end of frame.

Parameters

in	*frame*	- Pointer to the frame.
in	*length*	- Number of bytes to add

Returns

The last data buffer in the frame; NULL if no buffers exist. This buffer must be freed using osFrameBufferFree().

2.9.4.19 void osFrameDuplicate (os_frame_t * *src_frame*, os_frame_t * *dst_frame*)

Duplicate the given frame including all buffers

Parameters

in	*src_frame*	- Pointer to the source frame.
in	*dst_frame*	- Pointer to the destination frame.

2.9.4.20 INLINE void osFrameCopyToMem (os_frame_t * *frame*, uint8_t * *mem*)

Copies frame data to continuous memory

Parameters

in	*frame*	- Pointer to the frame.
in	*mem*	- The data buffer to copy to.

Warning

mem should point at a memory location with enough space for osFrameLengthGet() bytes

2.9.4.21 INLINE os_status osFrameLldHeaderVerify (os_frame_t * *frame*, uint32_t *header_length*)

Verifies the LLD gets 0 offset and enough header space.

Parameters

in	*frame*	- Pointer to the frame.
in	*header_length*	- The length (in bytes) to verify.

Returns

Status.

Warning

This function is not supported for Multi-Buffer frames

2.10 Clock API

2.10.1 Overview

Modules

- B486x Clock API

2.10.2 B486x Clock API

2.10.2.1 Overview

API for reading the B486x Clock clock

Macros

- #define osCoreClockGet() (g_core_clock + 0)
- #define osPlatformClockGet() (g_platform_clock + 0)
- #define osSlowPeriphClockGet() (osPlatformClockGet() / 2)

Functions

- os_status b486xInitializeClocks (float clk_in)

2.10.2.2 Macro Definition Documentation

2.10.2.2.1 #define osCoreClockGet( ) (g_core_clock + 0)

Get the core frequency

Returns

Frequency [MHz]

2.10.2.2.2 #define osPlatformClockGet( ) (g_platform_clock + 0)

Get the platform clock frequency

Returns

Frequency [MHz]

2.10.2.2.3 #define osSlowPeriphClockGet( ) (osPlatformClockGet() / 2)

Get the slow system (peripherals) frequency

Returns

Frequency [MHz]

2.10.2.3 Function Documentation

2.10.2.3.1 os_status b486xInitializeClocks (float *clk_in*)

Calculate the clock frequencies in the system

Called by the OS during osInitialize(). Can be called by the application if performed PLL relocking

Parameters

in	*clk_in*	- Frequency of the CLKIN [MHz].

Returns

OS_SUCCESS if successful

2.11 Multi Core and Multi Device Management

2.11.1 Overview

Modules

- Spinlocks API
- Barriers API
- Intercore Messages

2.11.2 Spinlocks API

2.11.2.1 Overview

Modules

- Spinlocks initialization functions
- Spinlocks runtime functions
- SC3900 Spinlocks with barriers API

2.11.2.2 Spinlocks initialization functions

2.11.2.2.1 Overview

Functions

- void osSpinLockDataStore (os_spinlock_handle *lock)

- void osSpinLockDataRelease (os_spinlock_handle *lock)
- INLINE void osSpinLockInitialize (os_spinlock_handle *lock)
- INLINE void osSpinLockDisable (os_spinlock_handle *lock)

2.11.2.2.2 Function Documentation

2.11.2.2.2.1 void osSpinLockDataStore (os_spinlock_handle * *lock*)

Stores all spinlock handles for using in L1 defense feature.

Parameters

in	*lock*	- a pointer to spinlock handle.

2.11.2.2.2.2 void osSpinLockDataRelease (os_spinlock_handle * *lock*)

Release the spinlock from the L1 defense spinlock lists.

Parameters

in	*lock*	- a pointer to spinlock handle.

2.11.2.2.2.3 INLINE void osSpinLockInitialize (os_spinlock_handle * *lock*)

Initialize the lock and save its location for l1-defense purposes. Application should use this initialization also when recovering from warm reset mode 3.

Parameters

in	*lock*	- a pointer to spinlock handle.

Don't call this function after recovering from warm reset modes 1 & 2.

2.11.2.2.2.4 INLINE void osSpinLockDisable (os_spinlock_handle * *lock*)

Disable the lock and release the spinlock from the L1 defense spinlock lists.

Parameters

in	*lock*	- a pointer to spinlock handle.

2.11.2.3 Spinlocks runtime functions

2.11.2.3.1 Overview

Macros

- #define osSpinLockSwiftRelease(address) osSpinLockRelease(address)
- #define osSpinLockSwiftIrqRelease(address) osSpinLockIrqRelease(address)
- #define osSharedResourceLock(resource) osSpinLockIrqGet((os_spinlock_handle *)(resource))
- #define osSharedResourceUnlock(resource) osSpinLockIrqRelease((os_spinlock_handle *)(resource))

Functions

- void osSpinLockGet (os_spinlock_handle *address)
- void osSpinLockSwiftGet (os_spinlock_handle *address)
- uint32_t osSpinLockTryGet (os_spinlock_handle *address)
- uint32_t osSpinLockSwiftTryGet (os_spinlock_handle *address)
- void osSpinLockRelease (os_spinlock_handle *address)
- void osSpinLockIrqGet (os_spinlock_handle *address)
- void osSpinLockSwiftIrqGet (os_spinlock_handle *address)
- void osSpinLockIrqRelease (os_spinlock_handle *address)

2.11.2.3.2 Macro Definition Documentation

2.11.2.3.2.1 #define osSpinLockSwiftRelease(*address*) osSpinLockRelease(address)

Releases the given spinlock. No checking for OS_GUARD_DISABLE

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.2.2 #define osSpinLockSwiftIrqRelease(*address*) osSpinLockIrqRelease(address)

Releases the given spinlock and then enables interrupts. No checking for OS_GUARD_DISABLE

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.2.3 #define osSharedResourceLock(*resource*) osSpinLockIrqGet((os_spinlock_handle *)(resource))

Locks a protected structure.

This function can be used on structures that contain a spinlock as their first field. In a multicore system it would acquire the spinlock and disable interrupts. In a single core system it would just disable interrupts.

Parameters

in	*resource*	- Pointer to the resource that should be locked.

Warning

The locked structure must have a spinlock (uint32_t) as the first field in that structure.

2.11.2.3.2.4 #define osSharedResourceUnlock(*resource*) osSpinLockIrqRelease((os_spinlock_handle *)(resource))

Unlocks a protected structure.

This function can be used on structures that contain a spinlock as their first field. In a multicore system it would release the spinlock and enable interrupts. In a single core system it would just enable interrupts.

Parameters

in	*resource*	- Pointer to the resource that should be unlocked.

Warning

The unlocked structure must have a spinlock (uint32_t) as the first field in that structure.

2.11.2.3.3 Function Documentation

2.11.2.3.3.1 void osSpinLockGet (os_spinlock_handle * *address*)

Acquires the given spinlock (blocking operation). Ignored in single-core.

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.3.2 void osSpinLockSwiftGet (os_spinlock_handle * *address*)

Acquires the given spinlock (blocking operation). No checking for OS_GUARD_DISABLE. Ignored in single-core.

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.3.3 uint32_t osSpinLockTryGet (os_spinlock_handle * *address*)

Tries to acquire the given spinlock. Returns whether or not the spinlock is acquired (non-blocking operation). In single-core equals to OS_SUCCESS.

Parameters

in	*address*	- Address upon which to apply atomic operations.

Return values

0	- Spinlock was not acquired.
1	- Spinlock was acquired successfully.

2.11.2.3.3.4 uint32_t osSpinLockSwiftTryGet (os_spinlock_handle * *address*)

Tries to acquire the given spinlock. Returns whether or not the spinlock is acquired (non-blocking operation). No checking for OS_GUARD_DISABLE. In single-core equals to OS_SUCCESS.

Parameters

in	*address*	- Address upon which to apply atomic operations.

Return values

0	- Spinlock was not acquired.
1	- Spinlock was acquired successfully.

2.11.2.3.3.5 void osSpinLockRelease (os_spinlock_handle * *address*)

Releases the given spinlock. Ignored in single-core.

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.3.6 void osSpinLockIrqGet (os_spinlock_handle * *address*)

Disables interrupts and then acquires the given spinlock (blocking operation). In single-core equals to osHwiSwiftDisable().

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.3.7 void osSpinLockSwiftIrqGet (os_spinlock_handle * *address*)

Disables interrupts and then acquires the given spinlock (blocking operation). No checking for OS_GU↩ ARD_DISABLE. In single-core equals to osHwiSwiftDisable().

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.3.8 void osSpinLockIrqRelease (os_spinlock_handle * *address*)

Releases the given spinlock and then enables interrupts. In single-core equals to osHwiSwiftEnable().

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.4 SC3900 Spinlocks with barriers API

2.11.2.3.4.1 Overview

API for SC3900 Spinlocks with integrated Barriers usage

Functions

- void osSpinLockBarrierGet (os_spinlock_handle *address, uint32_t cache_policy)
- void osSpinLockBarrierSwiftGet (os_spinlock_handle *address, uint32_t cache_policy)
- uint32_t osSpinLockBarrierTryGet (os_spinlock_handle *address, uint32_t cache_policy)
- uint32_t osSpinLockBarrierSwiftTryGet (os_spinlock_handle *address, uint32_t cache_policy)
- void osSpinLockBarrierRelease (os_spinlock_handle *address, uint32_t cache_policy)
- void osSpinLockBarrierIrqGet (os_spinlock_handle *address, uint32_t cache_policy)
- void osSpinLockBarrierSwiftIrqGet (os_spinlock_handle *address, uint32_t cache_policy)
- void osSpinLockBarrierIrqRelease (os_spinlock_handle *address, uint32_t cache_policy)
- void osSpinLockNoBarrierGet (os_spinlock_handle *address)
- void osSpinLockNoBarrierSwiftGet (os_spinlock_handle *address)
- uint32_t osSpinLockNoBarrierTryGet (os_spinlock_handle *address)
- uint32_t osSpinLockNoBarrierSwiftTryGet (os_spinlock_handle *address)
- void osSpinLockNoBarrierRelease (os_spinlock_handle *address)
- void osSpinLockNoBarrierIrqGet (os_spinlock_handle *address)
- void osSpinLockNoBarrierSwiftIrqGet (os_spinlock_handle *address)
- void osSpinLockNoBarrierIrqRelease (os_spinlock_handle *address)

2.11.2.3.4.2 Function Documentation

2.11.2.3.4.2.1 void osSpinLockBarrierGet (os_spinlock_handle * *address*, uint32_t *cache_policy*)

Acquires the given spinlock (blocking operation). Ignored in single-core. Assert a load barrier to ensure chronological order.

Parameters

in	*address*	- Address upon which to apply atomic operations.
in	*cache_policy*	- Cache policy of the application. options: NOT_CACHED Not cacheable in any level of cache. L1_CACHED Cacheable in L1 cache only L2_CACHED Cacheable in L2 cache only L1_L2_CACHED Cacheable in L1 and L2 cache

2.11.2.3.4.2.2 void osSpinLockBarrierSwiftGet (os_spinlock_handle * *address*, uint32_t *cache_policy*)

Acquires the given spinlock (blocking operation). No checking for OS_GUARD_DISABLE. Ignored in single-core. Assert a load barrier to ensure chronological order.

Parameters

in	*address*	- Address upon which to apply atomic operations.
in	*cache_policy*	- Cache policy of the application. options: NOT_CACHED Not cacheable in any level of cache. L1_CACHED Cacheable in L1 cache only L2_CACHED Cacheable in L2 cache only L1_L2_CACHED Cacheable in L1 and L2 cache

2.11.2.3.4.2.3 uint32_t osSpinLockBarrierTryGet (os_spinlock_handle * *address*, uint32_t *cache_policy*)

Tries to acquire the given spinlock. Returns whether or not the spinlock is acquired (non-blocking operation). In single-core equals to OS_SUCCESS. Assert a load barrier to ensure chronological order.

Parameters

in	*address*	- Address upon which to apply atomic operations.
in	*cache_policy*	- Cache policy of the application. options: NOT_CACHED Not cacheable in any level of cache. L1_CACHED Cacheable in L1 cache only L2_CACHED Cacheable in L2 cache only L1_L2_CACHED Cacheable in L1 and L2 cache

Return values

0	- Spinlock was not acquired.
1	- Spinlock was acquired successfully.

2.11.2.3.4.2.4 uint32_t osSpinLockBarrierSwiftTryGet (os_spinlock_handle * *address*, uint32_t *cache_policy*)

Tries to acquire the given spinlock. Returns whether or not the spinlock is acquired (non-blocking operation). No checking for OS_GUARD_DISABLE. In single-core equals to OS_SUCCESS. Assert a load barrier to ensure chronological order.

Parameters

in	*address*	- Address upon which to apply atomic operations.
in	*cache_policy*	- Cache policy of the application. options: NOT_CACHED Not cacheable in any level of cache. L1_CACHED Cacheable in L1 cache only L2_CACHED Cacheable in L2 cache only L1_L2_CACHED Cacheable in L1 and L2 cache

Return values

0	- Spinlock was not acquired.
1	- Spinlock was acquired successfully.

2.11.2.3.4.2.5 void osSpinLockBarrierRelease (os_spinlock_handle * *address*, uint32_t *cache_policy*)

Releases the given spinlock. Ignored in single-core. Assert a store barrier to ensure chronological order.

Parameters

in	*address*	- Address upon which to apply atomic operations.
in	*cache_policy*	- Cache policy of the application. options: NOT_CACHED Not cacheable in any level of cache. L1_CACHED Cacheable in L1 cache only L2_CACHED Cacheable in L2 cache only L1_L2_CACHED Cacheable in L1 and L2 cache

2.11.2.3.4.2.6 void osSpinLockBarrierIrqGet (os_spinlock_handle * *address*, uint32_t *cache_policy*)

Disables interrupts and then acquires the given spinlock (blocking operation). In single-core equals to osHwiSwiftDisable(). Assert a load barrier to ensure chronological order.

Parameters

in	*address*	- Address upon which to apply atomic operations.
in	*cache_policy*	- Cache policy of the application. options: NOT_CACHED Not cacheable in any level of cache. L1_CACHED Cacheable in L1 cache only L2_CACHED Cacheable in L2 cache only L1_L2_CACHED Cacheable in L1 and L2 cache

2.11.2.3.4.2.7 void osSpinLockBarrierSwiftIrqGet (os_spinlock_handle * *address*, uint32_t *cache_policy*)

Disables interrupts and then acquires the given spinlock (blocking operation). No checking for OS_GUARD_DISABLE. In single-core equals to osHwiSwiftDisable(). Assert a load barrier to ensure chronological order.

Parameters

in	*address*	- Address upon which to apply atomic operations.
in	*cache_policy*	- Cache policy of the application. options: NOT_CACHED Not cacheable in any level of cache. L1_CACHED Cacheable in L1 cache only L2_CACHED Cacheable in L2 cache only L1_L2_CACHED Cacheable in L1 and L2 cache

2.11.2.3.4.2.8 void osSpinLockBarrierIrqRelease (os_spinlock_handle * *address*, uint32_t *cache_policy*)

Releases the given spinlock and then enables interrupts. In single-core equals to osHwiSwiftEnable(). Assert a store barrier to ensure chronological order.

Parameters

in	*address*	- Address upon which to apply atomic operations.
in	*cache_policy*	- Cache policy of the application. options: NOT_CACHED Not cacheable in any level of cache. L1_CACHED Cacheable in L1 cache only L2_CACHED Cacheable in L2 cache only L1_L2_CACHED Cacheable in L1 and L2 cache

2.11.2.3.4.2.9 void osSpinLockNoBarrierGet (os_spinlock_handle * *address*)

Acquires the given spinlock (blocking operation). Ignored in single-core.

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.4.2.10 void osSpinLockNoBarrierSwiftGet (os_spinlock_handle * *address*)

Acquires the given spinlock (blocking operation). No checking for OS_GUARD_DISABLE. Ignored in single-core.

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.4.2.11 uint32_t osSpinLockNoBarrierTryGet (os_spinlock_handle * *address*)

Tries to acquire the given spinlock. Returns whether or not the spinlock is acquired (non-blocking operation). In single-core equals to OS_SUCCESS.

Parameters

in	*address*	- Address upon which to apply atomic operations.

Return values

0	- Spinlock was not acquired.
1	- Spinlock was acquired successfully.

2.11.2.3.4.2.12 uint32_t osSpinLockNoBarrierSwiftTryGet (os_spinlock_handle * *address*)

Tries to acquire the given spinlock. Returns whether or not the spinlock is acquired (non-blocking operation). No checking for OS_GUARD_DISABLE. In single-core equals to OS_SUCCESS.

Parameters

in	*address*	- Address upon which to apply atomic operations.

Return values

0	- Spinlock was not acquired.
1	- Spinlock was acquired successfully.

2.11.2.3.4.2.13 void osSpinLockNoBarrierRelease (os_spinlock_handle * *address*)

Releases the given spinlock. Ignored in single-core.

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.4.2.14 void osSpinLockNoBarrierIrqGet (os_spinlock_handle * *address*)

Disables interrupts and then acquires the given spinlock (blocking operation). In single-core equals to osHwiSwiftDisable().

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.4.2.15 void osSpinLockNoBarrierSwiftIrqGet (os_spinlock_handle * *address*)

Disables interrupts and then acquires the given spinlock (blocking operation). No checking for OS_GUARD_DISABLE. In single-core equals to osHwiSwiftDisable().

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.2.3.4.2.16 void osSpinLockNoBarrierIrqRelease (os_spinlock_handle * *address*)

Releases the given spinlock and then enables interrupts. In single-core equals to osHwiSwiftEnable().

Parameters

in	*address*	- Address upon which to apply atomic operations.

2.11.3 Barriers API

2.11.3.1 Overview

Multicore barrier points synchronization API.

Modules

- SC3900 Spinlocks with barriers API
- Barriers runtime functions
- Barriers initialization functions

Data Structures

- struct os_barrier_t

2.11.3.2 Data Structure Documentation

2.11.3.2.1 struct os_barrier_t

OS Synchronization Barrier Structure.

Data Fields

- os_spinlock_handle guard
- uint8_t counter
- uint8_t height
- uint8_t comm_sensor
- uint8_t * sensors
- volatile uint8_t init_flag

2.11.3.2.1.1 Field Documentation

2.11.3.2.1.1.1 os_spinlock_handle os_barrier_t::guard

Access synchronization element.

2.11.3.2.1.1.2 uint8_t os_barrier_t::counter

Number of cores present.

2.11.3.2.1.1.3 uint8_t os_barrier_t::height

Number of cores to wait for.

2.11.3.2.1.1.4 uint8_t os_barrier_t::comm_sensor

Combined presence sensor.

2.11.3.2.1.1.5 uint8_t* os_barrier_t::sensors

Senses each core's presence.

2.11.3.2.1.1.6 volatile uint8_t os_barrier_t::init_flag

Initialization indicator (0/1).

2.11.3.3 Barriers runtime functions

2.11.3.3.1 Overview

Multicore barrier points synchronization runtime functions.

Functions

- void osCentralBarrierWait (volatile os_barrier_t *barrier)
- void osCentralBarrierVerify ()
- void osWaitForAllCores ()

2.11.3.3.2 Function Documentation

2.11.3.3.2.1 void osCentralBarrierWait (volatile os_barrier_t * *barrier*)

Waits on the given central barrier.

Waits for an amount of cores to reach the place in the code where this function is called. The amount of cores is set by the *num_of_cores* parameter of osCentralBarrierInit().

Parameters

in	*barrier*	- Address of the central barrier to wait on.

Warning

You must first initialize the barrier using osCentralBarrierInit().

2.11.3.3.2.2 void osCentralBarrierVerify ( )

Releases the central barrier according To mode 1 of warm reset for using in L1 defense feature.

2.11.3.3.2.3 void osWaitForAllCores ( )

Waits for all cores to reach the place in the code where this function is called.

This function uses an internal central barrier that is initialized when the synchronization module is initialized. There is no need to initialize an additional central barrier for this function. Ignored in single-core.

2.11.3.4 Barriers initialization functions

2.11.3.4.1 Overview

Multicore barrier points synchronization initialization functions.

Functions

- os_status osCentralBarrierInit (volatile os_barrier_t *barrier, uint8_t num_of_cores)

2.11.3.4.2 Function Documentation

2.11.3.4.2.1 os_status osCentralBarrierInit (volatile os_barrier_t * *barrier,* uint8_t *num_of_cores*)

Initializes the given central barrier structure.

Parameters

in,out	*barrier*	- Address of the central barrier to initialize.
in	*num_of_cores*	- Number of cores that will use the barrier.

Return values

OS_SUCCESS	- The barrier was initialized successfully.
OS_ERR_NO_MEMORY	- Not enough memory.

Warning

Initialization of a barrier allocates a small amount of memory that is never freed. It is therefore recommended that you do not call this function repeatedly; instead, call the function only in your application's initialization code.

2.11.4 Intercore Messages

2.11.4.1 Overview

Intercore Messages API.

This module is available only on multicore systems.

Modules

- Messages API
- Intercore Messages

2.11.4.2 Messages API

2.11.4.3 Intercore Messages

2.11.4.3.1 Overview

Intercore Message Queues API.

Functions

- os_status osMessageQueueInitialize (uint16_t num_of_message_queues)
- os_status osMessageQueueCreate (os_msg_handle msg_num, os_hwi_priority priority, os_hwi_arg hwi_arg, os_hwi_function msg_handler, uint16_t queue_length, uint16_t item_size)
- os_status osMessageQueueHwiGet (os_msg_handle msg_num, os_hwi_handle *hwi_num)
- os_status osMessageQueuePost (os_msg_handle msg_num, uint32_t *msg_data)
- os_status osMessageQueueGet (os_msg_handle msg_num, uint32_t *msg_data)
- os_status osMessageQueueDispatcher (os_msg_handle msg_num, os_msg_function handler, os_hwi_arg hwi_arg, uint32_t *msg_data)

2.11.4.3.2 Function Documentation

2.11.4.3.2.1 os_status osMessageQueueInitialize (uint16_t *num_of_message_queues*)

Initializes the intercore message queue. Must be called before using message queue.

Parameters

in	*num_of_message_queues*	- Number of message queues to initizali.

Return values

OS_SUCCESS	- The message queues were successfully initilized.
OS_ERR_NO_MEMORY	- memory allocation for message queue structure failed.

2.11.4.3.2.2 os_status osMessageQueueCreate (os_msg_handle *msg_num*, os_hwi_priority *priority*, os_hwi_arg *hwi_arg*, os_hwi_function *msg_handler*, uint16_t *queue_length*, uint16_t *item_size*)

Installs an intercore message handler for the calling core.

Enables the calling core to be notified when the specified message arrives. The calling core can then handle the message.

Parameters

in	*msg_num*	- The intercore message number.
in	*priority*	- Priority of interrupt generated at post
in	*hwi_arg*	- The argument to use when msg_handler is called
in	*msg_handler*	- The function to call when the message arrives.
in	*queue_length*	- Number of entries in the queue
in	*item_size*	- Size of each entry in the queue

Return values

OS_SUCCESS	- The message handler was successfully installed.
OS_ERR_MSG_INVALID	- Invalid message number.
OS_ERR_MSG_FUNC↩ TION_INVALID	- NULL message handler.
OS_ERR_MSG_ALREA↩ DY_CREATED	- The message number is already created.

2.11.4.3.2.3 os_status osMessageQueueHwiGet (os_msg_handle *msg_num*, os_hwi_handle * *hwi_num*)

Retrive interrupt information of message queue

Parameters

in	*msg_num*	- The intercore message number.
out	*hwi_num*	- Interrupt number assotiated with message queue

Return values

OS_SUCCESS	- The message handler is linked to interrupt.
OS_FAIL	- Invalid message num.

2.11.4.3.2.4 os_status osMessageQueuePost (os_msg_handle *msg_num*, uint32_t * *msg_data*)

Posts the given message number with the given data to the destination core.

Parameters

in	*msg_num*	- The intercore message number.
in	*msg_data*	- The data to include in the message.

Return values

OS_SUCCESS	- The message was successfully posted.
OS_ERR_MSG_INVALID	- Invalid message number.
OS_ERR_MSG_DEST_↩ INVALID	- Invalid destination.
OS_ERR_MSG_BUSY	- Destination is locked for posting.

2.11.4.3.2.5 os_status osMessageQueueGet (os_msg_handle *msg_num,* uint32_t ∗ *msg_data*)

Retrieves the data from a message that was posted to the calling core.

This function must be called by the message handler.

Parameters

in	*msg_num*	- The intercore message number.
out	*msg_data*	- The intercore message.

Returns

OS_SUCCESS if succeeded.

2.11.4.3.2.6 os_status osMessageQueueDispatcher (os_msg_handle *msg_num,* os_msg_function *handler,* os_hwi_arg *hwi_arg,* uint32_t ∗ *msg_data*)

Retrieves the data from a message that was posted to the calling core.

This function must be called by the message handler.

Parameters

in	*msg_num*	- The intercore message number.
in	*handler*	- Pointer to message handler.
in	*hwi_arg*	- Message handler argument
out	*msg_data*	- The intercore message.

Returns

status of message processing

2.12 Multitasking

2.12.1 Overview

Modules

- Asymmetric Multiprocessing Scheduling

2.12.2 Asymmetric Multiprocessing Scheduling

2.12.2.1 Overview

Modules

- Event Semaphores API
- Event Queues API
- Tasks API

2.12.2.2 Event Semaphores API

2.12.2.2.1 Overview

```
Semaphore Events setup and control.
```

Modules

- Event Semaphores RunTime functions

2.12.2.2.2 Event Semaphores RunTime functions

2.12.2.2.2.1 Overview

Functions

- os_status osEventSemaphoreFind (os_event_handle *event_handle)
- os_status osEventSemaphoreCreate (os_event_handle event_handle, uint32_t count)
- os_status osEventSemaphoreDelete (os_event_handle event_handle)
- os_status osEventSemaphoreReset (os_event_handle event_handle, uint32_t count)
- os_status osEventSemaphoreCountGet (os_event_handle event_handle, uint32_t *count)
- os_status osEventSemaphorePend (os_event_handle event_handle, uint32_t timeout)
- os_status osEventSemaphoreAccept (os_event_handle event_handle)
- os_status osEventSemaphorePost (os_event_handle event_handle, os_task_handle *resumed_task)

2.12.2.2.2.2 Function Documentation

2.12.2.2.2.2.1 os_status osEventSemaphoreFind (os_event_handle * *event_handle*)

Get a handle for an event.

Get a handle that can be used to create an semaphore event.

Parameters

out	*event_handle*	- Handle to an event semaphore structure.

Return values

OS_SUCCESS	: Found.
OS_ERR_EVENT_SEM↩ A_UNAVAILABLE	: Not Found.

2.12.2.2.2.2.2 os_status osEventSemaphoreCreate (os_event_handle *event_handle,* uint32_t *count*)

Creates a semaphore event.

Initializes a semaphore and an empty waiting list. i.e - initialize the semaphore counter, and place it and the empty waiting list in an event structure.

Parameters

in	*event_handle*	- Handle to an event semaphore structure.
in	*count*	- initial value of semaphore.

Return values

OS_SUCCESS	: Created.
OS_ERR_EVENT_SEM↩ A_ALREADY_CREATED	: Not Created.

2.12.2.2.2.2.3 os_status osEventSemaphoreDelete (os_event_handle *event_handle*)

Deletes the semaphore event.

Frees the event structure that was used for a counting semaphore, and resume all the tasks that were pending on it.

Parameters

in	*event_handle*	- Handle to a semaphore event.

Return values

OS_SUCCESS	: Deleted.
OS_ERR_EVENT_SEM↩ A_INVALID	: Not a Semaphore event.

2.12.2.2.2.2.4 os_status osEventSemaphoreReset (os_event_handle *event_handle,* uint32_t *count*)

reset a semaphore event - release all waiting tasks and set semaphore's count.

Parameters

in	*event_handle*	- Handle to an event semaphore structure.
in	*count*	- initial value of semaphore.

Return values

OS_SUCCESS	: Reset.
OS_ERR_EVENT_SEM↩ A_INVALID	: Not a Semaphore event.

2.12.2.2.2.2.5 os_status osEventSemaphoreCountGet (os_event_handle *event_handle,* uint32_t * *count*)

Gets semaphore value.

Parameters

in	*event_handle*	- Handle to an event semaphore structure.
out	*count*	- value of semaphore.

Return values

OS_SUCCESS	: OK.
OS_ERR_EVENT_SEM↩ A_INVALID	: bad semaphore.

2.12.2.2.2.2.6 os_status osEventSemaphorePend (os_event_handle *event_handle,* uint32_t *timeout*)

Pend on a counting semaphore.

The calling task will block until a the semaphore will be posted or until its timeout shall expire.

Parameters

in	*event_handle*	- Handle to an event structure.
in	*timeout*	- Timeout to pend in ticks.

Return values

OS_SUCCESS	: Got the ersource.
OS_ERR_EVENT_SEM↩ A_PEND_INTERR	: called in interrupt.
OS_ERR_EVENT_SEM↩ A_TIMEOUT	: Timeout expired.

Warning

background task can not call this function.
SWI and HWI can not call this function.

2.12.2.2.2.2.7 os_status osEventSemaphoreAccept (os_event_handle *event_handle*)

Try to counting semaphore without pending.

This function checks if semaphore counter is not zero. If its not decrement it and return to the caller. Otherwise return a status indicating that counter is zero. This is a non blocking function so it can be used in interrupt and software interrupt contexts.

Parameters

in	*event_handle*	- Handle to an event structure.

Return values

OS_SUCCESS	: Got the semaphore.
OS_ERR_EVENT_SEM↩ A_UNAVAILABLE	: counter is 0.

2.12.2.2.2.2.8 os_status osEventSemaphorePost (os_event_handle *event_handle,* os_task_handle * *resumed_task*)

Posts a counting semaphore.

If tasks are waiting in the waiting list of the semaphore, wake the first one, otherwise increment the counter for tasks that will pend or accept it later.

Parameters

in	*event_handle*	- Handle to an event structure.
out	*resumed_task*	- Task that was resumed or NULL.

Return values

OS_SUCCESS	: Got the ersource.
OS_EVENT_SEMAPH↩ ORE_OVERFLOW	: if semaphore is full

2.12.2.3 Event Queues API

2.12.2.3.1 Overview

Queue Events setup and control.

Modules

- Event Queues RunTime functions

2.12.2.3.1.0.9 Event Queues RunTime functions

2.12.2.3.1.0.10 Overview

Functions

- os_status osEventQueueFind (os_event_handle ∗event_handle)
- os_status osEventQueueCreate (os_event_handle event_handle, uint16_t size)
- os_status osEventQueuePend (os_event_handle event_handle, uint32_t ∗message, uint32_t timeout)
- os_status osEventQueueAccept (os_event_handle event_handle, uint32_t ∗message)
- os_status osEventQueuePost (os_event_handle event_handle, uint32_t message, os_task_handle ∗resumed_task)
- os_status osEventQueueReset (os_event_handle event_handle, uint32_t message)

2.12.2.3.1.0.11 Function Documentation

2.12.2.3.1.0.12 os_status osEventQueueFind (os_event_handle ∗ *event_handle*)

Get a handle for an event.

Get a handle that can be used to create a queue event.

Parameters

out	*event_handle*	- Handle to an event queue structure.

Return values

OS_SUCCESS	: Found.
OS_ERR_Q_UNAVAIL↩ ABLE	: No more queues in the system.
OS_ERR_EVENT_QUE↩ UE_UNAVAILABLE	: No more event queues in the system.

2.12.2.3.1.0.13 os_status osEventQueueCreate (os_event_handle *event_handle,* uint16_t *size*)

Creates a queue event. Initializes a queue and an empty waiting list. i.e - find a queue using osQueue↩ Find(), creates it using osQueueCreate(). Then places a handle to this queue and to the empty waiting list in an event structure.

Parameters

in	*event_handle*	- Handle to an event queue structure.
in	*size*	- Queue size.

Return values

OS_SUCCESS	: Created.
OS_ERR_EVENT_QUE↩ UE_ALREADY_CREAT↩ ED	: Not Created.
Error	status, encoded in os_error.h, for other errors.

2.12.2.3.1.0.14 os_status osEventQueuePend (os_event_handle *event_handle,* uint32_t * *message,* uint32_t *timeout*)

Pend on an event queue.

The calling task will block until a message will be posted to the queue or until its timeout shall expire.

Parameters

in	*event_handle*	- Handle to an event structure.
out	*message*	- Recieved message.
in	*timeout*	- Timeout to pend in ticks.

Return values

OS_SUCCESS	: Got the resource.
OS_ERR_EVENT_QUE↩ UE_PEND_INTERR	: Cannot block in interrupt.
OS_ERR_EVENT_PEN↩ D_IN_LOCK	: Can not block a running locked task. will cause a crash.
OS_ERR_EVENT_QUE↩ UE_TIMEOUT	: Timeout expired..

Warning

background task can not call this function.
SWI and HWI can not call this function.

2.12.2.3.1.0.15 os_status osEventQueueAccept (os_event_handle *event_handle,* uint32_t ∗ *message*)

Try to get A message from a queue without pending.

This function checks if there is a message available in the queue and if there is returns it to the caller. Otherwise return a status indicating that queue is empty. This is a non blocking function so it can be used in interrupt and software interrupt contexts.

Parameters

in	*event_handle*	- Handle to an event structure.
out	*message*	- Recieved message.

Return values

OS_SUCCESS	: Got the resource.
OS_ERR_Q_NOT_CRE↩ ATED	: Queue is not created.
OS_ERR_Q_EMPTY	: No message in the queue.

2.12.2.3.1.0.16 os_status osEventQueuePost (os_event_handle *event_handle,* uint32_t *message,* os_task_handle ∗ *resumed_task*)

Posts a message to a queue.

If tasks are waiting in the waiting list of the queue, wake the first one and give it the message, otherwise queue the message for tasks that will pend or accept it later.

Parameters

in	*event_handle*	- Handle to an event structure.
in	*message*	- Message to post.
out	*resumed_task*	- Task that was resumed or NULL.

Return values

OS_SUCCESS	: Got the ersource.
OS_ERR_Q_NOT_CRE↩ ATED	: Queue was not created.
OS_EVENT_QUEUE_↩ OVERFLOW	: Queue is full.

2.12.2.3.1.0.17 os_status osEventQueueReset (os_event_handle *event_handle,* uint32_t *message*)

Release event pending tasks with posted message and empty queue of messages if there are no pending tasks.

Parameters

in	*event_handle*	- Handle to an event structure.
in	*message*	- Message to post.

Return values

OS_SUCCESS	: Got the event queue was reset.
OS_ERR_EVENT_QUE↩ UE_INVALID	: event_handle is not event queue.

2.12.2.4 Tasks API

2.12.2.4.1 Overview

`Multitasking setup and control.`

Modules

- Event Queues RunTime functions
- Tasks Runtime API.

2.13 Architectures

2.13.1 Overview

Modules

- CCSR Memory Maps
- Peripheral Memory Maps
- SC39XX Memory Maps
- B4860 LAW
- DSP Platform Definitions
- SmartDsp OS Devices
- Hardware Semaphore API
- GPIO (general purpose I/O) API
- IPC (Inter Process Communication) API
- Heterogeneous API
- HW Interrupts Map
- B4860 L1 Defense

2.13.2 CCSR Memory Maps

2.13.2.1 Overview

Modules

- B4860 CCSR Memory Maps

2.13.2.2 B4860 CCSR Memory Maps

2.13.2.2.1 Overview

Data Structures

- struct dpaa_memmap_t
- struct pa_cluster_l2_map_t

2.13.2.2.2 Data Structure Documentation

2.13.2.2.2.1 struct dpaa_memmap_t

Data Path Acceleration Architecture(DPAA) Registers.

2.13.2.2.2.2 struct pa_cluster_l2_map_t

PA Cluster L2 control Map.

Data Fields

- volatile uint32_t l2csr0
- volatile uint32_t l2csr1
- volatile uint32_t l2cfg0
- volatile uint32_t l2ipbrr1
- volatile uint32_t l2ipbrr2
- volatile uint32_t l2errinjhi
- volatile uint32_t l2errinjlo
- volatile uint32_t l2errinjctl
- volatile uint32_t l2captdatahi
- volatile uint32_t l2captdatalo
- volatile uint32_t l2captecc
- volatile uint32_t l2errdet
- volatile uint32_t l2errdis
- volatile uint32_t l2errinten
- volatile uint32_t l2errattr
- volatile uint32_t l2erreaddr
- volatile uint32_t l2erraddr
- volatile uint32_t l2errctl
- volatile uint32_t l2hdbcr0
- volatile uint32_t l2hdbcr1
- volatile uint32_t l2hdbcr2

2.13.2.2.2.2.1 Field Documentation

2.13.2.2.2.2.2 volatile uint32_t pa_cluster_l2_map_t::l2csr0

L2 Cache Control and Status Register 0.

2.13.2.2.2.2.3 volatile uint32_t pa_cluster_l2_map_t::l2csr1

L2 Cache Control and Status Register 1.

2.13.2.2.2.2.4 volatile uint32_t pa_cluster_l2_map_t::l2cfg0

L2 Cache Configuration Register 0.

2.13.2.2.2.2.5 volatile uint32_t pa_cluster_l2_map_t::l2ipbrr1

L2 IP Block Revision Register 1.

2.13.2.2.2.2.6 volatile uint32_t pa_cluster_l2_map_t::l2ipbrr2

L2 IP Block Revision Register 2.

2.13.2.2.2.2.7 volatile uint32_t pa_cluster_l2_map_t::l2errinjhi

L2 Cache Error Injection Mask High Register.

2.13.2.2.2.2.8 volatile uint32_t pa_cluster_l2_map_t::l2errinjlo

L2 Cache Error Injection Mask Low Register.

2.13.2.2.2.9 volatile uint32_t pa_cluster_l2_map_t::l2errinjctl

L2 Cache Error Injection Control Register.

2.13.2.2.2.10 volatile uint32_t pa_cluster_l2_map_t::l2captdatahi

L2 Error Capture Data High Register.

2.13.2.2.2.11 volatile uint32_t pa_cluster_l2_map_t::l2captdatalo

L2 Error Capture Data Low Register.

2.13.2.2.2.12 volatile uint32_t pa_cluster_l2_map_t::l2captecc

L2 Cache Error Capture ECC Syndrome Register.

2.13.2.2.2.13 volatile uint32_t pa_cluster_l2_map_t::l2errdet

L2 Cache Error Detect Register.

2.13.2.2.2.14 volatile uint32_t pa_cluster_l2_map_t::l2errdis

L2 Cache Error Disable Register.

2.13.2.2.2.15 volatile uint32_t pa_cluster_l2_map_t::l2errinten

L2 Cache Error Interrupt Enable Register.

2.13.2.2.2.16 volatile uint32_t pa_cluster_l2_map_t::l2errattr

L2 Cache Error Attribute Register.

2.13.2.2.2.17 volatile uint32_t pa_cluster_l2_map_t::l2erreaddr

L2 Error Extended Address Register.

2.13.2.2.2.18 volatile uint32_t pa_cluster_l2_map_t::l2erraddr

L2 Cache Error Address Register.

2.13.2.2.2.19 volatile uint32_t pa_cluster_l2_map_t::l2errctl

L2 Cache Error Control Register.

2.13.2.2.2.20 volatile uint32_t pa_cluster_l2_map_t::l2hdbcr0

L2 Hardware Debug Control Register 0.

2.13.2.2.2.21 volatile uint32_t pa_cluster_l2_map_t::l2hdbcr1

L2 Hardware Debug Control Register 1.

2.13.2.2.2.22 volatile uint32_t pa_cluster_l2_map_t::l2hdbcr2

L2 Hardware Debug Control Register 2.

2.13.3 Peripheral Memory Maps

2.13.3.1 Overview

Modules

- OCN DMA Memory Map
- CPRI Memory Map
- I2C Memory Map
- Maple-B3 Memory Map
- sRIO Memory Map
- Timers 16 bit Memory Map
- Timers 32 bit Memory Map
- Watchdog timer Memory Map
- VSG Memory Map

Macros

- #define CPRI_MAX_NUM_OF_ANTENNA_CARRIERS 24

2.13.3.2 Macro Definition Documentation

2.13.3.2.1 #define CPRI_MAX_NUM_OF_ANTENNA_CARRIERS 24

The maximum number of CPRI antenna carriers.

2.13.3.3 OCN DMA Memory Map

2.13.3.3.1 Overview

```
Definition of OCN DMA registers and memory map
```

Data Structures

- struct ocn_dma_ch_map_t
- struct ocn_dma_memmap_t

2.13.3.3.2 Data Structure Documentation

2.13.3.3.2.1 struct ocn_dma_ch_map_t

On-Chip_Network (OCN) DMA Channel Registers.

Data Fields

- volatile uint32_t mr
- volatile uint32_t sr
- volatile uint32_t eclndar
- volatile uint32_t clndar
- volatile uint32_t satr
- volatile uint32_t sar
- volatile uint32_t datr
- volatile uint32_t dar
- volatile uint32_t bcr
- volatile uint32_t enlndar
- volatile uint32_t nlndar
- volatile uint32_t eclsdar
- volatile uint32_t clsdar
- volatile uint32_t enlsdar
- volatile uint32_t nlsdar
- volatile uint32_t ssr
- volatile uint32_t dsr

2.13.3.3.2.1.1 Field Documentation

2.13.3.3.2.1.2 volatile uint32_t ocn_dma_ch_map_t::mr

Mode register.

2.13.3.3.2.1.3 volatile uint32_t ocn_dma_ch_map_t::sr

Status register.

2.13.3.3.2.1.4 volatile uint32_t ocn_dma_ch_map_t::eclndar

Current link descriptor extended address register.

2.13.3.3.2.1.5 volatile uint32_t ocn_dma_ch_map_t::clndar

Current link descriptor address register.

2.13.3.3.2.1.6 volatile uint32_t ocn_dma_ch_map_t::satr

Source attributes register.

2.13.3.3.2.1.7 volatile uint32_t ocn_dma_ch_map_t::sar

Source address register.

2.13.3.3.2.1.8 volatile uint32_t ocn_dma_ch_map_t::datr

Destination attributes register.

2.13.3.3.2.1.9 volatile uint32_t ocn_dma_ch_map_t::dar

Destination address register.

2.13.3.3.2.1.10 volatile uint32_t ocn_dma_ch_map_t::bcr

Byte count register.

2.13.3.3.2.1.11 volatile uint32_t ocn_dma_ch_map_t::enlndar

Next link descriptor extended address register.

2.13.3.3.2.1.12 volatile uint32_t ocn_dma_ch_map_t::nlndar

Next link descriptor address register.

2.13.3.3.2.1.13 volatile uint32_t ocn_dma_ch_map_t::eclsdar

Current list descriptor extended address register.

2.13.3.3.2.1.14 volatile uint32_t ocn_dma_ch_map_t::clsdar

Current list descriptor address register.

2.13.3.3.2.1.15 volatile uint32_t ocn_dma_ch_map_t::enlsdar

Next list descriptor extended address register.

2.13.3.3.2.1.16 volatile uint32_t ocn_dma_ch_map_t::nlsdar

Next list descriptor address register.

2.13.3.3.2.1.17 volatile uint32_t ocn_dma_ch_map_t::ssr

Source stride register.

2.13.3.3.2.1.18 volatile uint32_t ocn_dma_ch_map_t::dsr

Destination stride register.

2.13.3.3.2.2 struct ocn_dma_memmap_t

DMA to On-Chip_Network (OCN) Registers.

Data Fields

- ocn_dma_ch_map_t dma_ch_map1 [4]

- volatile uint32_t dgsr0
- ocn_dma_ch_map_t dma_ch_map2 [4]
- volatile uint32_t dgsr1

2.13.3.3.2.2.1 Field Documentation

2.13.3.3.2.2.2 ocn_dma_ch_map_t ocn_dma_memmap_t::dma_ch_map1[4]

OCN DMA Channels.

2.13.3.3.2.2.3 volatile uint32_t ocn_dma_memmap_t::dgsr0

General status register.

2.13.3.3.2.2.4 ocn_dma_ch_map_t ocn_dma_memmap_t::dma_ch_map2[4]

OCN DMA Channels.

2.13.3.3.2.2.5 volatile uint32_t ocn_dma_memmap_t::dgsr1

General status register.

2.13.3.4 CPRI Memory Map

2.13.3.4.1 Overview

```
Definition of CPRI registers and memory map
```

Data Structures

- struct cpri_framer_registers_t
- struct cpri_complex_registers_t
- struct cpri_control_registers_t
- struct cpri_status_registers_t
- struct cpri_configuration_memories_t
- struct cpri_general_registers_t
- struct cpri_unit_map_t

2.13.3.4.2 Data Structure Documentation

2.13.3.4.2.1 struct cpri_framer_registers_t

CPRI framer registers.

Data Fields

- volatile uint32_t cpri_status
- volatile uint32_t cpri_config
- volatile uint32_t cpri_lcv

- volatile uint32_t cpri_bfn
- volatile uint32_t cpri_hfn
- volatile uint32_t cpri_hw_reset
- volatile uint32_t cpri_cm_config
- volatile uint32_t cpri_cm_status
- volatile uint32_t cpri_rx_delay
- volatile uint32_t cpri_round_delay
- volatile uint32_t cpri_ex_delay_config
- volatile uint32_t cpri_ex_delay_status
- volatile uint32_t cpri_tx_prot_ver
- volatile uint32_t cpri_tx_scr_seed
- volatile uint32_t cpri_rx_scr_seed
- volatile uint32_t cpri_serdes_config
- volatile uint32_t cpri_map_config
- volatile uint32_t cpri_map_cnt_config
- volatile uint32_t cpri_map_tbl_config
- volatile uint32_t cpri_map_offset_rx
- volatile uint32_t cpri_map_offset_tx
- volatile uint32_t cpri_start_offset_tx
- volatile uint32_t cpri_iq_rx_buf_status1
- volatile uint32_t cpri_iq_rx_buf_status2
- volatile uint32_t cpri_iq_tx_buf_status1
- volatile uint32_t cpri_iq_tx_buf_status2
- volatile uint32_t eth_rx_status
- volatile uint32_t eth_config_1
- volatile uint32_t eth_config_2
- volatile uint32_t eth_rx_control
- volatile uint32_t eth_rx_ex_status
- volatile uint32_t eth_addr_msb
- volatile uint32_t eth_addr_lsb
- volatile uint32_t eth_hash_table
- volatile uint32_t eth_config_3
- volatile uint32_t eth_cnt_rx_frame
- volatile uint32_t eth_cnt_tx_frame
- volatile uint32_t hdlc_rx_status
- volatile uint32_t hdlc_config_1
- volatile uint32_t hdlc_config_2
- volatile uint32_t hdlc_rx_control
- volatile uint32_t hdlc_rx_ex_status
- volatile uint32_t hdlc_config_3
- volatile uint32_t hdlc_cnt_rx_frame
- volatile uint32_t hdlc_cnt_tx_frame

2.13.3.4.2.1.1 Field Documentation

2.13.3.4.2.1.2 volatile uint32_t cpri_framer_registers_t::cpri_status

CPRI status register.

2.13.3.4.2.1.3 volatile uint32_t cpri_framer_registers_t::cpri_config

CPRI configuration register.

2.13.3.4.2.1.4 volatile uint32_t cpri_framer_registers_t::cpri_lcv

CPRI Receive Line Coding Violation Counter.

2.13.3.4.2.1.5 volatile uint32_t cpri_framer_registers_t::cpri_bfn

CPRI Recovered BFN Counter.

2.13.3.4.2.1.6 volatile uint32_t cpri_framer_registers_t::cpri_hfn

CPRI Recovered HFN Counter.

2.13.3.4.2.1.7 volatile uint32_t cpri_framer_registers_t::cpri_hw_reset

CPRI Hardware Reset from Control Word.

2.13.3.4.2.1.8 volatile uint32_t cpri_framer_registers_t::cpri_cm_config

CPRI Control and Management Configuration.

2.13.3.4.2.1.9 volatile uint32_t cpri_framer_registers_t::cpri_cm_status

CPRI Control and Management Status.

2.13.3.4.2.1.10 volatile uint32_t cpri_framer_registers_t::cpri_rx_delay

CPRI Receive Delay.

2.13.3.4.2.1.11 volatile uint32_t cpri_framer_registers_t::cpri_round_delay

CPRI Round Trip Delay.

2.13.3.4.2.1.12 volatile uint32_t cpri_framer_registers_t::cpri_ex_delay_config

CPRI Extended Delay Measurement Configuration.

2.13.3.4.2.1.13 volatile uint32_t cpri_framer_registers_t::cpri_ex_delay_status

CPRI Extended Delay Measurement Status.

2.13.3.4.2.1.14 volatile uint32_t cpri_framer_registers_t::cpri_tx_prot_ver

CPRI Transmit Protocol Version.

2.13.3.4.2.1.15 volatile uint32_t cpri_framer_registers_t::cpri_tx_scr_seed

CPRI Transmit Scrambler Seed.

2.13.3.4.2.1.16 volatile uint32_t cpri_framer_registers_t::cpri_rx_scr_seed

CPRI Receive Scrambler Seed.

2.13.3.4.2.1.17 volatile uint32_t cpri_framer_registers_t::cpri_serdes_config

CPRI SerDes Interface Configuration.

2.13.3.4.2.1.18 volatile uint32_t cpri_framer_registers_t::cpri_map_config

CPRI Mapping Configuration.

2.13.3.4.2.1.19 volatile uint32_t cpri_framer_registers_t::cpri_map_cnt_config

CPRI Mapping Counter Configuration.

2.13.3.4.2.1.20 volatile uint32_t cpri_framer_registers_t::cpri_map_tbl_config

CPRI Mapping Table Configuration.

2.13.3.4.2.1.21 volatile uint32_t cpri_framer_registers_t::cpri_map_offset_rx

CPRI Mapping RX AxC Container Block Offset.

2.13.3.4.2.1.22 volatile uint32_t cpri_framer_registers_t::cpri_map_offset_tx

CPRI Mapping TX AxC Container Block Offset.

2.13.3.4.2.1.23 volatile uint32_t cpri_framer_registers_t::cpri_start_offset_tx

Offset for CPRI_TX_START Synchronization Output.

2.13.3.4.2.1.24 volatile uint32_t cpri_framer_registers_t::cpri_iq_rx_buf_status1

CPRI Mapping Buffer RX Status Register.

2.13.3.4.2.1.25 volatile uint32_t cpri_framer_registers_t::cpri_iq_rx_buf_status2

CPRI Mapping Buffer RX Status Register.

2.13.3.4.2.1.26 volatile uint32_t cpri_framer_registers_t::cpri_iq_tx_buf_status1

CPRI Mapping Buffer TX Status Register.

2.13.3.4.2.1.27 volatile uint32_t cpri_framer_registers_t::cpri_iq_tx_buf_status2

CPRI Mapping Buffer TX Status Register.

2.13.3.4.2.1.28 volatile uint32_t cpri_framer_registers_t::eth_rx_status

Ethernet Receive Status.

2.13.3.4.2.1.29 volatile uint32_t cpri_framer_registers_t::eth_config_1

Ethernet Feature Enable/Disable and Interrupt Enable Bits.

2.13.3.4.2.1.30 volatile uint32_t cpri_framer_registers_t::eth_config_2

Ethernet Miscellaneous Configuration.

2.13.3.4.2.1.31 volatile uint32_t cpri_framer_registers_t::eth_rx_control

Ethernet RX Packet Discard.

2.13.3.4.2.1.32 volatile uint32_t cpri_framer_registers_t::eth_rx_ex_status

Ethernet RX Additional Status.

2.13.3.4.2.1.33 volatile uint32_t cpri_framer_registers_t::eth_addr_msb

Ethernet MSB of MAC Address (16 bits)

2.13.3.4.2.1.34 volatile uint32_t cpri_framer_registers_t::eth_addr_lsb

Ethernet LSB of MAC Address (32 bits)

2.13.3.4.2.1.35 volatile uint32_t cpri_framer_registers_t::eth_hash_table

Ethernet Small 32 Entries Hash Table to Filter Multicast Traffic.

2.13.3.4.2.1.36 volatile uint32_t cpri_framer_registers_t::eth_config_3

Ethernet Configuration 3.

2.13.3.4.2.1.37 volatile uint32_t cpri_framer_registers_t::eth_cnt_rx_frame

Ethernet Receive Frame Counter.

2.13.3.4.2.1.38 volatile uint32_t cpri_framer_registers_t::eth_cnt_tx_frame

Ethernet Transmit Frame Counter.

2.13.3.4.2.1.39 volatile uint32_t cpri_framer_registers_t::hdlc_rx_status

HDLC Receive Status.

2.13.3.4.2.1.40 volatile uint32_t cpri_framer_registers_t::hdlc_config_1

HDLC Feature Enable/Disable and Interrupt Enable Bits.

2.13.3.4.2.1.41 volatile uint32_t cpri_framer_registers_t::hdlc_config_2

HDLC Miscellaneous Configuration.

2.13.3.4.2.1.42 volatile uint32_t cpri_framer_registers_t::hdlc_rx_control

HDLC RX Packet Discard.

2.13.3.4.2.1.43 volatile uint32_t cpri_framer_registers_t::hdlc_rx_ex_status

HDLC RX External Status.

2.13.3.4.2.1.44 volatile uint32_t cpri_framer_registers_t::hdlc_config_3

HDLC Configuration 3.

2.13.3.4.2.1.45 volatile uint32_t cpri_framer_registers_t::hdlc_cnt_rx_frame

HDLC Receive Frame Counter.

2.13.3.4.2.1.46 volatile uint32_t cpri_framer_registers_t::hdlc_cnt_tx_frame

HDLC Transmit Frame Counter.

2.13.3.4.2.2 struct cpri_complex_registers_t

CPRI complex/DMA registers.

In B4860 the MBUS is replaced by AXI.

Data Fields

- volatile uint32_t riqmts
- volatile uint32_t riqsdmts
- volatile uint32_t tiqmts
- volatile uint32_t rvssmts
- volatile uint32_t tvssmts
- volatile uint32_t riqsdba
- volatile uint32_t riqbs
- volatile uint32_t riqsdbs
- volatile uint32_t tiqbs
- volatile uint32_t rvssbs
- volatile uint32_t tvssbs
- volatile uint32_t rethbs
- volatile uint32_t rhdlcbs
- volatile uint32_t rvssba
- volatile uint32_t tvssba
- volatile uint32_t rebdrba
- volatile uint32_t tebdrba
- volatile uint32_t rhbdrba
- volatile uint32_t thbdrba
- volatile uint32_t rebdrs
- volatile uint32_t tebdrs
- volatile uint32_t rhbdrs
- volatile uint32_t thbdrs
- volatile uint32_t rgcm
- volatile uint32_t tgcm
- volatile uint32_t tscr
- volatile uint32_t tcfbs
- volatile uint32_t tctie1
- volatile uint32_t tctie2
- volatile uint32_t tmrc

- volatile uint32_t rfpw
- volatile uint32_t tfpw

2.13.3.4.2.2.1 Field Documentation

2.13.3.4.2.2.2 volatile uint32_t cpri_complex_registers_t::riqmts

Receive IQ MBus Transaction Size.

2.13.3.4.2.2.3 volatile uint32_t cpri_complex_registers_t::riqsdmts

Receive IQ Second Destination Mbus Transaction Size.

2.13.3.4.2.2.4 volatile uint32_t cpri_complex_registers_t::tiqmts

Transmit IQ MBus Transaction Size.

2.13.3.4.2.2.5 volatile uint32_t cpri_complex_registers_t::rvssmts

Receive VSS MBus Transaction Size.

2.13.3.4.2.2.6 volatile uint32_t cpri_complex_registers_t::tvssmts

Transmit VSS MBus Transaction Size.

2.13.3.4.2.2.7 volatile uint32_t cpri_complex_registers_t::riqsdba

Receive IQ Second Dest Base Address.

2.13.3.4.2.2.8 volatile uint32_t cpri_complex_registers_t::riqbs

Receive IQ Buffer Size.

2.13.3.4.2.2.9 volatile uint32_t cpri_complex_registers_t::riqsdbs

Receive IQ Second Destination Buffer Size.

2.13.3.4.2.2.10 volatile uint32_t cpri_complex_registers_t::tiqbs

Transmit IQ Buffer Size.

2.13.3.4.2.2.11 volatile uint32_t cpri_complex_registers_t::rvssbs

Receive VSS Buffer Size.

2.13.3.4.2.2.12 volatile uint32_t cpri_complex_registers_t::tvssbs

Transmit VSS Buffer Size.

2.13.3.4.2.2.13 volatile uint32_t cpri_complex_registers_t::rethbs

Receive ETH Buffer Size.

2.13.3.4.2.2.14 volatile uint32_t cpri_complex_registers_t::rhdlcbs

Receive HDLC Buffer Size.

2.13.3.4.2.2.15 volatile uint32_t cpri_complex_registers_t::rvssba

Receive VSS Buffer Base Address.

2.13.3.4.2.2.16 volatile uint32_t cpri_complex_registers_t::tvssba

Transmit VSS Buffer Base Address.

2.13.3.4.2.2.17 volatile uint32_t cpri_complex_registers_t::rebdrba

Receive Ethernet BD Ring Base Address.

2.13.3.4.2.2.18 volatile uint32_t cpri_complex_registers_t::tebdrba

Transmit Ethernet BD Ring Base Address.

2.13.3.4.2.2.19 volatile uint32_t cpri_complex_registers_t::rhbdrba

Receive HDLC BD Ring Base Address.

2.13.3.4.2.2.20 volatile uint32_t cpri_complex_registers_t::thbdrba

Transmit HDLC BD Ring Base Address.

2.13.3.4.2.2.21 volatile uint32_t cpri_complex_registers_t::rebdrs

Receive Ethernet Buffer Descriptor Ring Size.

2.13.3.4.2.2.22 volatile uint32_t cpri_complex_registers_t::tebdrs

Transmit Ethernet Buffer Descriptor Ring Size.

2.13.3.4.2.2.23 volatile uint32_t cpri_complex_registers_t::rhbdrs

Receive HDLC Buffer Descriptor Ring Size.

2.13.3.4.2.2.24 volatile uint32_t cpri_complex_registers_t::thbdrs

Transmit HDLC Buffer Descriptor Ring Size.

2.13.3.4.2.2.25 volatile uint32_t cpri_complex_registers_t::rgcm

Receive General CPRI Mode.

2.13.3.4.2.2.26 volatile uint32_t cpri_complex_registers_t::tgcm

Transmit General CPRI Mode.

2.13.3.4.2.2.27 volatile uint32_t cpri_complex_registers_t::tscr

Transmit Synchronization Configuration Register.

2.13.3.4.2.2.28 volatile uint32_t cpri_complex_registers_t::tcfbs

Transmit Cpri Framer Buffer Size.

2.13.3.4.2.2.29 volatile uint32_t cpri_complex_registers_t::tctie1

Tx control table insert enable 1.

2.13.3.4.2.2.30 volatile uint32_t cpri_complex_registers_t::tctie2

Tx control table insert enable 2.

2.13.3.4.2.2.31 volatile uint32_t cpri_complex_registers_t::tmrc

Timer Configuration.

2.13.3.4.2.2.32 volatile uint32_t cpri_complex_registers_t::rfpw

Receive Frame Pulse Width.

2.13.3.4.2.2.33 volatile uint32_t cpri_complex_registers_t::tfpw

Transmit Frame Pulse Width.

2.13.3.4.2.3 struct cpri_control_registers_t

CPRI control registers.

Data Fields

- volatile uint32_t rcr
- volatile uint32_t tcr
- volatile uint32_t raccr
- volatile uint32_t taccr
- volatile uint32_t rca
- volatile uint32_t rcd [CPRI_CONTROL_WORD_MAX_LENGTH_IN_UINT32]
- volatile uint32_t tca
- volatile uint32_t tcd [CPRI_CONTROL_WORD_MAX_LENGTH_IN_UINT32]
- volatile uint32_t riqft
- volatile uint32_t riqst
- volatile uint32_t riqt
- volatile uint32_t tiqft
- volatile uint32_t tiqst
- volatile uint32_t tiqt
- volatile uint32_t rvsst

- volatile uint32_t tvsst
- volatile uint32_t rethct
- volatile uint32_t tethct
- volatile uint32_t rcier
- volatile uint32_t tcier
- volatile uint32_t riqtsd
- volatile uint32_t eier
- volatile uint32_t tmre
- volatile uint32_t rewpr
- volatile uint32_t tewpr
- volatile uint32_t rhwpr
- volatile uint32_t thwpr
- volatile uint32_t racpr [CPRI_MAX_NUM_OF_ANTENNA_CARRIERS]
- volatile uint32_t tacpr [CPRI_MAX_NUM_OF_ANTENNA_CARRIERS]
- volatile uint32_t maskr [CPRI_NUM_OF_AUXILIARY_MASKING_REGISTERS]
- volatile uint32_t auxcr

2.13.3.4.2.3.1 Field Documentation

2.13.3.4.2.3.2 volatile uint32_t cpri_control_registers_t::rcr

Receive Control Register.

2.13.3.4.2.3.3 volatile uint32_t cpri_control_registers_t::tcr

Transmit Control Register.

2.13.3.4.2.3.4 volatile uint32_t cpri_control_registers_t::raccr

Receive AxC Control Register.

2.13.3.4.2.3.5 volatile uint32_t cpri_control_registers_t::taccr

Transmit AxC Control Register.

2.13.3.4.2.3.6 volatile uint32_t cpri_control_registers_t::rca

Receive Control Attribute.

2.13.3.4.2.3.7 volatile uint32_t cpri_control_registers_t::rcd[CPRI_CONTROL_WORD_MAX_LEN↩ GTH_IN_UINT32]

Receive Control Data 0 - 2.

2.13.3.4.2.3.8 volatile uint32_t cpri_control_registers_t::tca

Transmit Control Attribute.

2.13.3.4.2.3.9 volatile uint32_t cpri_control_registers_t::tcd[CPRI_CONTROL_WORD_MAX_LEN↩ GTH_IN_UINT32]

Transmit Control Data 0 - 2.

2.13.3.4.2.3.10 volatile uint32_t cpri_control_registers_t::riqft

Receive IQ First Threshold.

2.13.3.4.2.3.11 volatile uint32_t cpri_control_registers_t::riqst

Receive IQ Second Threshold.

2.13.3.4.2.3.12 volatile uint32_t cpri_control_registers_t::riqt

Receive IQ Threshold.

2.13.3.4.2.3.13 volatile uint32_t cpri_control_registers_t::tiqft

Transmit IQ First Threshold.

2.13.3.4.2.3.14 volatile uint32_t cpri_control_registers_t::tiqst

Transmit IQ Second Threshold.

2.13.3.4.2.3.15 volatile uint32_t cpri_control_registers_t::tiqt

Transmit IQ Threshold.

2.13.3.4.2.3.16 volatile uint32_t cpri_control_registers_t::rvsst

Receive VSS Threshold.

2.13.3.4.2.3.17 volatile uint32_t cpri_control_registers_t::tvsst

Transmit VSS Threshold.

2.13.3.4.2.3.18 volatile uint32_t cpri_control_registers_t::rethct

Receive Ethernet Coalescing Threshold.

2.13.3.4.2.3.19 volatile uint32_t cpri_control_registers_t::tethct

Transmit Ethernet Coalescing Threshold.

2.13.3.4.2.3.20 volatile uint32_t cpri_control_registers_t::rcier

CPRI Receive Control and Timing interrupt Enable Register.

2.13.3.4.2.3.21 volatile uint32_t cpri_control_registers_t::tcier

CPRI transmit Control and Timing Interrupt Enable Register.

2.13.3.4.2.3.22 volatile uint32_t cpri_control_registers_t::riqtsd

Receive IQ Threshold Second Destination.

2.13.3.4.2.3.23 volatile uint32_t cpri_control_registers_t::eier

CPRI Error Interrupt Enable Register.

2.13.3.4.2.3.24 volatile uint32_t cpri_control_registers_t::tmre

Timer Enable Register.

2.13.3.4.2.3.25 volatile uint32_t cpri_control_registers_t::rewpr

Receive Ethernet Write Pointer Ring.

2.13.3.4.2.3.26 volatile uint32_t cpri_control_registers_t::tewpr

Transmit Ethernet Write Pointer Ring.

2.13.3.4.2.3.27 volatile uint32_t cpri_control_registers_t::rhwpr

Receive HDLC Write Pointer Ring.

2.13.3.4.2.3.28 volatile uint32_t cpri_control_registers_t::thwpr

Transmit HDLC Write Pointer Ring.

2.13.3.4.2.3.29 volatile uint32_t cpri_control_registers_t::racpr[CPRI_MAX_NUM_OF_ANTE↩ NNA_CARRIERS]

Receive AxCn Parameter Register.

2.13.3.4.2.3.30 volatile uint32_t cpri_control_registers_t::tacpr[CPRI_MAX_NUM_OF_ANTEN↩ NA_CARRIERS]

Transmit AxCn Parameter Register.

2.13.3.4.2.3.31 volatile uint32_t cpri_control_registers_t::maskr[CPRI_NUM_OF_AUXILIARY_M↩ ASKING_REGISTERS]

cpri auxillary intetbface mask registers

2.13.3.4.2.3.32 volatile uint32_t cpri_control_registers_t::auxcr

CPRI Auxillary control register.

2.13.3.4.2.4 struct cpri_status_registers_t

CPRI status registers.

Data Fields

- volatile uint32_t riqbdr
- volatile uint32_t riqsdbdr

- volatile uint32_t tiqbdr
- volatile uint32_t rccr
- volatile uint32_t rvssbdr
- volatile uint32_t tvssbdr
- volatile uint32_t rethbd1
- volatile uint32_t rethbd2
- volatile uint32_t tethbd1
- volatile uint32_t tethbd2
- volatile uint32_t rerpr
- volatile uint32_t terpr
- volatile uint32_t rhdlcbd1
- volatile uint32_t rhdlcbd2
- volatile uint32_t thdlcbd1
- volatile uint32_t thdlcbd2
- volatile uint32_t rhrpr
- volatile uint32_t thrpr
- volatile uint32_t rer
- volatile uint32_t ter
- volatile uint32_t eer
- volatile uint32_t rethcs
- volatile uint32_t tethcs
- volatile uint32_t tmrsr
- volatile uint32_t rsr
- volatile uint32_t tsr

2.13.3.4.2.4.1 Field Documentation

2.13.3.4.2.4.2 volatile uint32_t cpri_status_registers_t::riqbdr

Receive IQ Buffer Displacement Register.

2.13.3.4.2.4.3 volatile uint32_t cpri_status_registers_t::riqsdbdr

Receive IQ Buffer Second Destination Displacement Register.

2.13.3.4.2.4.4 volatile uint32_t cpri_status_registers_t::tiqbdr

Transmit IQ Buffer Displacement Register.

2.13.3.4.2.4.5 volatile uint32_t cpri_status_registers_t::rccr

receive Chips Counter Register

2.13.3.4.2.4.6 volatile uint32_t cpri_status_registers_t::rvssbdr

Receive VSS Buffer Displacement Register.

2.13.3.4.2.4.7 volatile uint32_t cpri_status_registers_t::tvssbdr

Transmit VSS Buffer Displacement Register.

2.13.3.4.2.4.8 volatile uint32_t cpri_status_registers_t::rethbd1

Receive Ethernet Buffer Descriptor.

2.13.3.4.2.4.9 volatile uint32_t cpri_status_registers_t::rethbd2

Receive Ethernet Buffer Descriptor.

2.13.3.4.2.4.10 volatile uint32_t cpri_status_registers_t::tethbd1

Transmit Ethernet Buffer Descriptor.

2.13.3.4.2.4.11 volatile uint32_t cpri_status_registers_t::tethbd2

Transmit Ethernet Buffer Descriptor.

2.13.3.4.2.4.12 volatile uint32_t cpri_status_registers_t::rerpr

Receive Ethernet Read Pointer Ring.

2.13.3.4.2.4.13 volatile uint32_t cpri_status_registers_t::terpr

Transmit Ethernet Read Pointer Ring.

2.13.3.4.2.4.14 volatile uint32_t cpri_status_registers_t::rhdlcbd1

Receive HDLC Buffer Descriptor.

2.13.3.4.2.4.15 volatile uint32_t cpri_status_registers_t::rhdlcbd2

Receive HDLC Buffer Descriptor.

2.13.3.4.2.4.16 volatile uint32_t cpri_status_registers_t::thdlcbd1

Transmit HDLC Buffer Descriptor.

2.13.3.4.2.4.17 volatile uint32_t cpri_status_registers_t::thdlcbd2

Transmit HDLC Buffer Descriptor.

2.13.3.4.2.4.18 volatile uint32_t cpri_status_registers_t::rhrpr

Receive HDLC Read Pointer Ring.

2.13.3.4.2.4.19 volatile uint32_t cpri_status_registers_t::thrpr

Transmit HDLC Read Pointer Ring.

2.13.3.4.2.4.20 volatile uint32_t cpri_status_registers_t::rer

Receive Event Register.

2.13.3.4.2.4.21 volatile uint32_t cpri_status_registers_t::ter

Transmit Event Register.

2.13.3.4.2.4.22 volatile uint32_t cpri_status_registers_t::eer

Error Event Register.

2.13.3.4.2.4.23 volatile uint32_t cpri_status_registers_t::rethcs

Receive Ethernet Coalescing Status.

2.13.3.4.2.4.24 volatile uint32_t cpri_status_registers_t::tethcs

Transmit Ethernet Coalescing Status.

2.13.3.4.2.4.25 volatile uint32_t cpri_status_registers_t::tmrsr

Timer Status Register.

2.13.3.4.2.4.26 volatile uint32_t cpri_status_registers_t::rsr

Receive Status Register.

2.13.3.4.2.4.27 volatile uint32_t cpri_status_registers_t::tsr

Transmit Status Register.

2.13.3.4.2.5 struct cpri_configuration_memories_t

CPRI configuration memories registers.

Data Fields

- volatile uint32_t rcm [CPRI_NUM_OF_CONFIGURATION_MEMORY_REGISTERS]
- volatile uint32_t tcm [CPRI_NUM_OF_CONFIGURATION_MEMORY_REGISTERS]

2.13.3.4.2.5.1 Field Documentation

2.13.3.4.2.5.2 volatile uint32_t cpri_configuration_memories_t::rcm[CPRI_NUM_OF_CONFIGURATION_MEMORY_REGISTERS]

Receive Configuration Memory.

2.13.3.4.2.5.3 volatile uint32_t cpri_configuration_memories_t::tcm[CPRI_NUM_OF_CONFIGURATION_MEMORY_REGISTERS]

Transmit Configuration Memory.

2.13.3.4.2.6 struct cpri_general_registers_t

CPRI general registers.

Data Fields

- volatile uint32_t ccr
- volatile uint32_t ICR [CPRI_NUM_OF_INTERRUPT_CONTROL_REGISTERS]
- volatile uint32_t rccier
- volatile uint32_t tccier
- volatile uint32_t grsr
- volatile uint32_t gtsr
- volatile uint32_t iesr
- volatile uint32_t cpridmastop
- volatile uint32_t cpridmastopst
- volatile uint32_t cprirrocr

2.13.3.4.2.6.1 Field Documentation

2.13.3.4.2.6.2 volatile uint32_t cpri_general_registers_t::ccr

CPRI Clocks Control Register.

2.13.3.4.2.6.3 volatile uint32_t cpri_general_registers_t::ICR[CPRI_NUM_OF_INTERRUPT_CON↩ TROL_REGISTERS]

CPRI Interrupt Control Register.

2.13.3.4.2.6.4 volatile uint32_t cpri_general_registers_t::rccier

CPRI Receive CPU Control Interrupt Enable Register.

2.13.3.4.2.6.5 volatile uint32_t cpri_general_registers_t::tccier

CPRI Transmit CPU Control Interrupt Enable Register.

2.13.3.4.2.6.6 volatile uint32_t cpri_general_registers_t::grsr

General Receive Synchronization Register.

2.13.3.4.2.6.7 volatile uint32_t cpri_general_registers_t::gtsr

General Transmit Synchronization Register.

2.13.3.4.2.6.8 volatile uint32_t cpri_general_registers_t::iesr

CPRI Error Status Register.

2.13.3.4.2.6.9 volatile uint32_t cpri_general_registers_t::cpridmastop

CPRI DMA Stop Register.

2.13.3.4.2.6.10 volatile uint32_t cpri_general_registers_t::cpridmastopst

CPRI DMA Stop Status Register.

2.13.3.4.2.6.11 volatile uint32_t cpri_general_registers_t::cprirrocr

CPRI Reset Request Output Control Register.

2.13.3.4.2.7 struct cpri_unit_map_t

CPRI peripheral registers.

Data Fields

- cpri_framer_registers_t cpri_framer
- cpri_complex_registers_t cpri_complex
- cpri_control_registers_t cpri_control
- cpri_status_registers_t cpri_status
- cpri_configuration_memories_t cpri_configuration_memories
- cpri_general_registers_t cpri_general_registers

2.13.3.4.2.7.1 Field Documentation

2.13.3.4.2.7.2 cpri_framer_registers_t cpri_unit_map_t::cpri_framer

CPRI framer registers.

2.13.3.4.2.7.3 cpri_complex_registers_t cpri_unit_map_t::cpri_complex

CPRI complex registers.

2.13.3.4.2.7.4 cpri_control_registers_t cpri_unit_map_t::cpri_control

CPRI control registers.

2.13.3.4.2.7.5 cpri_status_registers_t cpri_unit_map_t::cpri_status

CPRI status registers.

2.13.3.4.2.7.6 cpri_configuration_memories_t cpri_unit_map_t::cpri_configuration_memories

CPRI configuration memories registers.

2.13.3.4.2.7.7 cpri_general_registers_t cpri_unit_map_t::cpri_general_registers

CPRI general registers; valid only in CPRI1 unit.

2.13.3.5 I2C Memory Map

2.13.3.5.1 Overview

Definition of I2C registers and memory map

Data Structures

- struct i2c_map_t

2.13.3.5.2 Data Structure Documentation

2.13.3.5.2.1 struct i2c_map_t

I2C Registers.

Data Fields

- volatile uint8_t i2cadr
- volatile uint8_t i2cfdr
- volatile uint8_t i2ccr
- volatile uint8_t i2csr
- volatile uint8_t i2cdr
- volatile uint8_t i2cdfsrr

2.13.3.5.2.1.1 Field Documentation

2.13.3.5.2.1.2 volatile uint8_t i2c_map_t::i2cadr

I2C address register.

2.13.3.5.2.1.3 volatile uint8_t i2c_map_t::i2cfdr

I2C frequesncy divider register.

2.13.3.5.2.1.4 volatile uint8_t i2c_map_t::i2ccr

I2C control register.

2.13.3.5.2.1.5 volatile uint8_t i2c_map_t::i2csr

I2C status register.

2.13.3.5.2.1.6 volatile uint8_t i2c_map_t::i2cdr

I2C data register.

2.13.3.5.2.1.7 volatile uint8_t i2c_map_t::i2cdfsrr

I2C digital filter sampling rate register.

2.13.3.6 Maple-B3 Memory Map

2.13.3.6.1 Overview

Data Structures

- struct maple_pe_bd_params_t
- struct maple_pe_bd_ring_params_t
- struct maple_pe_bd_steering_t
- struct maple_pe_bd_ring_steering_t
- struct maple_pe_mpisr_tid_t
- union ftpedss_t
- struct maple_mpisr_p_t
- struct maple_lte_pram_map_t
- struct maple_wcdma_pram_map_t
- struct maple_lte_psif_dram_map_t
- struct maple_wcdma_psif_dram_map_t
- struct maple_tvpe_map_t
- struct maple_ftpe_map_t
- struct maple_depe_map_t
- struct maple_eqpe_map_t
- struct maple_pdpe_map_t
- struct maple_lte_mbus_memmap_t
- struct maple_wcdma_mbus_memmap_t
- struct maple_lte_sbus_memmap_t

Macros

- #define PRAM_BD_RING_SIZE (0x18000 - 0x10000)
- #define MAPLE_TRACE_BUFFER_SIZE (0x18000 - 0x17000)
- #define FTPE_DSS_NUM 7
- #define TVPE_VITERBI_POLY_SETS_NUM 3
- #define PUFFT_SOC_TIMER_SETS 4
- #define PDSCH_SOC_TIMER_SETS 16
- #define MAPLE3LW_FTPE_NUM 3
- #define MAPLE3W_FTPE_NUM 2
- #define MAPLE_STP_NUM 4
- #define MAPLE_NUM_SLOTS 2
- #define MAPLE_NUM_SECTORS 4
- #define MAPLE_NUM_ANT 8

Typedefs

- typedef maple_lte_sbus_memmap_t maple_wcdma_sbus_memmap_t

Number of MAPLE BDs per Priority

- #define MAPLE_MAX_NUM_PE_BD_RINGS_LOW_PRIORITY 8
- #define MAPLE_MAX_NUM_PE_BD_RINGS_HIGH_PRIORITY 8
- #define MAPLE_PE_MAX_NUM_BD_RINGS (MAPLE_MAX_NUM_PE_BD_RINGS_LOW↩ _PRIORITY + MAPLE_MAX_NUM_PE_BD_RINGS_HIGH_PRIORITY)

2.13.3.6.2 Data Structure Documentation

2.13.3.6.2.1 struct maple_pe_bd_params_t

BD ring parameter A and parameter B.

Data Fields

- volatile uint32_t brpap
- volatile uint32_t brpbp

2.13.3.6.2.1.1 Field Documentation

2.13.3.6.2.1.2 volatile uint32_t maple_pe_bd_params_t::brpap

BD Ring X Priority A Parameter.

2.13.3.6.2.1.3 volatile uint32_t maple_pe_bd_params_t::brpbp

BD Ring X Priority B Parameter.

2.13.3.6.2.2 struct maple_pe_bd_ring_params_t

PE BD rings parameters.

Data Fields

- maple_pe_bd_params_t high_bd [MAPLE_MAX_NUM_PE_BD_RINGS_HIGH_PRIORITY]
- maple_pe_bd_params_t low_bd [MAPLE_MAX_NUM_PE_BD_RINGS_LOW_PRIORITY]

2.13.3.6.2.2.1 Field Documentation

2.13.3.6.2.2.2 maple_pe_bd_params_t maple_pe_bd_ring_params_t::high_bd[MAPLE_MAX_↩ NUM_PE_BD_RINGS_HIGH_PRIORITY]

High priority rings.

2.13.3.6.2.2.3 maple_pe_bd_params_t maple_pe_bd_ring_params_t::low_bd[MAPLE_MAX_N↩ UM_PE_BD_RINGS_LOW_PRIORITY]

Low priority rings.

2.13.3.6.2.3 struct maple_pe_bd_steering_t

PE BD ring steering bits parameters.

Data Fields

- volatile uint32_t tidstp [MAPLE_STP_NUM]

2.13.3.6.2.3.1 Field Documentation

2.13.3.6.2.3.2 volatile uint32_t maple_pe_bd_steering_t::tidstp[MAPLE_STP_NUM]

Task ID and Steering Parameter for BD ring.

2.13.3.6.2.4 struct maple_pe_bd_ring_steering_t

PE steering bits parameters.

Data Fields

- maple_pe_bd_steering_t high_bd [MAPLE_MAX_NUM_PE_BD_RINGS_HIGH_PRIORITY]
- maple_pe_bd_steering_t low_bd [MAPLE_MAX_NUM_PE_BD_RINGS_LOW_PRIORITY]

2.13.3.6.2.4.1 Field Documentation

2.13.3.6.2.4.2 maple_pe_bd_steering_t maple_pe_bd_ring_steering_t::high_bd[MAPLE_MAX_↩ NUM_PE_BD_RINGS_HIGH_PRIORITY]

Steering Parameters for high BD ring.

2.13.3.6.2.4.3 maple_pe_bd_steering_t maple_pe_bd_ring_steering_t::low_bd[MAPLE_MAX_↩ NUM_PE_BD_RINGS_LOW_PRIORITY]

Steering Parameters for low BD ring.

2.13.3.6.2.5 struct maple_pe_mpisr_tid_t

MAPLE Interrupt Service Routine <x> MMU TASK ID.

2.13.3.6.2.6 union ftpedss_t

FTPE data size sets parameters.

2.13.3.6.2.6.1 Field Documentation

2.13.3.6.2.6.2 volatile uint32_t ftpedss_t::ftpedsspy[3]

FTPE Data Size Set x Parameter y.

2.13.3.6.2.7 struct maple_mpisr_p_t

MAPLE Interrupt Service Routine parameters.

Data Fields

- volatile uint32_t mpisr_ack_address
- volatile uint32_t mpisr_ack_value
- volatile uint32_t all_tsk_link

2.13.3.6.2.7.1 Field Documentation

2.13.3.6.2.7.2 volatile uint32_t maple_mpisr_p_t::mpisr_ack_address

The address to generate the write acknowledge towards the soc in order to de-assert the interrupt.

2.13.3.6.2.7.3 volatile uint32_t maple_mpisr_p_t::mpisr_ack_value

The data to generate the write acknowledge towards the soc in order to de-assert the interrupt.

2.13.3.6.2.7.4 volatile uint32_t maple_mpisr_p_t::all_tsk_link

Array of bits which Indicates the active tasks that MPISR should be schedule every time MPISRx is asserted.

2.13.3.6.2.8 struct maple_lte_pram_map_t

Maple LTE PRAM memory map.

Data Fields

- volatile uint32_t mbdrcp0
- volatile uint32_t mbdrcp1
- volatile uint32_t mbdrcp2
- volatile uint32_t mucvp
- volatile uint32_t mp_tpp
- volatile uint32_t mcgcp
- volatile uint32_t mmc0p
- volatile uint32_t mmc1p
- volatile uint32_t mtvcp
- volatile uint32_t cdomcp
- volatile uint32_t cubmsp
- volatile uint32_t mpdschcp
- volatile uint32_t mpuschcp
- volatile uint32_t mcrrcip
- maple_pe_bd_ring_params_t tvpe_bd
- maple_pe_bd_ring_params_t ftpe_bd [MAPLE3LW_FTPE_NUM]
- maple_pe_bd_ring_params_t depe_bd
- maple_pe_bd_ring_params_t crcpe_bd
- maple_pe_bd_ring_params_t eqpe_bd
- maple_pe_bd_ring_params_t convpe_bd
- maple_pe_bd_ring_params_t pusch_bd
- maple_pe_bd_ring_params_t pdsch_bd
- maple_pe_bd_ring_params_t pufft_bd
- maple_pe_bd_ring_steering_t tvpe_steering
- maple_pe_bd_ring_steering_t ftpe_steering [MAPLE3LW_FTPE_NUM]
- maple_pe_bd_ring_steering_t depe_steering
- maple_pe_bd_ring_steering_t crcpe_steering
- maple_pe_bd_ring_steering_t eqpe_steering
- maple_pe_bd_ring_steering_t convpe_steering
- maple_pe_bd_ring_steering_t pusch_steering
- maple_pe_bd_steering_t pdsch_high_bd_steering [4]
- maple_pe_mpisr_tid_t mpisr_tid
- maple_pe_bd_steering_t pdsch_low_bd_steering [4]

- maple_pe_bd_ring_steering_t pufft_steering
- struct {
    volatile uint32_t mtvpvhcp
    volatile uint32_t mtvpvlcp
  } tvpe_puncture_vector [MAPLE_NUM_TVPE_PUNCTURING_VECTORS]

- union {
    volatile uint32_t mtvppcp [3]
  } tvpe_puncture_period

- struct {
    struct {
    } mtvpvsc0p
    struct {
    } mtvpvsc1p
  } mtvpvscp [TVPE_VITERBI_POLY_SETS_NUM]

- volatile uint32_t mpdsch_bf_config [8][MAPLE_NUM_SECTORS]
- ftpedss_t ftpedss
- struct {
    volatile uint32_t uprmbpp
    volatile uint32_t upsmbpp
    volatile uint32_t ubsp
  } multiply_ftpe [MAPLE3LW_FTPE_NUM]

- volatile uint32_t ftpecubrp
- volatile uint8_t mpdsch_ant_config_2nd_set [MAPLE_NUM_SECTORS][16]
- volatile uint32_t mtrcecp
- volatile uint32_t mtrcwpp
- volatile uint32_t mpufticp [MAPLE_NUM_SECTORS]
- volatile uint32_t mpufftpmtap [MAPLE_NUM_SECTORS]
- volatile uint16_t mpufftfcbr [MAPLE_NUM_SECTORS]
- volatile uint16_t mpufftfcbi [MAPLE_NUM_SECTORS]
- volatile uint32_t mpufftfcsr [MAPLE_NUM_SECTORS]
- volatile uint32_t mpufftfcsi [MAPLE_NUM_SECTORS]
- volatile uint32_t mpufftpstmtap [MAPLE_NUM_SECTORS]
- volatile uint32_t mpdsch_pad_data [4]
- volatile uint32_t mpdsch_pad_data1 [4]
- volatile uint32_t mpdsch_pad_data2 [4]
- volatile uint32_t mpdsch_sec_config [MAPLE_NUM_SECTORS]
- volatile uint8_t mpdsch_ant_config [MAPLE_NUM_SECTORS][16]
- volatile uint32_t mpdsch_cell_config [MAPLE_NUM_SECTORS]
- volatile uint32_t mpdsch_sec_config_2nd_set [MAPLE_NUM_SECTORS]
- volatile uint32_t mpdsch_cell_config_2nd_set [MAPLE_NUM_SECTORS]
- volatile uint32_t mpdsch_start_qse0p [8]
- volatile uint32_t mpdsch_pdpe_qse0p [8]
- volatile uint32_t mpdsch_done_qse0p [8]
- volatile uint32_t mpdsch_start_qse1p [8]
- volatile uint32_t mpdsch_pdpe_qse1p [8]
- volatile uint32_t mpdsch_done_qse1p [8]
- volatile uint32_t mpdsch_start_qse2p [8]

- volatile uint32_t mpdsch_pdpe_qse2p [8]
- volatile uint32_t mpdsch_done_qse2p [8]
- volatile uint32_t mpdsch_start_qse3p [8]
- volatile uint32_t mpdsch_pdpe_qse3p [8]
- volatile uint32_t mpdsch_done_qse3p [8]
- volatile uint32_t mpdsch_start_qs0wp
- volatile uint32_t mpdsch_pdpe_qs0wp
- volatile uint32_t mpdsch_done_qs0wp
- volatile uint32_t mpdsch_start_qs1wp
- volatile uint32_t mpdsch_pdpe_qs1wp
- volatile uint32_t mpdsch_done_qs1wp
- volatile uint32_t mpdsch_start_qs2wp
- volatile uint32_t mpdsch_pdpe_qs2wp
- volatile uint32_t mpdsch_done_qs2wp
- volatile uint32_t mpdsch_start_qs3wp
- volatile uint32_t mpdsch_pdpe_qs3wp
- volatile uint32_t mpdsch_done_qs3wp
- volatile uint32_t mpdsch_start_qs0rp
- volatile uint32_t mpdsch_pdpe_qs0rp
- volatile uint32_t mpdsch_done_qs0rp
- volatile uint32_t mpdsch_start_qs1rp
- volatile uint32_t mpdsch_pdpe_qs1rp
- volatile uint32_t mpdsch_done_qs1rp
- volatile uint32_t mpdsch_start_qs2rp
- volatile uint32_t mpdsch_pdpe_qs2rp
- volatile uint32_t mpdsch_done_qs2rp
- volatile uint32_t mpdsch_start_qs3rp
- volatile uint32_t mpdsch_pdpe_qs3rp
- volatile uint32_t mpdsch_done_qs3rp
- volatile uint32_t mpdsch_s0np [7]
- volatile uint32_t mpdsch_s1np [7]
- volatile uint32_t mpdsch_s2np [7]
- volatile uint32_t mpdsch_s3np [7]
- volatile uint32_t maple_xxpe_acc [14]
- volatile uint32_t mpdsch_bf_config_2nd_set [8][MAPLE_NUM_SECTORS]
- volatile uint32_t mpdschoba_2nd_set [MAPLE_NUM_SECTORS][MAPLE_NUM_ANT]
- volatile uint32_t mpdschobs_2nd_set [MAPLE_NUM_SECTORS]
- volatile uint32_t mpdsch_ack [PDSCH_SOC_TIMER_SETS][2]
- volatile uint32_t mpdschoba [MAPLE_NUM_SECTORS][MAPLE_NUM_ANT]
- volatile uint32_t mpdschobs [MAPLE_NUM_SECTORS]
- volatile uint32_t mpufftiba [MAPLE_NUM_SECTORS][MAPLE_NUM_ANT]
- volatile uint32_t mpufftibs [MAPLE_NUM_SECTORS]
- volatile uint8_t pper [MAPLE_NUM_TVPE_PUNCTURING_VECTORS]

2.13.3.6.2.8.1 Field Documentation

2.13.3.6.2.8.2 volatile uint32_t maple_lte_pram_map_t::mbdrcp0

Maple BD Rings Configuration Parameter 0.

2.13.3.6.2.8.3 volatile uint32_t maple_lte_pram_map_t::mbdrcp1

Maple BD Rings Configuration Parameter 1.

2.13.3.6.2.8.4 volatile uint32_t maple_lte_pram_map_t::mbdrcp2

Maple BD Rings Configuration Parameter 2.

2.13.3.6.2.8.5 volatile uint32_t maple_lte_pram_map_t::mucvp

Maple UCode Version Parameter.

2.13.3.6.2.8.6 volatile uint32_t maple_lte_pram_map_t::mp_tpp

Maple Timer Period Parameter.
This option is not available through SmartDSP OS.

2.13.3.6.2.8.7 volatile uint32_t maple_lte_pram_map_t::mcgcp

This parameter allows controlling (disabling/enabling) the input clocks for the MAPLE-B3W internal PEs.

2.13.3.6.2.8.8 volatile uint32_t maple_lte_pram_map_t::mmc0p

Maple Mode Configurations 0 Parameter.

2.13.3.6.2.8.9 volatile uint32_t maple_lte_pram_map_t::mmc1p

Maple Mode Configurations 1 Parameter.

2.13.3.6.2.8.10 volatile uint32_t maple_lte_pram_map_t::mtvcp

MAPLE eTVPE Configuration parameter.

2.13.3.6.2.8.11 volatile uint32_t maple_lte_pram_map_t::cdomcp

CRPE-DL Output Mode Configuration Parameter.

2.13.3.6.2.8.12 volatile uint32_t maple_lte_pram_map_t::cubmsp

CRPE-ULB Mode Configuration Parameter.

2.13.3.6.2.8.13 volatile uint32_t maple_lte_pram_map_t::mpdschcp

PDSCH EDF Configuration Parameter.

2.13.3.6.2.8.14 volatile uint32_t maple_lte_pram_map_t::mpuschcp

PUSCH EDF Configuration Parameter.

2.13.3.6.2.8.15 volatile uint32_t maple_lte_pram_map_t::mcrrcip

Maple CRPE Reset Completion Indication Parameter.

2.13.3.6.2.8.16 maple_pe_bd_ring_params_t maple_lte_pram_map_t::tvpe_bd

TVPE BD Ring Priority Configuration.

2.13.3.6.2.8.17 maple_pe_bd_ring_params_t maple_lte_pram_map_t::ftpe_bd[MAPLE3LW_FTPE_NUM]

FFTPE_x BD Ring Priority Configuration.

2.13.3.6.2.8.18 maple_pe_bd_ring_params_t maple_lte_pram_map_t::depe_bd

DEPE BD Ring Priority Configuration.

2.13.3.6.2.8.19 maple_pe_bd_ring_params_t maple_lte_pram_map_t::crcpe_bd

CRCPE BD Ring Priority Configuration.

2.13.3.6.2.8.20 maple_pe_bd_ring_params_t maple_lte_pram_map_t::eqpe_bd

EQPE BD Ring Priority Configuration.

2.13.3.6.2.8.21 maple_pe_bd_ring_params_t maple_lte_pram_map_t::convpe_bd

CONVPE BD Ring Priority Configuration.

2.13.3.6.2.8.22 maple_pe_bd_ring_params_t maple_lte_pram_map_t::pusch_bd

PUSCH BD Ring Priority Configuration.

2.13.3.6.2.8.23 maple_pe_bd_ring_params_t maple_lte_pram_map_t::pdsch_bd

PDSCH BD Ring Priority Configuration.

2.13.3.6.2.8.24 maple_pe_bd_ring_params_t maple_lte_pram_map_t::pufft_bd

PUFFT BD Ring Priority Configuration.

2.13.3.6.2.8.25 maple_pe_bd_ring_steering_t maple_lte_pram_map_t::tvpe_steering

Task ID and Steering Parameters.
TVPE Task ID and Steering Parameters

2.13.3.6.2.8.26 maple_pe_bd_ring_steering_t maple_lte_pram_map_t::ftpe_steering[MAPLE3LW_FTPE_NUM]

FTPE Task ID and Steering Parameters.

2.13.3.6.2.8.27 maple_pe_bd_ring_steering_t maple_lte_pram_map_t::depe_steering

DEPE Task ID and Steering Parameters.

2.13.3.6.2.8.28 maple_pe_bd_ring_steering_t maple_lte_pram_map_t::crcpe_steering

CRCPE Task ID and Steering Parameters.

2.13.3.6.2.8.29 maple_pe_bd_ring_steering_t maple_lte_pram_map_t::eqpe_steering

EQPE Task ID and Steering Parameters.

2.13.3.6.2.8.30 maple_pe_bd_ring_steering_t maple_lte_pram_map_t::convpe_steering

CONVPE Task ID and Steering Parameters.

2.13.3.6.2.8.31 maple_pe_bd_ring_steering_t maple_lte_pram_map_t::pusch_steering

PUSCH Task ID and Steering Parameters.

2.13.3.6.2.8.32 maple_pe_bd_steering_t maple_lte_pram_map_t::pdsch_high_bd_steering[4]

PDSCH Task ID and Steering Parameters, for high priority BD.

2.13.3.6.2.8.33 maple_pe_mpisr_tid_t maple_lte_pram_map_t::mpisr_tid

MMU TASK ID in use by the MPISRx.

2.13.3.6.2.8.34 maple_pe_bd_steering_t maple_lte_pram_map_t::pdsch_low_bd_steering[4]

PDSCH Task ID and Steering Parameters, for low priority BD.

2.13.3.6.2.8.35 maple_pe_bd_ring_steering_t maple_lte_pram_map_t::pufft_steering

PUFFT Task ID and Steering Parameters.

2.13.3.6.2.8.36 volatile uint32_t maple_lte_pram_map_t::mtvpvhcp

Maple Turbo Viterbi Puncturing Vector x High Configuration Parameter.

2.13.3.6.2.8.37 volatile uint32_t maple_lte_pram_map_t::mtvpvlcp

Maple Turbo Viterbi Puncturing Vector x Low Configuration Parameter.

2.13.3.6.2.8.38 struct { ... } maple_lte_pram_map_t::tvpe_puncture_vector[MAPLE_NUM_TVPE_PUNCTURING_VECTORS]

Maple TVPE Puncturing Vector parameters.

2.13.3.6.2.8.39 volatile uint8_t maple_lte_pram_map_t::pper[MAPLE_NUM_TVPE_PUNCTURING_VECTORS]

Maple Turbo Viterbi Puncturing Period.

2.13.3.6.2.8.40 volatile uint32_t maple_lte_pram_map_t::mtvppcp[3]

Maple Turbo Viterbi Puncturing Period Configuration y Parameter.

2.13.3.6.2.8.41 union { ... } maple_lte_pram_map_t::tvpe_puncture_period

Maple TVPE Puncturing Period parameters.

2.13.3.6.2.8.42 struct { ... } maple_lte_pram_map_t::mtvpvsc0p

Maple Turbo Viterbi Polynomial Vector Set x Configuration 0 parameter.

2.13.3.6.2.8.43 struct { ... } maple_lte_pram_map_t::mtvpvsc1p

Maple Turbo Viterbi Polynomial Vector Set x Configuration 1 parameter.

2.13.3.6.2.8.44 struct { ... } maple_lte_pram_map_t::mtvpvscp[TVPE_VITERBI_POLY_SETS_NUM]

Maple Tvpe Viterbi Polynomial Sets.

2.13.3.6.2.8.45 volatile uint32_t maple_lte_pram_map_t::mpdsch_bf_config[8][MAPLE_NUM_SECTORS]

Beam Forming Configuration Parameter - 8 register per sector.

2.13.3.6.2.8.46 ftpedss_t maple_lte_pram_map_t::ftpedss

Maple FTPE Data Size parameters.

2.13.3.6.2.8.47 volatile uint32_t maple_lte_pram_map_t::uprmbpp

Update Pre-Multiplication Buffer Pointer.

2.13.3.6.2.8.48 volatile uint32_t maple_lte_pram_map_t::upsmbpp

Update Post-Multiplication Buffer Pointer.

2.13.3.6.2.8.49 volatile uint32_t maple_lte_pram_map_t::ubsp

Update Buffers Size Parameter.

2.13.3.6.2.8.50 struct { ... } maple_lte_pram_map_t::multiply_ftpe[MAPLE3LW_FTPE_NUM]

eFTPE x Update Post/Pre-Multiplication parameters

2.13.3.6.2.8.51 volatile uint32_t maple_lte_pram_map_t::ftpecubrp

UBCUpdate Buffers Complete indication.

2.13.3.6.2.8.52 volatile uint8_t maple_lte_pram_map_t::mpdsch_ant_config_2nd_set[MAPLE_NUM_SECTORS][16]

PDSCH EDF Second Set of Target Antenna API Configuration Parameters.

2.13.3.6.2.8.53 volatile uint32_t maple_lte_pram_map_t::mtrcecp

MAPLE Trace Enable Configuration Register.

2.13.3.6.2.8.54 volatile uint32_t maple_lte_pram_map_t::mtrcwpp

MAPLE Trace Write Pointer Parameter.

2.13.3.6.2.8.55 volatile uint32_t maple_lte_pram_map_t::mpuffticp[MAPLE_NUM_SECTORS]

PUFFT Sector x(0..3) Initialization Configuration parameter.

2.13.3.6.2.8.56 volatile uint32_t maple_lte_pram_map_t::mpufftpmtap[MAPLE_NUM_SECTORS]

PUFFT Sector x Pre-multiplication table Address parameter (x=03)

2.13.3.6.2.8.57 volatile uint16_t maple_lte_pram_map_t::mpufftfcbr[MAPLE_NUM_SECTORS]

PUFFT Sector<x> Frequency Correction Base Value Real Part (x = 0...3).

2.13.3.6.2.8.58 volatile uint16_t maple_lte_pram_map_t::mpufftfcbi[MAPLE_NUM_SECTORS]

PUFFT Sector<x> Frequency Correction Base Value Imaginary Part (x= 0...3).

2.13.3.6.2.8.59 volatile uint32_t maple_lte_pram_map_t::mpufftfcsr[MAPLE_NUM_SECTORS]

PUFFT Sector<x> Frequency Correction Shift Value Real Part (x = 0...3).

2.13.3.6.2.8.60 volatile uint32_t maple_lte_pram_map_t::mpufftfcsi[MAPLE_NUM_SECTORS]

PUFFT Sector<x> Frequency Correction Shift Value Imaginary Part (x = 0...3).

2.13.3.6.2.8.61 volatile uint32_t maple_lte_pram_map_t::mpufftpstmtap[MAPLE_NUM_SECTORS]

PUFFT Sector<x> Vector Post multiplication table address (x = 0...3).

2.13.3.6.2.8.62 volatile uint32_t maple_lte_pram_map_t::mpdsch_pad_data[4]

MAPLE PDSCH PAD Data Configuration register - 0x1690 to 0x169f.

2.13.3.6.2.8.63 volatile uint32_t maple_lte_pram_map_t::mpdsch_pad_data1[4]

MAPLE PDSCH PAD Data Configuration 1 register - 0x16B0 to 0x16Bf.

2.13.3.6.2.8.64 volatile uint32_t maple_lte_pram_map_t::mpdsch_pad_data2[4]

MAPLE PDSCH PAD Data Configuration 2 register - 0x16c0 to 0x16cf.

2.13.3.6.2.8.65 volatile uint32_t maple_lte_pram_map_t::mpdsch_sec_config[MAPLE_NUM_SECTORS]

PDSCH EDF Sector API configuration parameters.

2.13.3.6.2.8.66 volatile uint8_t maple_lte_pram_map_t::mpdsch_ant_config[MAPLE_NUM_SECTORS][16]

PDSCH EDF Target Antenna API Configuration Parameters.

2.13.3.6.2.8.67 volatile uint32_t maple_lte_pram_map_t::mpdsch_cell_config[MAPLE_NUM_SECTORS]

PDSCH EDF Cell Configuration Parameters.

2.13.3.6.2.8.68 volatile uint32_t maple_lte_pram_map_t::mpdsch_sec_config_2nd_set[MAPLE_NUM_SECTORS]

PDSCH EDF Second Set of Sector API configuration parameters.

2.13.3.6.2.8.69 volatile uint32_t maple_lte_pram_map_t::mpdsch_cell_config_2nd_set[MAPLE_NUM_SECTORS]

PDSCH EDF Second Set of Cell Configuration Parameters.

2.13.3.6.2.8.70 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qse0p[8]

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 0.

2.13.3.6.2.8.71 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qse0p[8]

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 0.

2.13.3.6.2.8.72 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qse0p[8]

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 0.

2.13.3.6.2.8.73 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qse1p[8]

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 1.

2.13.3.6.2.8.74 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qse1p[8]

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 1.

2.13.3.6.2.8.75 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qse1p[8]

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 1.

2.13.3.6.2.8.76 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qse2p[8]

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 2.

2.13.3.6.2.8.77 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qse2p[8]

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 2.

2.13.3.6.2.8.78 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qse2p[8]

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 2.

2.13.3.6.2.8.79 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qse3p[8]

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 3.

2.13.3.6.2.8.80 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qse3p[8]

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 3.

2.13.3.6.2.8.81 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qse3p[8]

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 3.

2.13.3.6.2.8.82 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qs0wp

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 0 Write Pointer.

2.13.3.6.2.8.83 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qs0wp

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 0 Write Pointer.

2.13.3.6.2.8.84 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qs0wp

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 0 Write Pointer.

2.13.3.6.2.8.85 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qs1wp

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 1 Write Pointer.

2.13.3.6.2.8.86 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qs1wp

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 1 Write Pointer.

2.13.3.6.2.8.87 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qs1wp

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 1 Write Pointer.

2.13.3.6.2.8.88 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qs2wp

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 2 Write Pointer.

2.13.3.6.2.8.89 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qs2wp

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 2 Write Pointer.

2.13.3.6.2.8.90 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qs2wp

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 2 Write Pointer.

2.13.3.6.2.8.91 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qs3wp

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 3 Write Pointer.

2.13.3.6.2.8.92 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qs3wp

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 3 Write Pointer.

2.13.3.6.2.8.93 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qs3wp

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 3 Write Pointer.

2.13.3.6.2.8.94 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qs0rp

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 0 Read Pointer.

2.13.3.6.2.8.95 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qs0rp

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 0 Read Pointer.

2.13.3.6.2.8.96 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qs0rp

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 0 Read Pointer.

2.13.3.6.2.8.97 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qs1rp

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 1 Read Pointer.

2.13.3.6.2.8.98 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qs1rp

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 1 Read Pointer.

2.13.3.6.2.8.99 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qs1rp

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 1 Read Pointer.

2.13.3.6.2.8.100 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qs2rp

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 2 Read Pointer.

2.13.3.6.2.8.101 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qs2rp

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 2 Read Pointer.

2.13.3.6.2.8.102 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qs2rp

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 2 Read Pointer.

2.13.3.6.2.8.103 volatile uint32_t maple_lte_pram_map_t::mpdsch_start_qs3rp

MAPLE PDSCH2_EDF PDSCH_START Queue Sector 3 Read Pointer.

2.13.3.6.2.8.104 volatile uint32_t maple_lte_pram_map_t::mpdsch_pdpe_qs3rp

MAPLE PDSCH2_EDF PDSCH_PDPE Queue Sector 3 Read Pointer.

2.13.3.6.2.8.105 volatile uint32_t maple_lte_pram_map_t::mpdsch_done_qs3rp

MAPLE PDSCH2_EDF PDSCH_DONE Queue Sector 3 Read Pointer.

2.13.3.6.2.8.106 volatile uint32_t maple_lte_pram_map_t::mpdsch_s0np[7]

MAPLE PDSCH2_EDF Sector 0 status #n parameter where n = 0..6 (n == 6 is error parameter)

2.13.3.6.2.8.107 volatile uint32_t maple_lte_pram_map_t::mpdsch_s1np[7]

MAPLE PDSCH2_EDF Sector 1 status #n parameter where n = 0..6 (n == 6 is error parameter)

2.13.3.6.2.8.108 volatile uint32_t maple_lte_pram_map_t::mpdsch_s2np[7]

MAPLE PDSCH2_EDF Sector 2 status #n parameter where n = 0..6 (n == 6 is error parameter)

2.13.3.6.2.8.109 volatile uint32_t maple_lte_pram_map_t::mpdsch_s3np[7]

MAPLE PDSCH2_EDF Sector 3 status #n parameter where n = 0..6 (n == 6 is error parameter)

2.13.3.6.2.8.110 volatile uint32_t maple_lte_pram_map_t::maple_xxpe_acc[14]

MAPLE PE Load accumulator : #0 - MAPLE EQPE Load accumulator.

#1-3 - MAPLE eFTPE(0,1,2) Load accumulator. #4 - MAPLE TVPE Load accumulator. #5 - MAPLE DEPE Load accumulator.

2.13.3.6.2.8.111 volatile uint32_t maple_lte_pram_map_t::mpdsch_bf_config_2nd_set[8][MAPLE_NUM_SECTORS]

PDSCH EDF second set Beam Forming Configuration Parameter - 8 register per sector.

2.13.3.6.2.8.112 volatile uint32_t maple_lte_pram_map_t::mpdschoba_2nd_set[MAPLE_NUM_SECTORS][MAPLE_NUM_ANT]

MAPLE PDSCH_EDF Second set sector x Antenna y Output address.

2.13.3.6.2.8.113 volatile uint32_t maple_lte_pram_map_t::mpdschobs_2nd_set[MAPLE_NUM_SECTORS]

MAPLE PDSCH_EDF Second set sector x Antenna y Output size.

2.13.3.6.2.8.114 volatile uint32_t maple_lte_pram_map_t::mpdsch_ack[PDSCH_SOC_TIMER_SETS][2]

PDSCH_EDF SoC Timers configuration parameters: BASE TMR0, DATA TMR0, BASE TMR1, DATA TMR1.

2.13.3.6.2.8.115 volatile uint32_t maple_lte_pram_map_t::mpdschoba[MAPLE_NUM_SECTORS][MAPLE_NUM_ANT]

MAPLE PDSCH_EDF sector x Antenna y Output address.

2.13.3.6.2.8.116 volatile uint32_t maple_lte_pram_map_t::mpdschobs[MAPLE_NUM_SECTORS]

MAPLE PDSCH_EDF sector x Antenna y Output size.

2.13.3.6.2.8.117 volatile uint32_t maple_lte_pram_map_t::mpufftiba[MAPLE_NUM_SECTORS][MAPLE_NUM_ANT]

MAPLE PUFFT sector x Antenna y Input address.

2.13.3.6.2.8.118 volatile uint32_t maple_lte_pram_map_t::mpufftibs[MAPLE_NUM_SECTORS]

MAPLE PUFFT sector x Antenna y Input size.

2.13.3.6.2.9 struct maple_wcdma_pram_map_t

Maple WCDMA PRAM memory map.

Data Fields

- volatile uint32_t mbdrcp0
- volatile uint32_t mbdrcp1
- volatile uint32_t mucvp
- volatile uint32_t mp_tpp
- volatile uint32_t mcgcp
- volatile uint32_t mmc0p
- volatile uint32_t mmc1p
- volatile uint32_t mtvmp
- volatile uint32_t cdomcp
- volatile uint32_t cubmsp

- volatile uint32_t mcrrcip
- maple_pe_bd_ring_params_t ftpe_bd [MAPLE3W_FTPE_NUM]
- maple_pe_bd_ring_params_t tcpe_bd
- maple_pe_bd_ring_params_t crcpe_bd
- maple_pe_bd_ring_params_t cgpe_bd
- maple_pe_bd_ring_params_t pufft_bd
- maple_pe_bd_ring_steering_t ftpe_steering [MAPLE3W_FTPE_NUM]
- maple_pe_bd_ring_steering_t tcpe_steering
- maple_pe_bd_ring_steering_t crcpe_steering
- maple_pe_bd_ring_steering_t cgpe_steering
- maple_pe_bd_ring_steering_t pufft_steering
- ftpedss_t ftpedss
- struct {
  volatile uint32_t uprmbpp
  volatile uint32_t upsmbpp
  volatile uint32_t ubsp
  } multiply_ftpe [MAPLE3W_FTPE_NUM]

- volatile uint32_t ftpecubrp
- maple_pe_mpisr_tid_t mpisr_tid
- volatile uint32_t mpuffticp [MAPLE_NUM_SECTORS]
- volatile uint32_t mpufftpmtap [MAPLE_NUM_SECTORS]
- volatile uint16_t mpufftfcbr [MAPLE_NUM_SECTORS]
- volatile uint16_t mpufftfcbi [MAPLE_NUM_SECTORS]
- volatile uint32_t mpufftfcsr [MAPLE_NUM_SECTORS]
- volatile uint32_t mpufftfcsi [MAPLE_NUM_SECTORS]
- volatile uint32_t mpufftpstmtap [MAPLE_NUM_SECTORS]
- maple_mpisr_p_t mpisr_param [4]
- volatile uint16_t mpisr_cnt_int_to_task [4]
- volatile uint32_t mpufftiba [MAPLE_NUM_SECTORS][MAPLE_NUM_ANT]
- volatile uint32_t mpufftibs [MAPLE_NUM_SECTORS]

2.13.3.6.2.9.1 Field Documentation

2.13.3.6.2.9.2 volatile uint32_t maple_wcdma_pram_map_t::mbdrcp0

Maple BD Rings Configuration Parameter 0.

2.13.3.6.2.9.3 volatile uint32_t maple_wcdma_pram_map_t::mbdrcp1

Maple BD Rings Configuration Parameter 1.

2.13.3.6.2.9.4 volatile uint32_t maple_wcdma_pram_map_t::mucvp

Maple UCode Version Parameter.

2.13.3.6.2.9.5 volatile uint32_t maple_wcdma_pram_map_t::mp_tpp

Maple Timer Period Parameter.

This option is not available through SmartDSP OS.

2.13.3.6.2.9.6 volatile uint32_t maple_wcdma_pram_map_t::mcgcp

MAPLE Clock Gating Control Parameter.

2.13.3.6.2.9.7 volatile uint32_t maple_wcdma_pram_map_t::mmc0p

Maple Mode Configurations 0 Parameter.

2.13.3.6.2.9.8 volatile uint32_t maple_wcdma_pram_map_t::mmc1p

Maple Mode Configurations 1 Parameter.

2.13.3.6.2.9.9 volatile uint32_t maple_wcdma_pram_map_t::mtvmp

Maple ETVPE_MODE parameter.

2.13.3.6.2.9.10 volatile uint32_t maple_wcdma_pram_map_t::cdomcp

CRPE-DL Output Mode Configuration Parameter.

2.13.3.6.2.9.11 volatile uint32_t maple_wcdma_pram_map_t::cubmsp

CRPE-ULB Mode Configuration Parameter.

2.13.3.6.2.9.12 volatile uint32_t maple_wcdma_pram_map_t::mcrrcip

Maple CRPE Reset Completion Indication Parameter.

2.13.3.6.2.9.13 maple_pe_bd_ring_params_t maple_wcdma_pram_map_t::ftpe_bd[MAPLE3W↩ _FTPE_NUM]

FFTPE_x BD Ring Priority Configuration.

2.13.3.6.2.9.14 maple_pe_bd_ring_params_t maple_wcdma_pram_map_t::tcpe_bd

TCPE BD Ring Priority Configuration.

2.13.3.6.2.9.15 maple_pe_bd_ring_params_t maple_wcdma_pram_map_t::crcpe_bd

CRCPE BD Ring Priority Configuration.

2.13.3.6.2.9.16 maple_pe_bd_ring_params_t maple_wcdma_pram_map_t::cgpe_bd

CGPE BD Ring Priority Configuration.

2.13.3.6.2.9.17 maple_pe_bd_ring_params_t maple_wcdma_pram_map_t::pufft_bd

PUFFT BD Ring Priority Configuration.

2.13.3.6.2.9.18 maple_pe_bd_ring_steering_t maple_wcdma_pram_map_t::ftpe_steering[MAPLE3W_FTPE_NUM]

Task ID and Steering Parameters.

FTPE Task ID and Steering Parameters

2.13.3.6.2.9.19 maple_pe_bd_ring_steering_t maple_wcdma_pram_map_t::tcpe_steering

TCPE Task ID and Steering Parameters.

2.13.3.6.2.9.20 maple_pe_bd_ring_steering_t maple_wcdma_pram_map_t::crcpe_steering

CRCPE Task ID and Steering Parameters.

2.13.3.6.2.9.21 maple_pe_bd_ring_steering_t maple_wcdma_pram_map_t::cgpe_steering

CGPE Task ID and Steering Parameters.

2.13.3.6.2.9.22 maple_pe_bd_ring_steering_t maple_wcdma_pram_map_t::pufft_steering

PUFFT Task ID and Steering Parameters.

2.13.3.6.2.9.23 ftpedss_t maple_wcdma_pram_map_t::ftpedss

Maple FTPE Data Size parameters.

2.13.3.6.2.9.24 volatile uint32_t maple_wcdma_pram_map_t::uprmbpp

Update Pre-Multiplication Buffer Pointer.

2.13.3.6.2.9.25 volatile uint32_t maple_wcdma_pram_map_t::upsmbpp

Update Post-Multiplication Buffer Pointer.

2.13.3.6.2.9.26 volatile uint32_t maple_wcdma_pram_map_t::ubsp

Update Buffers Size Parameter.

2.13.3.6.2.9.27 struct { ... } maple_wcdma_pram_map_t::multiply_ftpe[MAPLE3W_FTPE_NUM]

eFTPE x Update Post/Pre-Multiplication parameters

2.13.3.6.2.9.28 volatile uint32_t maple_wcdma_pram_map_t::ftpecubrp

UBCUpdate Buffers Complete indication.

2.13.3.6.2.9.29 maple_pe_mpisr_tid_t maple_wcdma_pram_map_t::mpisr_tid

MMU TASK ID in use by the MPISRx.

2.13.3.6.2.9.30 volatile uint32_t maple_wcdma_pram_map_t::mpuffticp[MAPLE_NUM_SECTORS]

Pufft Sector x(0..3) Initialization Configuration parameter.

2.13.3.6.2.9.31 volatile uint32_t maple_wcdma_pram_map_t::mpufftpmtap[MAPLE_NUM_SECTORS]

PUFFT Sector x Pre-multiplication table Address parameter (x=03)

2.13.3.6.2.9.32 volatile uint16_t maple_wcdma_pram_map_t::mpufftfcbr[MAPLE_NUM_SECTORS]

PUFFT Sector Frequency Correction Base Value Real Part <x> parameter (x = 0...3).

2.13.3.6.2.9.33 volatile uint16_t maple_wcdma_pram_map_t::mpufftfcbi[MAPLE_NUM_SECTORS]

PUFFT Frequency Correction Base Value Imaginary Part.

<x> parameter (x= 0...3).

2.13.3.6.2.9.34 volatile uint32_t maple_wcdma_pram_map_t::mpufftfcsr[MAPLE_NUM_SECTORS]

PUFFT Frequency Correction Shift Value Real Part <x> parameter (x = 0...3).

2.13.3.6.2.9.35 volatile uint32_t maple_wcdma_pram_map_t::mpufftfcsi[MAPLE_NUM_SECTORS]

PUFFT Frequency Correction Shift Value Imaginary Part <x> parameter (x = 0...3).

2.13.3.6.2.9.36 volatile uint32_t maple_wcdma_pram_map_t::mpufftpstmtap[MAPLE_NUM_SECTORS]

PUFFT Vector Post multiplication table address <x> parameter (x = 0...3).

2.13.3.6.2.9.37 maple_mpisr_p_t maple_wcdma_pram_map_t::mpisr_param[4]

MAPLE Interrupt Service Routine parameters.

2.13.3.6.2.9.38 volatile uint16_t maple_wcdma_pram_map_t::mpisr_cnt_int_to_task[4]

MAPLE Interrupt Service Routine Task *Counter*.

2.13.3.6.2.9.39 volatile uint32_t maple_wcdma_pram_map_t::mpufftiba[MAPLE_NUM_SECTORS][MAPLE_NUM_ANT]

MAPLE PUFFT sector x Antenna y Input address.

2.13.3.6.2.9.40 volatile uint32_t maple_wcdma_pram_map_t::mpufftibs[MAPLE_NUM_SECTORS]

MAPLE PUFFT sector x Antenna y Input size.

2.13.3.6.2.10 struct maple_lte_psif_dram_map_t

Maple PSIF memory map.

Public Member Functions

- OS_MEM_RESERVED (0x18000, 0x19C26)

Data Fields

- maple_lte_pram_map_t pram
- volatile uint8_t bd_rings [PRAM_BD_RING_SIZE]
- volatile uint16_t mpdschdcp [MAPLE_NUM_SECTORS][46]
- maple_mpisr_p_t mpisr_param [4]
- volatile uint16_t mpisr_cnt_int_to_task [4]
- volatile uint8_t smartdsp_os_reserved [0x200]

2.13.3.6.2.10.1 Member Function Documentation

2.13.3.6.2.10.2 maple_lte_psif_dram_map_t::OS_MEM_RESERVED (0x18000 , 0x19C26)

RECONFIGURATION_BASE = 0x19C26.

2.13.3.6.2.10.3 Field Documentation

2.13.3.6.2.10.4 maple_lte_pram_map_t maple_lte_psif_dram_map_t::pram

See maple_lte_pram_map_t.

2.13.3.6.2.10.5 volatile uint8_t maple_lte_psif_dram_map_t::bd_rings[PRAM_BD_RING_SIZE]

BD rings space in PRAM.

2.13.3.6.2.10.6 volatile uint16_t maple_lte_psif_dram_map_t::mpdschdcp[MAPLE_NUM_SECTORS][46]

PDSCH Cell Reconfiguration Parameters.

2.13.3.6.2.10.7 maple_mpisr_p_t maple_lte_psif_dram_map_t::mpisr_param[4]

MAPLE Interrupt Service Routine parameters.

2.13.3.6.2.10.8 volatile uint16_t maple_lte_psif_dram_map_t::mpisr_cnt_int_to_task[4]

MAPLE Interrupt Service Routine Task *Counter*.

2.13.3.6.2.10.9 volatile uint8_t maple_lte_psif_dram_map_t::smartdsp_os_reserved[0x200]

Area reserved for SmartDSP OS.

2.13.3.6.2.11 struct maple_wcdma_psif_dram_map_t

Maple WCDMA PSIF memory map.

Data Fields

- maple_wcdma_pram_map_t pram
- volatile uint8_t bd_rings [PRAM_BD_RING_SIZE]
- volatile uint8_t smartdsp_os_reserved [0x200]

2.13.3.6.2.11.1 Field Documentation

2.13.3.6.2.11.2 maple_wcdma_pram_map_t maple_wcdma_psif_dram_map_t::pram

See maple_wcdma_pram_map_t.

2.13.3.6.2.11.3 volatile uint8_t maple_wcdma_psif_dram_map_t::bd_rings[PRAM_BD_RING_S↩ IZE]

BD rings space in PRAM.

2.13.3.6.2.11.4 volatile uint8_t maple_wcdma_psif_dram_map_t::smartdsp_os_reserved[0x200]

Area reserved for SmartDSP OS.

2.13.3.6.2.12 struct maple_tvpe_map_t

TVPE registers memory map.

Data Fields

- volatile uint32_t tvaqcr

2.13.3.6.2.12.1 Field Documentation

2.13.3.6.2.12.2 volatile uint32_t maple_tvpe_map_t::tvaqcr

eTVPE Aposteriori Quality Configuration Register

2.13.3.6.2.13 struct maple_ftpe_map_t

FTPE registers memory map.

Data Fields

- volatile uint32_t ftpedsr0
- volatile uint32_t ftpedsr1
- volatile uint32_t ftpedsr2
- volatile uint32_t ftpecr

- volatile uint32_t ftpestr

2.13.3.6.2.13.1 Field Documentation

2.13.3.6.2.13.2 volatile uint32_t maple_ftpe_map_t::ftpedsr0

FTPE Data Size Register 0.

2.13.3.6.2.13.3 volatile uint32_t maple_ftpe_map_t::ftpedsr1

FTPE Data Size Register 1.

2.13.3.6.2.13.4 volatile uint32_t maple_ftpe_map_t::ftpedsr2

FTPE Data Size Register 2.

2.13.3.6.2.13.5 volatile uint32_t maple_ftpe_map_t::ftpecr

FTPE_x Configuration Register.

2.13.3.6.2.13.6 volatile uint32_t maple_ftpe_map_t::ftpestr

FTPE_x ECC Status Register.

2.13.3.6.2.14 struct maple_depe_map_t

DEPE registers memory map.

2.13.3.6.2.15 struct maple_eqpe_map_t

EQPE registers memory map.

Data Fields

- volatile uint32_t eq_eccevent

2.13.3.6.2.15.1 Field Documentation

2.13.3.6.2.15.2 volatile uint32_t maple_eqpe_map_t::eq_eccevent

EQPE ECC Event Register.

2.13.3.6.2.16 struct maple_pdpe_map_t

PDPE registers memory map.

Data Fields

- volatile uint32_t pdpe_pesr
- volatile uint32_t pdpe_sec_conf [MAPLE_NUM_SECTORS]

2.13.3.6.2.16.1 Field Documentation

2.13.3.6.2.16.2 volatile uint32_t maple_pdpe_map_t::pdpe_pesr

This register records ECC events of the PDPE memories.

2.13.3.6.2.16.3 volatile uint32_t maple_pdpe_map_t::pdpe_sec_conf[MAPLE_NUM_SECTORS]

0x37f010 Sector Configuration register

2.13.3.6.2.17 struct maple_lte_mbus_memmap_t

Maple3lw MBUS memory map.

Data Fields

- maple_tvpe_map_t tvpe
- maple_ftpe_map_t ftpe [MAPLE3LW_FTPE_NUM]
- maple_depe_map_t depe
- maple_eqpe_map_t eqpe
- maple_pdpe_map_t pdpe

2.13.3.6.2.17.1 Field Documentation

2.13.3.6.2.17.2 maple_tvpe_map_t maple_lte_mbus_memmap_t::tvpe

eTVPE Registers

2.13.3.6.2.17.3 maple_ftpe_map_t maple_lte_mbus_memmap_t::ftpe[MAPLE3LW_FTPE_NUM]

FTPE Registers.

2.13.3.6.2.17.4 maple_depe_map_t maple_lte_mbus_memmap_t::depe

DEPE Registers.

2.13.3.6.2.17.5 maple_eqpe_map_t maple_lte_mbus_memmap_t::eqpe

EQPE Registers.

2.13.3.6.2.17.6 maple_pdpe_map_t maple_lte_mbus_memmap_t::pdpe

PDPE Registers.

2.13.3.6.2.18 struct maple_wcdma_mbus_memmap_t

Maple3w MBUS memory map.

Data Fields

- maple_ftpe_map_t ftpe [MAPLE3W_FTPE_NUM]

2.13.3.6.2.18.1 Field Documentation

2.13.3.6.2.18.2 maple_ftpe_map_t maple_wcdma_mbus_memmap_t::ftpe[MAPLE3W_FTPE_NUM]

FTPE Registers.

2.13.3.6.2.19 struct maple_lte_sbus_memmap_t

Maple3lw SBUS memory map.

Data Fields

- volatile uint32_t m_dalr
- volatile uint32_t m_damr
- volatile uint32_t m_ecr
- volatile uint32_t m_ear
- volatile uint32_t m_cr
- volatile uint32_t m_dsdval
- volatile uint32_t m_dsda_pl
- volatile uint32_t m_dsdb_pl
- volatile uint32_t m_dsdc_pl
- volatile uint32_t m_dsdm_pl
- volatile uint32_t m_dmper
- volatile uint32_t pcr
- volatile uint32_t pspicer0
- volatile uint32_t pspicer1
- volatile uint32_t pspicer2
- volatile uint32_t pspicelr
- volatile uint32_t pspicmr0
- volatile uint32_t pspicmr1
- volatile uint32_t pspicmr2
- volatile uint32_t pspiciacr
- volatile uint32_t mpp_dst
- volatile uint32_t clk_counter
- volatile uint32_t m_dsda
- volatile uint32_t m_dsdb
- volatile uint32_t m_dsdc
- volatile uint32_t m_dsdd

2.13.3.6.2.19.1 Field Documentation

2.13.3.6.2.19.2 volatile uint32_t maple_lte_sbus_memmap_t::m_dalr

MMU Trash Destinaltion Address LSB Register.

2.13.3.6.2.19.3 volatile uint32_t maple_lte_sbus_memmap_t::m_damr

MMU Trash Destinaltion Address MSB Register.

2.13.3.6.2.19.4 volatile uint32_t maple_lte_sbus_memmap_t::m_ecr

This register captures MMU error events and their reason.

2.13.3.6.2.19.5 volatile uint32_t maple_lte_sbus_memmap_t::m_ear

This register captures the address which caused the MMU error event.

2.13.3.6.2.19.6 volatile uint32_t maple_lte_sbus_memmap_t::m_cr

This register holds the MMU general control parameters.

2.13.3.6.2.19.7 volatile uint32_t maple_lte_sbus_memmap_t::m_dsdval

The M_DSDVAL contains 1 bit per data segment descriptor.

2.13.3.6.2.19.8 volatile uint32_t maple_lte_sbus_memmap_t::m_dsda_pl

Preload register that is used to configure data segment descriptor A registers.

2.13.3.6.2.19.9 volatile uint32_t maple_lte_sbus_memmap_t::m_dsdb_pl

Preload register that is used to configure data segment descriptor B registers.

2.13.3.6.2.19.10 volatile uint32_t maple_lte_sbus_memmap_t::m_dsdc_pl

Preload register that is used to configure data segment descriptor C registers.

2.13.3.6.2.19.11 volatile uint32_t maple_lte_sbus_memmap_t::m_dsdm_pl

Upon writing to this register, the relevant descriptor is being uploaded with the values of M_DSDA_PL, M_DSDB_PL, M_DSDC_PL and some attributes of M_DSDM_PL.

2.13.3.6.2.19.12 volatile uint32_t maple_lte_sbus_memmap_t::m_dmper

This register indicates on a descriptor programming error.

2.13.3.6.2.19.13 volatile uint32_t maple_lte_sbus_memmap_t::m_dsda

Data Segment Descriptor Registers A0.

2.13.3.6.2.19.14 volatile uint32_t maple_lte_sbus_memmap_t::m_dsdb

Data Segment Descriptor Registers B0.

2.13.3.6.2.19.15 volatile uint32_t maple_lte_sbus_memmap_t::m_dsdc

Data Segment Descriptor Registers C0.

2.13.3.6.2.19.16 volatile uint32_t maple_lte_sbus_memmap_t::m_dsdd

Data Segment Descriptor Registers D0.

2.13.3.6.2.19.17 volatile uint32_t maple_lte_sbus_memmap_t::pcr

PSIF Command Register.

2.13.3.6.2.19.18 volatile uint32_t maple_lte_sbus_memmap_t::pspicer0

PSIF Pic Event Register 0, R/(W 1 to clear)

2.13.3.6.2.19.19 volatile uint32_t maple_lte_sbus_memmap_t::pspicer1

PSIF Pic Event Register 1, R/(W 1 to clear)

2.13.3.6.2.19.20 volatile uint32_t maple_lte_sbus_memmap_t::pspicer2

PSIF Pic Event Register 2, R/(W 1 to clear)

2.13.3.6.2.19.21 volatile uint32_t maple_lte_sbus_memmap_t::pspicelr

PSIF Pic Edge/Level Register.

2.13.3.6.2.19.22 volatile uint32_t maple_lte_sbus_memmap_t::pspicmr0

PSIF Pic Mask Register 0.

2.13.3.6.2.19.23 volatile uint32_t maple_lte_sbus_memmap_t::pspicmr1

PSIF Pic Mask Register 1.

2.13.3.6.2.19.24 volatile uint32_t maple_lte_sbus_memmap_t::pspicmr2

PSIF Pic Mask Register 2.

2.13.3.6.2.19.25 volatile uint32_t maple_lte_sbus_memmap_t::pspiciacr

PSIF PIC Interrupt Assertion Clocks Registers.

2.13.3.6.2.19.26 volatile uint32_t maple_lte_sbus_memmap_t::mpp_dst

This register is a status register indicating the power status (on, off) of the internal PEs.

2.13.3.6.2.19.27 volatile uint32_t maple_lte_sbus_memmap_t::clk_counter

MAPLE COUNTER - counts MAPLE cycles at MAPLE CLK rate.

2.13.3.6.3 Macro Definition Documentation

2.13.3.6.3.1 #define PRAM_BD_RING_SIZE (0x18000 - 0x10000)

BD ring size in Maple PRAM.

2.13.3.6.3.2 #define MAPLE_TRACE_BUFFER_SIZE (0x18000 - 0x17000)

MAPLE Trace Buffer Size.

2.13.3.6.3.3 #define FTPE_DSS_NUM 7

The number of Data Size Sets 0 - 5 including set number 7 which is used internally.

2.13.3.6.3.4 #define TVPE_VITERBI_POLY_SETS_NUM 3

The number of Maple Turbo Viterbi Polynomial Vector Set.

2.13.3.6.3.5 #define PUFFT_SOC_TIMER_SETS 4

The number of Maple PUFFT configurable SOC timers sets.

2.13.3.6.3.6 #define PDSCH_SOC_TIMER_SETS 16

The number of Maple PDSCH configurable SOC timers sets.

2.13.3.6.3.7 #define MAPLE3LW_FTPE_NUM 3

The number of ftpes in MAPLE3LW.

2.13.3.6.3.8 #define MAPLE3W_FTPE_NUM 2

The number of ftpes in MAPLE3W.

2.13.3.6.3.9 #define MAPLE_STP_NUM 4

Then number of steering bits registers.

2.13.3.6.3.10 #define MAPLE_NUM_SLOTS 2

Number of slots.

2.13.3.6.3.11 #define MAPLE_NUM_SECTORS 4

Number of supported sectors.

2.13.3.6.3.12 #define MAPLE_NUM_ANT 8

Number of antennas.

2.13.3.6.3.13 #define MAPLE_MAX_NUM_PE_BD_RINGS_LOW_PRIORITY 8

Number of MAPLE BDs per low priority.

2.13.3.6.3.14 #define MAPLE_MAX_NUM_PE_BD_RINGS_HIGH_PRIORITY 8

Number of MAPLE BDs per high priority.

2.13.3.6.3.15 #define MAPLE_PE_MAX_NUM_BD_RINGS (MAPLE_MAX_NUM_PE_BD_RING↩ S_LOW_PRIORITY + MAPLE_MAX_NUM_PE_BD_RINGS_HIGH_PRIORITY)

Total number of MAPLE BDs per priority.

2.13.3.6.4 Typedef Documentation

2.13.3.6.4.1 typedef maple_lte_sbus_memmap_t maple_wcdma_sbus_memmap_t

Maple3w SBUS memory map.

2.13.3.7 sRIO Memory Map

2.13.3.7.1 Overview

```
Definition of sRIO registers and memory map
```

Data Structures

- struct rapidio_arch_map_t
- struct rapidio_port_status_map_t
- struct rapidio_1xlp_serial_map_t
- struct rapidio_err_log_map_t
- struct rapidio_err_phy_map_t
- struct rapidio_port_general_map_t
- struct rapidio_general_map_t
- struct rapidio_rev_map_t
- struct rapidio_atmu_out_t
- struct rapidio_atmu_in_t
- struct rapidio_atmu_map_t
- struct srio_memmap_t

2.13.3.7.2 Data Structure Documentation

2.13.3.7.2.1 struct rapidio_arch_map_t

RapidIO Architecture Registers.

Data Fields

- volatile uint32_t didcar
- volatile uint32_t dicar
- volatile uint32_t aidcar
- volatile uint32_t aicar
- volatile uint32_t pefcar
- volatile uint32_t socar
- volatile uint32_t docar
- volatile uint32_t dsicar
- volatile uint32_t dsllcsr

- volatile uint32_t pellccsr
- volatile uint32_t lcsba1csr
- volatile uint32_t bdidcsr
- volatile uint32_t hbdidlcsr
- volatile uint32_t ctcsr

2.13.3.7.2.1.1 Field Documentation

2.13.3.7.2.1.2 volatile uint32_t rapidio_arch_map_t::didcar

Device identity capability register.

2.13.3.7.2.1.3 volatile uint32_t rapidio_arch_map_t::dicar

Device information capability register.

2.13.3.7.2.1.4 volatile uint32_t rapidio_arch_map_t::aidcar

Assembly identity capability register.

2.13.3.7.2.1.5 volatile uint32_t rapidio_arch_map_t::aicar

Assembly information capability register.

2.13.3.7.2.1.6 volatile uint32_t rapidio_arch_map_t::pefcar

Processing element features capability register.

2.13.3.7.2.1.7 volatile uint32_t rapidio_arch_map_t::socar

Source operations capability register.

2.13.3.7.2.1.8 volatile uint32_t rapidio_arch_map_t::docar

Destination operations capability register.

2.13.3.7.2.1.9 volatile uint32_t rapidio_arch_map_t::dsicar

Data streaming information capability register - Not implemented on all platforms.

2.13.3.7.2.1.10 volatile uint32_t rapidio_arch_map_t::dsllcsr

Data streaming logic layer command and status register - Not implemented on all platforms.

2.13.3.7.2.1.11 volatile uint32_t rapidio_arch_map_t::pellccsr

Processing element logical layer control command and status register.

2.13.3.7.2.1.12 volatile uint32_t rapidio_arch_map_t::lcsba1csr

Local configuration space base address 1 command and status register.

2.13.3.7.2.1.13 volatile uint32_t rapidio_arch_map_t::bdidcsr

Base device ID command and status register.

2.13.3.7.2.1.14 volatile uint32_t rapidio_arch_map_t::hbdidlcsr

Host base device ID lock command and status register.

2.13.3.7.2.1.15 volatile uint32_t rapidio_arch_map_t::ctcsr

Component tag command and status register.

2.13.3.7.2.2 struct rapidio_port_status_map_t

RapidIO Port Status Registers.

Data Fields

- volatile uint32_t lmreqcsr
- volatile uint32_t lmrespcsr
- volatile uint32_t lascsr
- volatile uint32_t escsr
- volatile uint32_t ccsr

2.13.3.7.2.2.1 Field Documentation

2.13.3.7.2.2.2 volatile uint32_t rapidio_port_status_map_t::lmreqcsr

Port x Link Maintenance request Command and Status Register.

2.13.3.7.2.2.3 volatile uint32_t rapidio_port_status_map_t::lmrespcsr

Port x Link maintenance response command and status register.

2.13.3.7.2.2.4 volatile uint32_t rapidio_port_status_map_t::lascsr

Port x Local ackID Command and Status Register.

2.13.3.7.2.2.5 volatile uint32_t rapidio_port_status_map_t::escsr

Error and status command and status register.

2.13.3.7.2.2.6 volatile uint32_t rapidio_port_status_map_t::ccsr

Port x Control command and status register.

2.13.3.7.2.3 struct rapidio_1xlp_serial_map_t

RapidIO Serial Port Registers.

Data Fields

- volatile uint32_t pmbh0
- volatile uint32_t pltoccsr
- volatile uint32_t prtoccsr
- volatile uint32_t pgccsr
- rapidio_port_status_map_t port_status [NUMBER_OF_SRIO_PHY_PORTS]

2.13.3.7.2.3.1 Field Documentation

2.13.3.7.2.3.2 volatile uint32_t rapidio_1xlp_serial_map_t::pmbh0

Port maintenance block header 0.

2.13.3.7.2.3.3 volatile uint32_t rapidio_1xlp_serial_map_t::pltoccsr

Port link time-out control command and status register.

2.13.3.7.2.3.4 volatile uint32_t rapidio_1xlp_serial_map_t::prtoccsr

Port response time-out control command and status register.

2.13.3.7.2.3.5 volatile uint32_t rapidio_1xlp_serial_map_t::pgccsr

Port General control command and status register (SRIO_GCCSR)

2.13.3.7.2.3.6 rapidio_port_status_map_t rapidio_1xlp_serial_map_t::port_status[NUMBER_O↩ F_SRIO_PHY_PORTS]

Ports command and status registers.

2.13.3.7.2.4 struct rapidio_err_log_map_t

RapidIO Logical Error Registers.

Data Fields

- volatile uint32_t erbh
- volatile uint32_t ltledcsr
- volatile uint32_t ltleecsr
- volatile uint32_t ltlaccsr
- volatile uint32_t ltldidccsr
- volatile uint32_t ltlcccsr

2.13.3.7.2.4.1 Field Documentation

2.13.3.7.2.4.2 volatile uint32_t rapidio_err_log_map_t::erbh

Error reporting block header.

2.13.3.7.2.4.3 volatile uint32_t rapidio_err_log_map_t::ltledcsr

Logical/Transport layer error detect command and status register.

2.13.3.7.2.4.4 volatile uint32_t rapidio_err_log_map_t::ltleecsr

Logical/Transport layer error enable command and status register.

2.13.3.7.2.4.5 volatile uint32_t rapidio_err_log_map_t::ltlaccsr

Logical/Transport layer address capture command and status register.

2.13.3.7.2.4.6 volatile uint32_t rapidio_err_log_map_t::ltldidccsr

Logical/Transport layer device ID capture command and status register.

2.13.3.7.2.4.7 volatile uint32_t rapidio_err_log_map_t::ltlcccsr

Logical/Transport layer control capture command and status register.

2.13.3.7.2.5 struct rapidio_err_phy_map_t

RapidIO Physical Error Registers.

Data Fields

- volatile uint32_t edcsr
- volatile uint32_t erecsr
- volatile uint32_t ecacsr
- volatile uint32_t pcseccsr [4]
- volatile uint32_t ercsr
- volatile uint32_t ertcsr

2.13.3.7.2.5.1 Field Documentation

2.13.3.7.2.5.2 volatile uint32_t rapidio_err_phy_map_t::edcsr

Port Error detect command and status register.

2.13.3.7.2.5.3 volatile uint32_t rapidio_err_phy_map_t::erecsr

Port Error rate enable command and status register.

2.13.3.7.2.5.4 volatile uint32_t rapidio_err_phy_map_t::ecacsr

Port Error capture attributes command and status.

2.13.3.7.2.5.5 volatile uint32_t rapidio_err_phy_map_t::pcseccsr[4]

Port Packet/control symbol error capture command and status registers.

2.13.3.7.2.5.6 volatile uint32_t rapidio_err_phy_map_t::ercsr

Port Error rate command and status register.

2.13.3.7.2.5.7 volatile uint32_t rapidio_err_phy_map_t::ertcsr

Port Error rate threshold command and status register.

2.13.3.7.2.6 struct rapidio_port_general_map_t

RapidIO Port Registers.

Data Fields

- volatile uint32_t pnadidcsr
- volatile uint32_t pnaacr
- volatile uint32_t pnlopttlcr
- volatile uint32_t pniecsr
- volatile uint32_t pnpcr
- volatile uint32_t pnslcsr
- volatile uint32_t pnsleicr

2.13.3.7.2.6.1 Field Documentation

2.13.3.7.2.6.2 volatile uint32_t rapidio_port_general_map_t::pnadidcsr

Port n Alternate device ID command and status register (SRIO_P1ADIDCSR)

2.13.3.7.2.6.3 volatile uint32_t rapidio_port_general_map_t::pnaacr

Port n Accept-all configuration register (SRIO_P1AACR)

2.13.3.7.2.6.4 volatile uint32_t rapidio_port_general_map_t::pnlopttlcr

Port n Logical Outbound Packet Time-to-Live Configuration Register (SRIO_P1LOPTTLCR)

2.13.3.7.2.6.5 volatile uint32_t rapidio_port_general_map_t::pniecsr

Port n Implementation error command and status register (SRIO_P1IECSR)

2.13.3.7.2.6.6 volatile uint32_t rapidio_port_general_map_t::pnpcr

Port n Physical configuration register (SRIO_P1PCR)

2.13.3.7.2.6.7 volatile uint32_t rapidio_port_general_map_t::pnslcsr

Port n Serial link command and status register (SRIO_P1SLCSR)

2.13.3.7.2.6.8 volatile uint32_t rapidio_port_general_map_t::pnsleicr

Port n Serial link error injection configuration register (SRIO_P1SLEICR)

2.13.3.7.2.7 struct rapidio_general_map_t

RapidIO General Registers.

Data Fields

- volatile uint32_t llcr
- volatile uint32_t epwisr
- volatile uint32_t lretcr
- volatile uint32_t pretcr

2.13.3.7.2.7.1 Field Documentation

2.13.3.7.2.7.2 volatile uint32_t rapidio_general_map_t::llcr

Logical layer configuration register (SRIO_LLCR)

2.13.3.7.2.7.3 volatile uint32_t rapidio_general_map_t::epwisr

Error / port-write interrupt status register (SRIO_EPWISR)

2.13.3.7.2.7.4 volatile uint32_t rapidio_general_map_t::lretcr

Logical retry error threshold configuration register (SRIO_LRETCR)

2.13.3.7.2.7.5 volatile uint32_t rapidio_general_map_t::pretcr

Physical retry error threshold configuration register (SRIO_PRETCR)

2.13.3.7.2.8 struct rapidio_rev_map_t

RapidIO Revision Registers.

Data Fields

- volatile uint32_t ipbrr [2]

2.13.3.7.2.8.1 Field Documentation

2.13.3.7.2.8.2 volatile uint32_t rapidio_rev_map_t::ipbrr[2]

IP Block Revision Register (SRIO_IPBRR1)

2.13.3.7.2.9 struct rapidio_atmu_out_t

RapidIO Outbound Address Translation Management Unit (ATMU) Registers.

Data Fields

- volatile uint32_t rowtar
- volatile uint32_t rowtear
- volatile uint32_t rowbar
- volatile uint32_t rowar

- volatile uint32_t rowsr [3]

2.13.3.7.2.9.1 Field Documentation

2.13.3.7.2.9.2 volatile uint32_t rapidio_atmu_out_t::rowtar

RapidIO outbound window translation address (SRIO_P1ROWTAR0)

2.13.3.7.2.9.3 volatile uint32_t rapidio_atmu_out_t::rowtear

RapidIO outbound window translation extended address (SRIO_P1ROWTEAR0)

2.13.3.7.2.9.4 volatile uint32_t rapidio_atmu_out_t::rowbar

RapidIO outbound window base address.

2.13.3.7.2.9.5 volatile uint32_t rapidio_atmu_out_t::rowar

RapidIO outbound window attributes.

2.13.3.7.2.9.6 volatile uint32_t rapidio_atmu_out_t::rowsr[3]

RapidIO outbound window segments.

2.13.3.7.2.10 struct rapidio_atmu_in_t

RapidIO Inbond Address Translation Management Unit (ATMU) Registers.

Data Fields

- volatile uint32_t riwtar
- volatile uint32_t riwbar
- volatile uint32_t riwar

2.13.3.7.2.10.1 Field Documentation

2.13.3.7.2.10.2 volatile uint32_t rapidio_atmu_in_t::riwtar

RapidIO Inbound window translation address register n (SRIO_P1RIWTAR4)

2.13.3.7.2.10.3 volatile uint32_t rapidio_atmu_in_t::riwbar

RapidIO Inbound window base address register n (SRIO_P1RIWBAR4)

2.13.3.7.2.10.4 volatile uint32_t rapidio_atmu_in_t::riwar

RapidIO inbound window attributes register n (SRIO_P1RIWAR4)

2.13.3.7.2.11 struct rapidio_atmu_map_t

RapidIO Address Translation Management Unit (ATMU) Registers.

2.13.3.7.2.12 struct srio_memmap_t

RapidIO Memory Map.

Data Fields

- rapidio_arch_map_t rapidio_arch
- rapidio_1xlp_serial_map_t rapidio_1xlp_serial
- rapidio_err_log_map_t rapidio_err_log
- rapidio_err_phy_map_t rapidio_err_phy [NUMBER_OF_SRIO_PHY_PORTS]
- rapidio_general_map_t rapidio_general
- rapidio_rev_map_t rapidio_rev
- rapidio_atmu_map_t rapidio_atmu [NUMBER_OF_SRIO_PHY_PORTS]

2.13.3.7.2.12.1 Field Documentation

2.13.3.7.2.12.2 rapidio_arch_map_t srio_memmap_t::rapidio_arch

See rapidio_arch_map_t.

2.13.3.7.2.12.3 rapidio_1xlp_serial_map_t srio_memmap_t::rapidio_1xlp_serial

See rapidio_1xlp_serial_map_t.

2.13.3.7.2.12.4 rapidio_err_log_map_t srio_memmap_t::rapidio_err_log

See rapidio_err_log_map_t.

2.13.3.7.2.12.5 rapidio_err_phy_map_t srio_memmap_t::rapidio_err_phy[NUMBER_OF_SRIO↩ _PHY_PORTS]

Error phys.

2.13.3.7.2.12.6 rapidio_general_map_t srio_memmap_t::rapidio_general

See rapidio_general_map_t.

2.13.3.7.2.12.7 rapidio_rev_map_t srio_memmap_t::rapidio_rev

See rapidio_rev_map_t.

2.13.3.7.2.12.8 rapidio_atmu_map_t srio_memmap_t::rapidio_atmu[NUMBER_OF_SRIO_PH↩ Y_PORTS]

Rapidio ATMU.

2.13.3.8 Timers 16 bit Memory Map

2.13.3.8.1 Overview

Data Structures

- struct timer_map_t
- struct tmr_map_t

2.13.3.8.2 Data Structure Documentation

2.13.3.8.2.1 struct timer_map_t

SoC Timer Quadrate Registers.

Data Fields

- volatile uint16_t tmr_cmp1
- volatile uint16_t tmr_cmp2
- volatile uint16_t tmr_cap
- volatile uint16_t tmr_load
- volatile uint16_t tmr_hold
- volatile uint16_t tmr_cntr
- volatile uint16_t tmr_ctrl
- volatile uint16_t tmr_sctl
- volatile uint16_t tmr_cmpld1
- volatile uint16_t tmr_cmpld2
- volatile uint16_t tmr_comscr

2.13.3.8.2.1.1 Field Documentation

2.13.3.8.2.1.2 volatile uint16_t timer_map_t::tmr_cmp1

Timer channel Compare Register 1.

2.13.3.8.2.1.3 volatile uint16_t timer_map_t::tmr_cmp2

Timer channel Compare Register 2.

2.13.3.8.2.1.4 volatile uint16_t timer_map_t::tmr_cap

Timer channel Capture Register.

2.13.3.8.2.1.5 volatile uint16_t timer_map_t::tmr_load

Timer channel Load Register.

2.13.3.8.2.1.6 volatile uint16_t timer_map_t::tmr_hold

Timer channel Hold Register.

2.13.3.8.2.1.7 volatile uint16_t timer_map_t::tmr_cntr

Timer channel Counter Register.

2.13.3.8.2.1.8 volatile uint16_t timer_map_t::tmr_ctrl

Timer channel Control Register.

2.13.3.8.2.1.9 volatile uint16_t timer_map_t::tmr_sctl

Timer channel Status and Control Register.

2.13.3.8.2.1.10 volatile uint16_t timer_map_t::tmr_cmpld1

Timer channel Compare 1 Load Register.

2.13.3.8.2.1.11 volatile uint16_t timer_map_t::tmr_cmpld2

Timer channel Compare 2 Load Register.

2.13.3.8.2.1.12 volatile uint16_t timer_map_t::tmr_comscr

Timer channel Comparator Status and Control Register.

2.13.3.8.2.2 struct tmr_map_t

SoC Timer Registers.

Data Fields

- timer_map_t tmr [NUM_OF_HW_TIMERS_16b_PER_MODULE]

2.13.3.8.2.2.1 Field Documentation

2.13.3.8.2.2.2 timer_map_t tmr_map_t::tmr[NUM_OF_HW_TIMERS_16b_PER_MODULE]

Timer Channels Registers.

2.13.3.9 Timers 32 bit Memory Map

2.13.3.9.1 Overview

Data Structures

- struct timer_32b_map_t
- struct tmr32b_map_t

2.13.3.9.2 Data Structure Documentation

2.13.3.9.2.1 struct timer_32b_map_t

SoC 32 bit Timer registers.

Data Fields

- volatile uint32_t tmr_cmp1
- volatile uint32_t tmr_cmp2
- volatile uint32_t tmr_cap
- volatile uint32_t tmr_load
- volatile uint32_t tmr_hold
- volatile uint32_t tmr_cntr
- volatile uint32_t tmr_ctrl
- volatile uint32_t tmr_sctl
- volatile uint32_t tmr_cmpld1
- volatile uint32_t tmr_cmpld2
- volatile uint32_t tmr_comscr

2.13.3.9.2.1.1 Field Documentation

2.13.3.9.2.1.2 volatile uint32_t timer_32b_map_t::tmr_cmp1

Timer channel Compare Register 1.

2.13.3.9.2.1.3 volatile uint32_t timer_32b_map_t::tmr_cmp2

Timer channel Compare Register 2.

2.13.3.9.2.1.4 volatile uint32_t timer_32b_map_t::tmr_cap

Timer channel Capture Register.

2.13.3.9.2.1.5 volatile uint32_t timer_32b_map_t::tmr_load

Timer channel Load Register.

2.13.3.9.2.1.6 volatile uint32_t timer_32b_map_t::tmr_hold

Timer channel Hold Register.

2.13.3.9.2.1.7 volatile uint32_t timer_32b_map_t::tmr_cntr

Timer channel Counter Register.

2.13.3.9.2.1.8 volatile uint32_t timer_32b_map_t::tmr_ctrl

Timer channel Control Register.

2.13.3.9.2.1.9 volatile uint32_t timer_32b_map_t::tmr_sctl

Timer channel Status and Control Register.

2.13.3.9.2.1.10 volatile uint32_t timer_32b_map_t::tmr_cmpld1

Timer channel Compare 1 Load Register.

2.13.3.9.2.1.11 volatile uint32_t timer_32b_map_t::tmr_cmpld2

Timer channel Compare 2 Load Register.

2.13.3.9.2.1.12 volatile uint32_t timer_32b_map_t::tmr_comscr

Timer channel Comparator Status and Control Register.

2.13.3.9.2.2 struct tmr32b_map_t

SoC 32 bit Timer Module.

Data Fields

- timer_32b_map_t tmr [NUM_OF_HW_TIMERS_32b_PER_MODULE]
- volatile uint32_t tmr_glb
- volatile uint32_t tmr_glbctl
- volatile uint32_t tmr_saf
- volatile uint32_t tmr_clrl

2.13.3.9.2.2.1 Field Documentation

2.13.3.9.2.2.2 timer_32b_map_t tmr32b_map_t::tmr[NUM_OF_HW_TIMERS_32b_PER_MODULE]

32 bit Timer Channels Registers

2.13.3.9.2.2.3 volatile uint32_t tmr32b_map_t::tmr_glb

Timer_32b Global System Timer Register.

2.13.3.9.2.2.4 volatile uint32_t tmr32b_map_t::tmr_glbctl

Timer_32b Global System Timer Control Register.

2.13.3.9.2.2.5 volatile uint32_t tmr32b_map_t::tmr_saf

Timer_32b Timer Set and Forget Register.

2.13.3.9.2.2.6 volatile uint32_t tmr32b_map_t::tmr_clrl

Timer_32b Timer Clear Lock Register.

2.13.3.10 Watchdog timer Memory Map

2.13.3.10.1 Overview

Data Structures

- struct wdt_map_t

2.13.3.10.2 Data Structure Documentation

2.13.3.10.2.1 struct wdt_map_t

Watch Dog Timer 16 bit (WDT) Registers.

Data Fields

- volatile uint32_t swcrr
- volatile uint32_t swcnr
- volatile uint16_t swsrr

2.13.3.10.2.1.1 Field Documentation

2.13.3.10.2.1.2 volatile uint32_t wdt_map_t::swcrr

System Watchdog Control Register.

2.13.3.10.2.1.3 volatile uint32_t wdt_map_t::swcnr

System Watchdog Count Register.

2.13.3.10.2.1.4 volatile uint16_t wdt_map_t::swsrr

System Watchdog Service Register.

2.13.4 SC39XX Memory Maps

2.13.4.1 Overview

Modules

- SC39XX Platform Memory Maps
- SC39XX Cluster Memory Maps

2.13.4.2 SC39XX Platform Memory Maps

2.13.4.2.1 Overview

Data Structures

- struct dsp_plat_mmu_map_t
- struct dsp_plat_cme_map_t
- struct dsp_plat_epic_map_t
- struct run_control_map_t
- struct detector_map_t
- struct address_and_data_detection_map_t
- struct indirect_event_unit_map_t
- struct triad_unit_map_t
- struct profiling_unit_map_t
- struct trace_unit_map_t
- struct dsp_plat_dtu_map_t
- struct dsp_plat_timer_t
- struct dsp_plat_timer_map_t
- struct dsp_plat_map_t

2.13.4.2.2 Data Structure Documentation

2.13.4.2.2.1 struct dsp_plat_mmu_map_t

Memory Management Unit (MMU) Registers.

Data Fields

- volatile uint32_t m_cr
- volatile uint32_t m_dvpc
- volatile uint32_t m_dva
- volatile uint32_t m_dsr
- volatile uint32_t m_ndvr
- volatile uint32_t m_ndsr
- volatile uint32_t m_pir
- volatile uint32_t m_dbl
- volatile uint32_t m_dbe
- volatile uint32_t m_desra0
- volatile uint32_t m_desra1
- volatile uint32_t m_desrs
- volatile uint32_t msg_pir
- volatile uint32_t msg_gpir
- volatile uint32_t msg_lpidr
- volatile uint32_t ccsr_base
- volatile uint32_t m_dsdval
- volatile uint32_t m_dsdmask
- volatile uint32_t m_dsda_pl
- volatile uint32_t m_dsdb_pl
- volatile uint32_t m_dsdc_pl
- volatile uint32_t m_dsdm_pl
- volatile uint32_t m_dsda_pl_sb
- volatile uint32_t m_dsdb_pl_sb

- volatile uint32_t m_dsdc_pl_sb
- volatile uint32_t m_dsdm_pl_sb
- volatile uint32_t m_dmper
- volatile uint32_t m_wmcfg
- volatile uint32_t m_pva
- volatile uint32_t m_psr
- volatile uint32_t m_npvr
- volatile uint32_t m_npsr
- volatile uint32_t m_pesra0
- volatile uint32_t m_pesra1
- volatile uint32_t m_pesrs
- volatile uint32_t m_psdval
- volatile uint32_t m_psdmask
- volatile uint32_t m_psda_pl
- volatile uint32_t m_psdb_pl
- volatile uint32_t m_psdc_pl
- volatile uint32_t m_psdm_pl
- volatile uint32_t m_psda_pl_sb
- volatile uint32_t m_psdb_pl_sb
- volatile uint32_t m_psdc_pl_sb
- volatile uint32_t m_psdm_pl_sb
- volatile uint32_t m_pmper
- volatile uint32_t m_dsda
- volatile uint32_t m_dsdb
- volatile uint32_t m_dsdc
- volatile uint32_t m_psda
- volatile uint32_t m_psdb
- volatile uint32_t m_psdc

2.13.4.2.2.1.1 Field Documentation

2.13.4.2.2.1.2 volatile uint32_t dsp_plat_mmu_map_t::m_cr

MMU Control Register.

2.13.4.2.2.1.3 volatile uint32_t dsp_plat_mmu_map_t::m_dvpc

Data Violation PC Register.

2.13.4.2.2.1.4 volatile uint32_t dsp_plat_mmu_map_t::m_dva

Data Violation Address Register.

2.13.4.2.2.1.5 volatile uint32_t dsp_plat_mmu_map_t::m_dsr

Data Status Register.

2.13.4.2.2.1.6 volatile uint32_t dsp_plat_mmu_map_t::m_ndvr

Non-precise Data Violation Address Register.

2.13.4.2.2.1.7 volatile uint32_t dsp_plat_mmu_map_t::m_ndsr

Non-precise Data Error Status Register.

2.13.4.2.2.1.8 volatile uint32_t dsp_plat_mmu_map_t::m_pir

Platform Information Register.

2.13.4.2.2.1.9 volatile uint32_t dsp_plat_mmu_map_t::m_dbl

Doorbell Level Register.

2.13.4.2.2.1.10 volatile uint32_t dsp_plat_mmu_map_t::m_dbe

Doorbell Edge Register.

2.13.4.2.2.1.11 volatile uint32_t dsp_plat_mmu_map_t::m_desra0

Data Exception Service Routine Address0.

2.13.4.2.2.1.12 volatile uint32_t dsp_plat_mmu_map_t::m_desra1

Data Exception Service Routine Address1.

2.13.4.2.2.1.13 volatile uint32_t dsp_plat_mmu_map_t::m_desrs

Data Exception Service Routine Select.

2.13.4.2.2.1.14 volatile uint32_t dsp_plat_mmu_map_t::msg_pir

Message Proccess ID Register.

2.13.4.2.2.1.15 volatile uint32_t dsp_plat_mmu_map_t::msg_gpir

Message Guest Proccess ID Register.

2.13.4.2.2.1.16 volatile uint32_t dsp_plat_mmu_map_t::msg_lpidr

Message Logical Proccess ID Register.

2.13.4.2.2.1.17 volatile uint32_t dsp_plat_mmu_map_t::ccsr_base

CCSR Base Address Register.

2.13.4.2.2.1.18 volatile uint32_t dsp_plat_mmu_map_t::m_dsdval

Data Segment Descriptor VAL.

2.13.4.2.2.1.19 volatile uint32_t dsp_plat_mmu_map_t::m_dsdmask

Data Segment Descriptor MASK.

2.13.4.2.2.1.20 volatile uint32_t dsp_plat_mmu_map_t::m_dsda_pl

Core Preload Register for Data Segment Descriptor A.

2.13.4.2.2.1.21 volatile uint32_t dsp_plat_mmu_map_t::m_dsdb_pl

Core Preload Register for Data Segment Descriptor B.

2.13.4.2.2.1.22 volatile uint32_t dsp_plat_mmu_map_t::m_dsdc_pl

Core Preload Register for Data Segment Descriptor C.

2.13.4.2.2.1.23 volatile uint32_t dsp_plat_mmu_map_t::m_dsdm_pl

Core Preload Register for Data Segment Descriptor M.

2.13.4.2.2.1.24 volatile uint32_t dsp_plat_mmu_map_t::m_dsda_pl_sb

Slave Preload Register for Data Segment Descriptor A.

2.13.4.2.2.1.25 volatile uint32_t dsp_plat_mmu_map_t::m_dsdb_pl_sb

Slave Preload Register for Data Segment Descriptor B.

2.13.4.2.2.1.26 volatile uint32_t dsp_plat_mmu_map_t::m_dsdc_pl_sb

Slave Preload Register for Data Segment Descriptor C.

2.13.4.2.2.1.27 volatile uint32_t dsp_plat_mmu_map_t::m_dsdm_pl_sb

Slave Preload Register for Data Segment Descriptor M.

2.13.4.2.2.1.28 volatile uint32_t dsp_plat_mmu_map_t::m_dmper

Data MATT Programming Error Register.

2.13.4.2.2.1.29 volatile uint32_t dsp_plat_mmu_map_t::m_wmcfg

SGB Watermark preload value.

2.13.4.2.2.1.30 volatile uint32_t dsp_plat_mmu_map_t::m_dsda

Data Segment Descriptor Registers A0.

2.13.4.2.2.1.31 volatile uint32_t dsp_plat_mmu_map_t::m_dsdb

Data Segment Descriptor Registers B0.

2.13.4.2.2.1.32 volatile uint32_t dsp_plat_mmu_map_t::m_dsdc

Data Segment Descriptor Registers C0.

2.13.4.2.2.1.33 volatile uint32_t dsp_plat_mmu_map_t::m_pva

Program Violation Address Register.

2.13.4.2.2.1.34 volatile uint32_t dsp_plat_mmu_map_t::m_psr

Program Status Register.

2.13.4.2.2.1.35 volatile uint32_t dsp_plat_mmu_map_t::m_npvr

Non-precise Program Violation Address Register.

2.13.4.2.2.1.36 volatile uint32_t dsp_plat_mmu_map_t::m_npsr

Non-precise Program Error Status Register.

2.13.4.2.2.1.37 volatile uint32_t dsp_plat_mmu_map_t::m_pesra0

Program Exception Service Routine Address0.

2.13.4.2.2.1.38 volatile uint32_t dsp_plat_mmu_map_t::m_pesra1

Program Exception Service Routine Address1.

2.13.4.2.2.1.39 volatile uint32_t dsp_plat_mmu_map_t::m_pesrs

Program Exception Service Routine Select.

2.13.4.2.2.1.40 volatile uint32_t dsp_plat_mmu_map_t::m_psdval

Program Segment Descriptor VAL.

2.13.4.2.2.1.41 volatile uint32_t dsp_plat_mmu_map_t::m_psdmask

Program Segment Descriptor MASK.

2.13.4.2.2.1.42 volatile uint32_t dsp_plat_mmu_map_t::m_psda_pl

Core Preload Register for Program Segment Descriptor A.

2.13.4.2.2.1.43 volatile uint32_t dsp_plat_mmu_map_t::m_psdb_pl

Core Preload Register for Program Segment Descriptor B.

2.13.4.2.2.1.44 volatile uint32_t dsp_plat_mmu_map_t::m_psdc_pl

Core Preload Register for Program Segment Descriptor C.

2.13.4.2.2.1.45 volatile uint32_t dsp_plat_mmu_map_t::m_psdm_pl

Core Preload Register for Program Segment Descriptor M.

2.13.4.2.2.1.46 volatile uint32_t dsp_plat_mmu_map_t::m_psda_pl_sb

Slave Preload Register for Program Segment Descriptor A.

2.13.4.2.2.1.47 volatile uint32_t dsp_plat_mmu_map_t::m_psdb_pl_sb

Slave Preload Register for Program Segment Descriptor B.

2.13.4.2.2.1.48 volatile uint32_t dsp_plat_mmu_map_t::m_psdc_pl_sb

Slave Preload Register for Program Segment Descriptor C.

2.13.4.2.2.1.49 volatile uint32_t dsp_plat_mmu_map_t::m_psdm_pl_sb

Slave Preload Register for Program Segment Descriptor M.

2.13.4.2.2.1.50 volatile uint32_t dsp_plat_mmu_map_t::m_pmper

Program MATT Programming Error Register.

2.13.4.2.2.1.51 volatile uint32_t dsp_plat_mmu_map_t::m_psda

Program Segment Descriptor Registers A0.

2.13.4.2.2.1.52 volatile uint32_t dsp_plat_mmu_map_t::m_psdb

Program Segment Descriptor Registers B0.

2.13.4.2.2.1.53 volatile uint32_t dsp_plat_mmu_map_t::m_psdc

Program Segment Descriptor Registers C0.

2.13.4.2.2.2 struct dsp_plat_cme_map_t

Cache Management (CME) Registers.

Data Fields

- volatile uint32_t cme_ctr
- volatile uint32_t cme_dcc
- volatile uint32_t cme_dca
- volatile uint32_t cme_dcr
- volatile uint32_t cme_qcr
- volatile uint32_t cme_dqu1
- volatile uint32_t cme_dqu2
- volatile uint32_t cme_qcc
- volatile uint32_t cme_qca
- volatile uint32_t cme_qu1
- volatile uint32_t cme_qu2
- volatile uint32_t cme_cc
- volatile uint32_t cme_cs
- volatile uint32_t cme_ca
- volatile uint32_t cme_cr

- volatile uint32_t cme_dst
- volatile uint32_t cme_der
- volatile uint32_t cme_din
- volatile uint32_t cme_dms
- volatile uint32_t cme_pst
- volatile uint32_t cme_per
- volatile uint32_t cme_pin
- volatile uint32_t cme_pms
- volatile uint32_t cme_dc
- volatile uint32_t cme_ds
- volatile uint32_t cme_da
- volatile uint32_t cme_pc
- volatile uint32_t cme_ps
- volatile uint32_t cme_pa

2.13.4.2.2.2.1 Field Documentation

2.13.4.2.2.2.2 volatile uint32_t dsp_plat_cme_map_t::cme_ctr

CME Control Register.

2.13.4.2.2.2.3 volatile uint32_t dsp_plat_cme_map_t::cme_dcc

CME Debug Channel Control Programming Register.

2.13.4.2.2.2.4 volatile uint32_t dsp_plat_cme_map_t::cme_dca

CME Debug Channel Address Programming Register.

2.13.4.2.2.2.5 volatile uint32_t dsp_plat_cme_map_t::cme_dcr

CME Debug Status Register.

2.13.4.2.2.2.6 volatile uint32_t dsp_plat_cme_map_t::cme_qcr

CME External Query Status Register.

2.13.4.2.2.2.7 volatile uint32_t dsp_plat_cme_map_t::cme_dqu1

CME Debug Query Result Register 1.

2.13.4.2.2.2.8 volatile uint32_t dsp_plat_cme_map_t::cme_dqu2

CME Debug Query Result Register 2.

2.13.4.2.2.2.9 volatile uint32_t dsp_plat_cme_map_t::cme_qcc

CME External Query Control Programming Register.

2.13.4.2.2.2.10 volatile uint32_t dsp_plat_cme_map_t::cme_qca

CME External Query Address Programming Register.

2.13.4.2.2.2.11 volatile uint32_t dsp_plat_cme_map_t::cme_qu1

CME External Query Result Register 1.

2.13.4.2.2.2.12 volatile uint32_t dsp_plat_cme_map_t::cme_qu2

CME External Query Result Register 2.

2.13.4.2.2.2.13 volatile uint32_t dsp_plat_cme_map_t::cme_cc

CME Block Control Programming Register.

2.13.4.2.2.2.14 volatile uint32_t dsp_plat_cme_map_t::cme_cs

CME Block Stride Programming Register.

2.13.4.2.2.2.15 volatile uint32_t dsp_plat_cme_map_t::cme_ca

CME Block Address Programming Register.

2.13.4.2.2.2.16 volatile uint32_t dsp_plat_cme_map_t::cme_cr

CME Block Programming Status Register.

2.13.4.2.2.2.17 volatile uint32_t dsp_plat_cme_map_t::cme_dst

CME Data Status Register.

2.13.4.2.2.2.18 volatile uint32_t dsp_plat_cme_map_t::cme_der

CME Data Error Register.

2.13.4.2.2.2.19 volatile uint32_t dsp_plat_cme_map_t::cme_din

CME Data Interrupt Status Register.

2.13.4.2.2.2.20 volatile uint32_t dsp_plat_cme_map_t::cme_dms

CME Data External Doorbell Interrupt Status Register.

2.13.4.2.2.2.21 volatile uint32_t dsp_plat_cme_map_t::cme_dc

CME Data Channel Control 0 Register.

2.13.4.2.2.2.22 volatile uint32_t dsp_plat_cme_map_t::cme_ds

CME Data Channel Stride 0 Register.

2.13.4.2.2.2.23 volatile uint32_t dsp_plat_cme_map_t::cme_da

CME Data Channel Address 0 Register.

2.13.4.2.2.2.24 volatile uint32_t dsp_plat_cme_map_t::cme_pst

CME Program Status Register.

2.13.4.2.2.2.25 volatile uint32_t dsp_plat_cme_map_t::cme_per

CME Program Error Register.

2.13.4.2.2.2.26 volatile uint32_t dsp_plat_cme_map_t::cme_pin

CME Program Interrupt Status Register.

2.13.4.2.2.2.27 volatile uint32_t dsp_plat_cme_map_t::cme_pms

CME Program External Doorbell Interrupt Status Register.

2.13.4.2.2.2.28 volatile uint32_t dsp_plat_cme_map_t::cme_pc

CME Program Channel Control 1 Register.

2.13.4.2.2.2.29 volatile uint32_t dsp_plat_cme_map_t::cme_ps

CME Program Channel Stride 1 Register.

2.13.4.2.2.2.30 volatile uint32_t dsp_plat_cme_map_t::cme_pa

CME Program Channel Address 1 Register.

2.13.4.2.2.3 struct dsp_plat_epic_map_t

Extended Programmable Interrupt Controller (EPIC) Registers.

Data Fields

- volatile uint32_t p_ipl [NUMBER_OF_IPL_REGISTERS]
- volatile uint32_t p_disp [NUMBER_OF_DISP_REGISTERS]
- volatile uint32_t p_trgt [NUMBER_OF_TRGT_REGISTERS]
- volatile uint32_t p_elr [NUMBER_OF_ELR_REGSISTERS]
- volatile uint32_t p_ipr [NUMBER_OF_IPR_REGSISTERS]
- volatile uint32_t p_endis [NUMBER_OF_ENDIS_REGSISTERS]
- volatile uint32_t p_swii
- volatile uint32_t p_di

2.13.4.2.2.3.1 Field Documentation

2.13.4.2.2.3.2 volatile uint32_t dsp_plat_epic_map_t::p_ipl[NUMBER_OF_IPL_REGISTERS]

EPIC Interrupt Priority Level Registers.

2.13.4.2.2.3.3 volatile uint32_t dsp_plat_epic_map_t::p_disp[NUMBER_OF_DISP_REGISTERS]

EPIC Interrupt Dispatcher Selector Register.

2.13.4.2.2.3.4 volatile uint32_t dsp_plat_epic_map_t::p_trgt[NUMBER_OF_TRGT_REGISTERS]

EPIC Interrupt Dispatcher Target Register.

2.13.4.2.2.3.5 volatile uint32_t dsp_plat_epic_map_t::p_elr[NUMBER_OF_ELR_REGSISTERS]

EPIC Edge/Level Trigger Registers.

2.13.4.2.2.3.6 volatile uint32_t dsp_plat_epic_map_t::p_ipr[NUMBER_OF_IPR_REGSISTERS]

EPIC Interrupt Pending Registers.

2.13.4.2.2.3.7 volatile uint32_t dsp_plat_epic_map_t::p_endis[NUMBER_OF_ENDIS_REGSISTERS]

EPIC Enable/Disable Interrupts Registers.

2.13.4.2.2.3.8 volatile uint32_t dsp_plat_epic_map_t::p_swii

EPIC Software Induced Interrupt Register.

2.13.4.2.2.3.9 volatile uint32_t dsp_plat_epic_map_t::p_di

EPIC Disable Interrupts Register.

2.13.4.2.2.4 struct run_control_map_t

Run and Control (DTU) Registers.

Data Fields

- volatile uint32_t ccr [NUMBER_OF_CORE_COMMAND_REGISTERS]
- volatile uint32_t cccr
- volatile uint32_t ccd [NUMBER_OF_CORE_COMMAND_DATA_REGISTERS]
- volatile uint32_t pc_next
- volatile uint32_t rcr
- volatile uint32_t dmeer
- volatile uint32_t dmrsr
- volatile uint32_t dmcsr
- volatile uint32_t dhrrr
- volatile uint32_t dmrrr
- volatile uint32_t duicr
- volatile uint32_t drasr
- volatile uint32_t desr
- volatile uint32_t sasr
- volatile uint32_t dturev

2.13.4.2.2.4.1 Field Documentation

2.13.4.2.2.4.2 volatile uint32_t run_control_map_t::ccr[NUMBER_OF_CORE_COMMAND_REGISTERS]

Core Command Registers 3 through 0.

2.13.4.2.2.4.3 volatile uint32_t run_control_map_t::cccr

Core Command control Register.

2.13.4.2.2.4.4 volatile uint32_t run_control_map_t::ccd[NUMBER_OF_CORE_COMMAND_DATA_REGISTERS]

core Command Data register

2.13.4.2.2.4.5 volatile uint32_t run_control_map_t::pc_next

PC for next unexecuted VLES.

2.13.4.2.2.4.6 volatile uint32_t run_control_map_t::rcr

Run Control Register.

2.13.4.2.2.4.7 volatile uint32_t run_control_map_t::dmeer

Debug mode event enabling register.

2.13.4.2.2.4.8 volatile uint32_t run_control_map_t::dmrsr

Debug mode reason status register.

2.13.4.2.2.4.9 volatile uint32_t run_control_map_t::dmcsr

Debug mode control status register.

2.13.4.2.2.4.10 volatile uint32_t run_control_map_t::dhrrr

Debug host resource reservation register.

2.13.4.2.2.4.11 volatile uint32_t run_control_map_t::dmrrr

Debug monitor resources reservation register.

2.13.4.2.2.4.12 volatile uint32_t run_control_map_t::duicr

DTU interface control register.

2.13.4.2.2.4.13 volatile uint32_t run_control_map_t::drasr

Debug resources activity status register.

2.13.4.2.2.4.14 volatile uint32_t run_control_map_t::desr

Debug event status register.

2.13.4.2.2.4.15 volatile uint32_t run_control_map_t::sasr

Subsystem Activity Status Register.

2.13.4.2.2.4.16 volatile uint32_t run_control_map_t::dturev

Debug and Trace Unit Revision Register.

2.13.4.2.2.5 struct detector_map_t

Detector (DTU) Registers.

Data Fields

- volatile uint32_t ardcr
- volatile uint32_t depcr
- volatile uint32_t padrra
- volatile uint32_t padrrb

2.13.4.2.2.5.1 Field Documentation

2.13.4.2.2.5.2 volatile uint32_t detector_map_t::ardcr

Address Range Detector Control Register.

2.13.4.2.2.5.3 volatile uint32_t detector_map_t::depcr

Dual Exact PC Detector Control Register.

2.13.4.2.2.5.4 volatile uint32_t detector_map_t::padrra

PC & Address Detector Reference Register A.

2.13.4.2.2.5.5 volatile uint32_t detector_map_t::padrrb

PC & Address Detector Reference Register B.

2.13.4.2.2.6 struct address_and_data_detection_map_t

Address and Data Detection Unit (DTU) Registers.

Data Fields

- detector_map_t detector [4]

2.13.4.2.2.6.1 Field Documentation

2.13.4.2.2.6.2 detector_map_t address_and_data_detection_map_t::detector[4]

Detectors control and reference registers.

2.13.4.2.2.7 struct indirect_event_unit_map_t

Indirect Event unit (DTU) Registers.

Data Fields

- volatile uint32_t iectls
- indirect_event_cond_trans_map_t iect [NUMBER_OF_INDIRECT_EVENT_COND_TRASITI↩ ON_CONF_REGISTERS]
- volatile uint32_t ieutr

2.13.4.2.2.7.1 Field Documentation

2.13.4.2.2.7.2 volatile uint32_t indirect_event_unit_map_t::iectls

Indirect Control and Status Register.

2.13.4.2.2.7.3 indirect_event_cond_trans_map_t indirect_event_unit_map_t::iect[NUMBER_OF↩ _INDIRECT_EVENT_COND_TRASITION_CONF_REGISTERS]

Indirect Event Conditional Transition Configuration Register n.

2.13.4.2.2.7.4 volatile uint32_t indirect_event_unit_map_t::ieutr

Indirect Event Unconditional Transition Configuration Register.

2.13.4.2.2.8 struct triad_unit_map_t

Triad Unit (DTU) Registers.

Data Fields

- volatile uint32_t ptcr
- volatile uint32_t pcvr [NUMBER_OF_PROFILING_COUNTER_VALUE_REGISTERS_A]
- volatile uint32_t pcsr [NUMBER_OF_PROFILING_SNAPSHOT_REGISTERS_A]

2.13.4.2.2.8.1 Field Documentation

2.13.4.2.2.8.2 volatile uint32_t triad_unit_map_t::ptcr

Profiling Triad Control Register A.

2.13.4.2.2.8.3 volatile uint32_t triad_unit_map_t::pcvr[NUMBER_OF_PROFILING_COUNTE↩ R_VALUE_REGISTERS_A]

Profiling Counter Value Register An.

2.13.4.2.2.8.4 volatile uint32_t triad_unit_map_t::pcsr[NUMBER_OF_PROFILING_SNAPSHOT_REGISTERS_A]

Profiling Counter Snapshot Registers An.

2.13.4.2.2.9 struct profiling_unit_map_t

Profiling Unit (DTU) Registers.

Data Fields

- volatile uint32_t pccsr
- triad_unit_map_t triad [2]
- volatile uint32_t rccr0
- volatile uint32_t rcvr0
- volatile uint32_t rcrr0
- volatile uint32_t rcsr0

2.13.4.2.2.9.1 Field Documentation

2.13.4.2.2.9.2 volatile uint32_t profiling_unit_map_t::pccsr

Profiling Counters Control and Status Register.

2.13.4.2.2.9.3 triad_unit_map_t profiling_unit_map_t::triad[2]

Profiling triad Counters registers.

2.13.4.2.2.9.4 volatile uint32_t profiling_unit_map_t::rccr0

Reloadable Counter Control Register 0.

2.13.4.2.2.9.5 volatile uint32_t profiling_unit_map_t::rcvr0

Reloadable Counter Value Register 0.

2.13.4.2.2.9.6 volatile uint32_t profiling_unit_map_t::rcrr0

Reloadable Counter Reload Register 0.

2.13.4.2.2.9.7 volatile uint32_t profiling_unit_map_t::rcsr0

Reloadable Counter Snapshot Register 0.

2.13.4.2.2.10 struct trace_unit_map_t

Trace Unit (DTU) Registers.

Data Fields

- volatile uint32_t tc1
- volatile uint32_t trsr

Architectures

- volatile uint32_t tc3
- volatile uint32_t tc4
- volatile uint32_t tpmcr
- volatile uint32_t twmsk
- volatile uint32_t tmdati

2.13.4.2.2.10.1 Field Documentation

2.13.4.2.2.10.2 volatile uint32_t trace_unit_map_t::tc1

Trace Control Register 1.

2.13.4.2.2.10.3 volatile uint32_t trace_unit_map_t::trsr

Trace Status Register.

2.13.4.2.2.10.4 volatile uint32_t trace_unit_map_t::tc3

Trace Control Register 3.

2.13.4.2.2.10.5 volatile uint32_t trace_unit_map_t::tc4

Trace Control Register 4.

2.13.4.2.2.10.6 volatile uint32_t trace_unit_map_t::tpmcr

Trace Profiling Message Control Register.

2.13.4.2.2.10.7 volatile uint32_t trace_unit_map_t::twmsk

Trace Watchpoint Mask Register.

2.13.4.2.2.10.8 volatile uint32_t trace_unit_map_t::tmdati

Image of the TMDAT core register.

2.13.4.2.2.11 struct dsp_plat_dtu_map_t

Debug and Trace Unit (DTU) Registers.

2.13.4.2.2.12 struct dsp_plat_timer_t

Platform Timer Configuration MAP.

Data Fields

- volatile uint32_t tm_tc
- volatile uint32_t tm_tp

2.13.4.2.2.12.1 Field Documentation

2.13.4.2.2.12.2 volatile uint32_t dsp_plat_timer_t::tm_tc

Timer Control Register.

2.13.4.2.2.12.3 volatile uint32_t dsp_plat_timer_t::tm_tp

Timer Pre-load Register.

2.13.4.2.2.13 struct dsp_plat_timer_map_t

Platform Timer Registers.

Data Fields

- dsp_plat_timer_t platform_timer [4]
- volatile uint32_t tm_tv [4]
- volatile uint32_t tm_sc
- volatile uint32_t tm_s [2]

2.13.4.2.2.13.1 Field Documentation

2.13.4.2.2.13.2 dsp_plat_timer_t dsp_plat_timer_map_t::platform_timer[4]

Timers configuration registers.

2.13.4.2.2.13.3 volatile uint32_t dsp_plat_timer_map_t::tm_tv[4]

Timer value Register.

2.13.4.2.2.13.4 volatile uint32_t dsp_plat_timer_map_t::tm_sc

Timer shadow control.

2.13.4.2.2.13.5 volatile uint32_t dsp_plat_timer_map_t::tm_s[2]

Timer shadow value register.

2.13.4.2.2.14 struct dsp_plat_map_t

DSP Subsystem memory map.

Data Fields

- dsp_plat_dcache_map_t dcache
- dsp_plat_icache_map_t icache
- dsp_plat_dtu_map_t dtu
- dsp_plat_mmu_map_t mmu
- dsp_plat_cme_map_t cme
- dsp_plat_epic_map_t epic
- dsp_plat_timer_map_t timer

2.13.4.2.2.14.1 Field Documentation

2.13.4.2.2.14.2 dsp_plat_dcache_map_t dsp_plat_map_t::dcache

L1 Data Cache Registers.

2.13.4.2.2.14.3 dsp_plat_icache_map_t dsp_plat_map_t::icache

L1 Instruction Cache Registers.

2.13.4.2.2.14.4 dsp_plat_dtu_map_t dsp_plat_map_t::dtu

Debug and Trace Unit Registers.

2.13.4.2.2.14.5 dsp_plat_mmu_map_t dsp_plat_map_t::mmu

Memory Management Unit Registers.

2.13.4.2.2.14.6 dsp_plat_cme_map_t dsp_plat_map_t::cme

Cache management Unit Registers.

2.13.4.2.2.14.7 dsp_plat_epic_map_t dsp_plat_map_t::epic

Interrupt Controller Registers.

2.13.4.2.2.14.8 dsp_plat_timer_map_t dsp_plat_map_t::timer

Timer Registers.

2.13.4.3 SC39XX Cluster Memory Maps

2.13.4.3.1 Overview

Data Structures

- struct dsp_cluster_wdt_t
- struct l2_partition_id_t
- struct dsp_cluster_l2_map_t
- struct dsp_cluster_map_t

2.13.4.3.2 Data Structure Documentation

2.13.4.3.2.1 struct dsp_cluster_wdt_t

Cluster Watchdog Timer Configuration Map.

Data Fields

- volatile uint32_t wdtc
- volatile uint32_t wdtpl0
- volatile uint32_t wdtpl1
- volatile uint32_t wdtsc
- volatile uint32_t wdtsv0
- volatile uint32_t wdtsv1

2.13.4.3.2.1.1 Field Documentation

2.13.4.3.2.1.2 volatile uint32_t dsp_cluster_wdt_t::wdtc

WD Timer <n> Control Register.

2.13.4.3.2.1.3 volatile uint32_t dsp_cluster_wdt_t::wdtpl0

WD Timer <n> Pre-load Register 0.

2.13.4.3.2.1.4 volatile uint32_t dsp_cluster_wdt_t::wdtpl1

WD Timer <n> Pre-load Register 1.

2.13.4.3.2.1.5 volatile uint32_t dsp_cluster_wdt_t::wdtsc

WD Shadow Register <n> Control.

2.13.4.3.2.1.6 volatile uint32_t dsp_cluster_wdt_t::wdtsv0

WD Shadow Value <n> Register 0.

2.13.4.3.2.1.7 volatile uint32_t dsp_cluster_wdt_t::wdtsv1

WD Shadow Value <n> Register 1.

2.13.4.3.2.2 struct l2_partition_id_t

L2 partition control map.

Data Fields

- volatile uint32_t l2pir
- volatile uint32_t l2par
- volatile uint32_t l2pwr

2.13.4.3.2.2.1 Field Documentation

2.13.4.3.2.2.2 volatile uint32_t l2_partition_id_t::l2pir

L2 Cache Partitioning Identification Registers.

2.13.4.3.2.2.3 volatile uint32_t l2_partition_id_t::l2par

L2 Cache Partitioning Allocation Registers.

2.13.4.3.2.2.4 volatile uint32_t l2_partition_id_t::l2pwr

L2 Cache Partitioning Way Registers.

2.13.4.3.2.3 struct dsp_cluster_l2_map_t

DSP Cluster L2 control Map.

Data Fields

- volatile uint32_t l2csr0
- volatile uint32_t l2csr1
- volatile uint32_t l2cfg0
- volatile uint32_t l2errinjhi
- volatile uint32_t l2errinjlo
- volatile uint32_t l2errinjctl
- volatile uint32_t l2captdatahi
- volatile uint32_t l2captdatalo
- volatile uint32_t l2captecc
- volatile uint32_t l2errdet
- volatile uint32_t l2errdis
- volatile uint32_t l2errinten
- volatile uint32_t l2errattr
- volatile uint32_t l2erreaddr
- volatile uint32_t l2erraddr
- volatile uint32_t l2errctl

2.13.4.3.2.3.1 Field Documentation

2.13.4.3.2.3.2 volatile uint32_t dsp_cluster_l2_map_t::l2csr0

L2 Cache Control and Status Register 0.

2.13.4.3.2.3.3 volatile uint32_t dsp_cluster_l2_map_t::l2csr1

L2 Cache Control and Status Register 1.

2.13.4.3.2.3.4 volatile uint32_t dsp_cluster_l2_map_t::l2cfg0

L2 Cache Configuration Register 0.

2.13.4.3.2.3.5 volatile uint32_t dsp_cluster_l2_map_t::l2errinjhi

L2 Cache Error Injection Mask High Register.

2.13.4.3.2.3.6 volatile uint32_t dsp_cluster_l2_map_t::l2errinjlo

L2 Cache Error Injection Mask Low Register.

2.13.4.3.2.3.7 volatile uint32_t dsp_cluster_l2_map_t::l2errinjctl

L2 Cache Error Injection Mask Low Register.

2.13.4.3.2.3.8 volatile uint32_t dsp_cluster_l2_map_t::l2captdatahi

L2 Error Capture Data High Register.

2.13.4.3.2.3.9 volatile uint32_t dsp_cluster_l2_map_t::l2captdatalo

L2 Error Capture Data Low Register.

2.13.4.3.2.3.10 volatile uint32_t dsp_cluster_l2_map_t::l2captecc

L2 Cache Error Capture ECC Syndrome Register.

2.13.4.3.2.3.11 volatile uint32_t dsp_cluster_l2_map_t::l2errdet

L2 Cache Error Detect Register.

2.13.4.3.2.3.12 volatile uint32_t dsp_cluster_l2_map_t::l2errdis

L2 Cache Error Disable Register.

2.13.4.3.2.3.13 volatile uint32_t dsp_cluster_l2_map_t::l2errinten

L2 Cache Error Interrupt Enable Register.

2.13.4.3.2.3.14 volatile uint32_t dsp_cluster_l2_map_t::l2errattr

L2 Cache Error Attribute Register.

2.13.4.3.2.3.15 volatile uint32_t dsp_cluster_l2_map_t::l2erreaddr

L2 Error Extended Address Register.

2.13.4.3.2.3.16 volatile uint32_t dsp_cluster_l2_map_t::l2erraddr

L2 Cache Error Address Register.

2.13.4.3.2.3.17 volatile uint32_t dsp_cluster_l2_map_t::l2errctl

L2 Cache Error Control Register.

2.13.4.3.2.4 struct dsp_cluster_map_t

DSP Cluster memory map.

2.13.5 B4860 LAW

2.13.5.1 Overview

LAW initialization.

Data Structures

- struct law_init_params_t

Enumerations

- enum law_win_size_t {
  LAW_WIN_SIZE_ILLIGAL = 0, LAW_WIN_SIZE_4K = 0xB, LAW_WIN_SIZE_8K = 0xC,
  LAW_WIN_SIZE_16K = 0xD, LAW_WIN_SIZE_32K = 0xE, LAW_WIN_SIZE_64K = 0xF,
  LAW_WIN_SIZE_128K = 0x10, LAW_WIN_SIZE_256K = 0x11, LAW_WIN_SIZE_512K = 0x12,
  LAW_WIN_SIZE_1M = 0x13, LAW_WIN_SIZE_2M = 0x14, LAW_WIN_SIZE_4M = 0x15,
  LAW_WIN_SIZE_8M = 0x16, LAW_WIN_SIZE_16M = 0x17, LAW_WIN_SIZE_32M = 0x18,
  LAW_WIN_SIZE_64M = 0x19, LAW_WIN_SIZE_128M = 0x1A, LAW_WIN_SIZE_256M = 0x1B,
  LAW_WIN_SIZE_512M = 0x1C, LAW_WIN_SIZE_1G = 0x1D, LAW_WIN_SIZE_2G = 0x1E,
  LAW_WIN_SIZE_4G = 0x1F, LAW_WIN_SIZE_8G = 0x20, LAW_WIN_SIZE_16G = 0x21,
  LAW_WIN_SIZE_32G = 0x22, LAW_WIN_SIZE_64G = 0x23 }
- enum law_win_id_t
- enum law_target_id_t

Functions

- os_status osLawInitialize (struct law_init_params_s *soc_law_init_params, unsigned int num_↩ windows)
- os_status b486xLawWinAdd (law_win_id_t law_index, uint64_t law_addr, law_win_size_t law_↩ size, law_target_id_t target_id, uint32_t csd_id)
- void b486xLawWinDisable (law_win_id_t law_index)
- os_status b486xLawWinFindTid (law_target_id_t target_id, law_win_size_t *size, uint64_t *base↩ _addr)

2.13.5.2 Data Structure Documentation

2.13.5.2.1 struct law_init_params_t

Initialization structure of LAW.

Data Fields

- unsigned int num_windows

2.13.5.2.1.1 Field Documentation

2.13.5.2.1.1.1 unsigned int law_init_params_t::num_windows

Number of law windows to initialize, equals number of element in law_params[].

2.13.5.3 Enumeration Type Documentation

2.13.5.3.1 enum law_win_size_t

LAW window size (Local Access Window size = 2^(SIZE+1))

Enumerator

LAW_WIN_SIZE_ILLIGAL LAW window, illigal size.
LAW_WIN_SIZE_4K LAW window, size is 4K.
LAW_WIN_SIZE_8K LAW window, size is 8K.
LAW_WIN_SIZE_16K LAW window, size is 16K.
LAW_WIN_SIZE_32K LAW window, size is 32K.
LAW_WIN_SIZE_64K LAW window, size is 64K.
LAW_WIN_SIZE_128K LAW window, size is 128K.
LAW_WIN_SIZE_256K LAW window, size is 256K.
LAW_WIN_SIZE_512K LAW window, size is 512K.
LAW_WIN_SIZE_1M LAW window, size is 1M.
LAW_WIN_SIZE_2M LAW window, size is 2M.
LAW_WIN_SIZE_4M LAW window, size is 4M.
LAW_WIN_SIZE_8M LAW window, size is 8M.
LAW_WIN_SIZE_16M LAW window, size is 16M.
LAW_WIN_SIZE_32M LAW window, size is 32M.
LAW_WIN_SIZE_64M LAW window, size is 64M.
LAW_WIN_SIZE_128M LAW window, size is 128M.
LAW_WIN_SIZE_256M LAW window, size is 256M.
LAW_WIN_SIZE_512M LAW window, size is 512M.
LAW_WIN_SIZE_1G LAW window, size is 1G.
LAW_WIN_SIZE_2G LAW window, size is 2G.
LAW_WIN_SIZE_4G LAW window, size is 4G.
LAW_WIN_SIZE_8G LAW window, size is 8G.
LAW_WIN_SIZE_16G LAW window, size is 16G.
LAW_WIN_SIZE_32G LAW window, size is 32G.
LAW_WIN_SIZE_64G LAW window, size is 64G.

2.13.5.3.2 enum law_win_id_t

LAW window ID.

2.13.5.3.3 enum law_target_id_t

LAW target ID.

2.13.5.4 Function Documentation

2.13.5.4.1 os_status osLawInitialize (struct law_init_params_s * *soc_law_init_params,* unsigned int *num_windows*)

LAW initialization method

Parameters

in	*soc_law_init_params*	- Initialization parameters for the LAW windows
in	*num_windows*	- Number of LAW windows to initiate

Returns

OS status.

2.13.5.4.2 os_status b486xLawWinAdd (law_win_id_t *law_index,* uint64_t *law_addr,* law_win_size_t *law_size,* law_target_id_t *target_id,* uint32_t *csd_id*)

Add a new LAW window

Parameters

in	*law_index*	- LAW index
in	*law_addr*	- physical address of LAW window
in	*law_size*	- size of LAW window
in	*target_id*	- Target ID

in	*csd_id*	- Coherency subdomain identifier

Returns

OS status.

2.13.5.4.3 void b486xLawWinDisable (law_win_id_t *law_index*)

Disable given LAW configuration

Parameters

in	*law_index*	- LAW index

Returns

OS status.

2.13.5.4.4 os_status b486xLawWinFindTid (law_target_id_t *target_id*, law_win_size_t * *size*, uint64_t * *base_addr*)

find a LAW with a given target id

Parameters

in	*base_addr*	- physical address of LAW window
in	*size*	- size of LAW window
in	*target_id*	- Target ID

Returns

OS status.

2.13.6 DSP Platform Definitions

2.13.6.1 Overview

Modules

- SC39XX Definitions

2.13.7 SmartDsp OS Devices

2.13.7.1 Overview

Modules

- B4860 SmartDsp OS Devices

2.13.8 Hardware Semaphore API

2.13.8.1 Overview

Macros

- #define hwSemaphoreIsTaken(sem_num) (hwSemaphoreGetVal(sem_num) != 0)
- #define hwSemaphoreIsFree(sem_num) (hwSemaphoreGetVal(sem_num) == 0)

Functions

- void hwSemaphoreTake (int sem_num, uint8_t val)
- void hwSemaphoreIrqTake (int sem_num, uint8_t val)
- void hwSemaphoreRelease (int sem_num)
- void hwSemaphoreIrqRelease (int sem_num)
- os_status hwSemaphoreTryTake (int sem_num, uint8_t val)
- uint8_t hwSemaphoreGetVal (int sem_num)
- os_status hwSemaphoreInitialize ()

2.13.8.2 Macro Definition Documentation

2.13.8.2.1 #define hwSemaphoreIsTaken(*sem_num*) (hwSemaphoreGetVal(sem_num) != 0)

Checks if hardware semaphore is taken.

Parameters

in	*sem_num*	- Number of the semaphore to check.

Return values

TRUE	- Semaphore is taken

FALSE	- Semaphore is not taken

2.13.8.2.2 #define hwSemaphoreIsFree(*sem_num*) (hwSemaphoreGetVal(sem_num) == 0)

Checks if hardware semaphore is free.

Parameters

in	*sem_num*	- Number of the semaphore to check.

Return values

FALSE	- Semaphore is not taken
TRUE	- Semaphore is taken

2.13.8.3 Function Documentation

2.13.8.3.1 void hwSemaphoreTake (int *sem_num,* uint8_t *val*)

Performs a busy wait trying to acquire a hardware semaphore.

Parameters

in	*sem_num*	- Number of the semaphore to take.
in	*val*	- Value to write to the semaphore (0x01 - 0xFF)

2.13.8.3.2 void hwSemaphoreIrqTake (int *sem_num,* uint8_t *val*)

Disables interrupts and then performs a busy wait trying to acquire a hardware semaphore.

Parameters

in	*sem_num*	- Number of the semaphore to take.
in	*val*	- Value to write to the semaphore (0x01 - 0xFF)

2.13.8.3.3 void hwSemaphoreRelease (int *sem_num*)

Releases a hardware semaphore.

Parameters

in	*sem_num*	- Number of the semaphore to release.

Warning

This function doesn't validate the the software entity releasing the semaphore is the same as that acquired it

2.13.8.3.4 void hwSemaphoreIrqRelease (int *sem_num*)

Releases a hardware semaphore and then enables interrupts

Parameters

in	*sem_num*	- Number of the semaphore to release.

Warning

This function doesn't validate the the software entity releasing the semaphore is the same as that acquired it

2.13.8.3.5 os_status hwSemaphoreTryTake (int *sem_num,* uint8_t *val*)

Try taking a hardware semaphore.

Parameters

in	*sem_num*	- Number of the semaphore to take.
in	*val*	- Value to write to the semaphore (0x01 - 0xFF)

Return values

OS_SUCCESS	- Semaphore taken successfully
OS_FAIL	- Semaphore already taken by other entity

2.13.8.3.6 uint8_t hwSemaphoreGetVal (int *sem_num*)

Read the value currently written in a hardware semaphore.

Parameters

in	*sem_num*	- Number of the semaphore to read.

Returns

Value currently written

2.13.8.3.7 os_status hwSemaphoreInitialize ( )

Initializes the hardware semaphore module.

Return values

OS_SUCCESS	- The multicore synchronization module was initialized successfully.

2.13.9 GPIO (general purpose I/O) API

2.13.10 IPC (Inter Process Communication) API

2.13.10.1 Overview

Modules

- B486x IPC (Inter Process Communication) API

2.13.11 Heterogeneous API

2.13.11.1 Overview

Modules

- B486x Heterogeneous API
- B486x Heterogeneous Common API
- B486x Heterogeneous Debug API
- B486x Heterogeneous Debug print API
- B486x Heterogeneous IPC API
- B486x L1 defence heterogeneous API
- B486x Heterogeneous Memory Descriptor structure API

2.13.12 HW Interrupts Map

2.13.12.1 Overview

Modules

- B4860 HW Interrupts

2.13.12.2 B4860 HW Interrupts

2.13.12.2.1 Overview

Macros

- #define EPIC_INTERRUPTS_OFFSET 14
- #define EPIC_EDGE_INTERRUPT_START (EPIC_INTERRUPTS_OFFSET + 48)
- #define EPIC_EDGE_INTERRUPT_END (EPIC_INTERRUPTS_OFFSET + 95)
- #define EPIC_INTERNAL_INTERRUPT_END (EPIC_INTERRUPTS_OFFSET + 0x1F)
- #define OS_HWI_DEFAULT_DISPATCHER osHwiFullScheduler
- #define OS_SWI_DEFALT_DISPATCHER osSwiTaskDispatcher
- #define OS_TASK_DEFALT_DISPATCHER osHwiTaskScheduler

EPIC Intercore interrupts

- #define OS_INT_FROM_CORE0_0 (EPIC_INTERRUPTS_OFFSET + 32)
- #define OS_INT_FROM_CORE0_1 (EPIC_INTERRUPTS_OFFSET + 33)
- #define OS_INT_FROM_CORE1_0 (EPIC_INTERRUPTS_OFFSET + 34)
- #define OS_INT_FROM_CORE1_1 (EPIC_INTERRUPTS_OFFSET + 35)
- #define OS_INT_FROM_CORE2_0 (EPIC_INTERRUPTS_OFFSET + 36)
- #define OS_INT_FROM_CORE2_1 (EPIC_INTERRUPTS_OFFSET + 37)
- #define OS_INT_FROM_CORE3_0 (EPIC_INTERRUPTS_OFFSET + 38)
- #define OS_INT_FROM_CORE3_1 (EPIC_INTERRUPTS_OFFSET + 39)
- #define OS_INT_FROM_CORE4_0 (EPIC_INTERRUPTS_OFFSET + 40)
- #define OS_INT_FROM_CORE4_1 (EPIC_INTERRUPTS_OFFSET + 41)
- #define OS_INT_FROM_CORE5_0 (EPIC_INTERRUPTS_OFFSET + 42)
- #define OS_INT_FROM_CORE5_1 (EPIC_INTERRUPTS_OFFSET + 43)

EPIC VIRQ Interrupts

- #define OS_INT_VIRQ0 (EPIC_INTERRUPTS_OFFSET + 64)
- #define OS_INT_VIRQ1 (EPIC_INTERRUPTS_OFFSET + 65)
- #define OS_INT_VIRQ2 (EPIC_INTERRUPTS_OFFSET + 66)
- #define OS_INT_VIRQ3 (EPIC_INTERRUPTS_OFFSET + 67)
- #define OS_INT_VIRQ4 (EPIC_INTERRUPTS_OFFSET + 68)
- #define OS_INT_VIRQ5 (EPIC_INTERRUPTS_OFFSET + 69)
- #define OS_INT_VIRQ6 (EPIC_INTERRUPTS_OFFSET + 70)
- #define OS_INT_VIRQ7 (EPIC_INTERRUPTS_OFFSET + 71)
- #define OS_INT_VIRQ8 (EPIC_INTERRUPTS_OFFSET + 72)
- #define OS_INT_VIRQ9 (EPIC_INTERRUPTS_OFFSET + 73)
- #define OS_INT_VIRQ10 (EPIC_INTERRUPTS_OFFSET + 74)

- #define OS_INT_VIRQ11 (EPIC_INTERRUPTS_OFFSET + 75)
- #define OS_INT_VIRQ12 (EPIC_INTERRUPTS_OFFSET + 76)
- #define OS_INT_VIRQ13 (EPIC_INTERRUPTS_OFFSET + 77)
- #define OS_INT_VIRQ14 (EPIC_INTERRUPTS_OFFSET + 78)
- #define OS_INT_VIRQ15 (EPIC_INTERRUPTS_OFFSET + 79)
- #define OS_INT_VIRQ16 (EPIC_INTERRUPTS_OFFSET + 80)
- #define OS_INT_VIRQ17 (EPIC_INTERRUPTS_OFFSET + 81)
- #define OS_INT_VIRQ18 (EPIC_INTERRUPTS_OFFSET + 82)
- #define OS_INT_VIRQ19 (EPIC_INTERRUPTS_OFFSET + 83)
- #define OS_INT_VIRQ20 (EPIC_INTERRUPTS_OFFSET + 84)
- #define OS_INT_VIRQ21 (EPIC_INTERRUPTS_OFFSET + 85)
- #define OS_INT_VIRQ22 (EPIC_INTERRUPTS_OFFSET + 86)
- #define OS_INT_VIRQ23 (EPIC_INTERRUPTS_OFFSET + 87)
- #define OS_INT_VIRQ24 (EPIC_INTERRUPTS_OFFSET + 88)
- #define OS_INT_VIRQ25 (EPIC_INTERRUPTS_OFFSET + 89)
- #define OS_INT_VIRQ26 (EPIC_INTERRUPTS_OFFSET + 90)
- #define OS_INT_VIRQ27 (EPIC_INTERRUPTS_OFFSET + 91)
- #define OS_INT_VIRQ28 (EPIC_INTERRUPTS_OFFSET + 92)
- #define OS_INT_VIRQ29 (EPIC_INTERRUPTS_OFFSET + 93)
- #define OS_INT_VIRQ30 (EPIC_INTERRUPTS_OFFSET + 94)
- #define OS_INT_VIRQ31 (EPIC_INTERRUPTS_OFFSET + 95)

EPIC MAPLE-B3 interrupts

- #define OS_INT_MAPLE_0_CH_0 (EPIC_INTERRUPTS_OFFSET + 282)
- #define OS_INT_MAPLE_0_CH_1 (OS_INT_MAPLE_0_CH_0 + 1)
- #define OS_INT_MAPLE_0_CH_2 (OS_INT_MAPLE_0_CH_1 + 1)
- #define OS_INT_MAPLE_0_CH_3 (OS_INT_MAPLE_0_CH_2 + 1)
- #define OS_INT_MAPLE_0_CH_4 (OS_INT_MAPLE_0_CH_3 + 1)
- #define OS_INT_MAPLE_0_CH_5 (OS_INT_MAPLE_0_CH_4 + 1)
- #define OS_INT_MAPLE_0_CH_6 (OS_INT_MAPLE_0_CH_5 + 1)
- #define OS_INT_MAPLE_0_CH_7 (OS_INT_MAPLE_0_CH_6 + 1)
- #define OS_INT_MAPLE_0_CH_8 (OS_INT_MAPLE_0_CH_7 + 1)
- #define OS_INT_MAPLE_0_CH_9 (OS_INT_MAPLE_0_CH_8 + 1)
- #define OS_INT_MAPLE_0_CH_10 (OS_INT_MAPLE_0_CH_9 + 1)
- #define OS_INT_MAPLE_0_CH_11 (OS_INT_MAPLE_0_CH_10 + 1)
- #define OS_INT_MAPLE_0_CH_12 (OS_INT_MAPLE_0_CH_11 + 1)
- #define OS_INT_MAPLE_0_CH_13 (OS_INT_MAPLE_0_CH_12 + 1)
- #define OS_INT_MAPLE_0_CH_14 (OS_INT_MAPLE_0_CH_13 + 1)
- #define OS_INT_MAPLE_0_CH_15 (OS_INT_MAPLE_0_CH_14 + 1)
- #define OS_INT_MAPLE_0_CH_16 (OS_INT_MAPLE_0_CH_15 + 1)
- #define OS_INT_MAPLE_0_CH_17 (OS_INT_MAPLE_0_CH_16 + 1)
- #define OS_INT_MAPLE_0_CH_18 (OS_INT_MAPLE_0_CH_17 + 1)
- #define OS_INT_MAPLE_0_CH_19 (OS_INT_MAPLE_0_CH_18 + 1)
- #define OS_INT_MAPLE_0_CH_20 (OS_INT_MAPLE_0_CH_19 + 1)
- #define OS_INT_MAPLE_0_CH_21 (OS_INT_MAPLE_0_CH_20 + 1)
- #define OS_INT_MAPLE_0_CH_22 (OS_INT_MAPLE_0_CH_21 + 1)
- #define OS_INT_MAPLE_0_CH_23 (OS_INT_MAPLE_0_CH_22 + 1)
- #define OS_INT_MAPLE_0_CH_24 (OS_INT_MAPLE_0_CH_23 + 1)
- #define OS_INT_MAPLE_0_CH_25 (OS_INT_MAPLE_0_CH_24 + 1)
- #define OS_INT_MAPLE_0_CH_26 (OS_INT_MAPLE_0_CH_25 + 1)
- #define OS_INT_MAPLE_0_CH_27 (OS_INT_MAPLE_0_CH_26 + 1)
- #define OS_INT_MAPLE_0_CH_28 (OS_INT_MAPLE_0_CH_27 + 1)
- #define OS_INT_MAPLE_0_CH_29 (OS_INT_MAPLE_0_CH_28 + 1)
- #define OS_INT_MAPLE_0_CH_30 (OS_INT_MAPLE_0_CH_29 + 1)

- #define OS_INT_MAPLE_0_CH_31 (OS_INT_MAPLE_0_CH_30 + 1)
- #define OS_INT_MAPLE_1_CH_0 (EPIC_INTERRUPTS_OFFSET + 316)
- #define OS_INT_MAPLE_1_CH_1 (OS_INT_MAPLE_1_CH_0 + 1)
- #define OS_INT_MAPLE_1_CH_2 (OS_INT_MAPLE_1_CH_1 + 1)
- #define OS_INT_MAPLE_1_CH_3 (OS_INT_MAPLE_1_CH_2 + 1)
- #define OS_INT_MAPLE_1_CH_4 (OS_INT_MAPLE_1_CH_3 + 1)
- #define OS_INT_MAPLE_1_CH_5 (OS_INT_MAPLE_1_CH_4 + 1)
- #define OS_INT_MAPLE_1_CH_6 (OS_INT_MAPLE_1_CH_5 + 1)
- #define OS_INT_MAPLE_1_CH_7 (OS_INT_MAPLE_1_CH_6 + 1)
- #define OS_INT_MAPLE_1_CH_8 (OS_INT_MAPLE_1_CH_7 + 1)
- #define OS_INT_MAPLE_1_CH_9 (OS_INT_MAPLE_1_CH_8 + 1)
- #define OS_INT_MAPLE_1_CH_10 (OS_INT_MAPLE_1_CH_9 + 1)
- #define OS_INT_MAPLE_1_CH_11 (OS_INT_MAPLE_1_CH_10 + 1)
- #define OS_INT_MAPLE_1_CH_12 (OS_INT_MAPLE_1_CH_11 + 1)
- #define OS_INT_MAPLE_1_CH_13 (OS_INT_MAPLE_1_CH_12 + 1)
- #define OS_INT_MAPLE_1_CH_14 (OS_INT_MAPLE_1_CH_13 + 1)
- #define OS_INT_MAPLE_1_CH_15 (OS_INT_MAPLE_1_CH_14 + 1)
- #define OS_INT_MAPLE_1_CH_16 (OS_INT_MAPLE_1_CH_15 + 1)
- #define OS_INT_MAPLE_1_CH_17 (OS_INT_MAPLE_1_CH_16 + 1)
- #define OS_INT_MAPLE_1_CH_18 (OS_INT_MAPLE_1_CH_17 + 1)
- #define OS_INT_MAPLE_1_CH_19 (OS_INT_MAPLE_1_CH_18 + 1)
- #define OS_INT_MAPLE_1_CH_20 (OS_INT_MAPLE_1_CH_19 + 1)
- #define OS_INT_MAPLE_1_CH_21 (OS_INT_MAPLE_1_CH_20 + 1)
- #define OS_INT_MAPLE_1_CH_22 (OS_INT_MAPLE_1_CH_21 + 1)
- #define OS_INT_MAPLE_1_CH_23 (OS_INT_MAPLE_1_CH_22 + 1)
- #define OS_INT_MAPLE_1_CH_24 (OS_INT_MAPLE_1_CH_23 + 1)
- #define OS_INT_MAPLE_1_CH_25 (OS_INT_MAPLE_1_CH_24 + 1)
- #define OS_INT_MAPLE_1_CH_26 (OS_INT_MAPLE_1_CH_25 + 1)
- #define OS_INT_MAPLE_1_CH_27 (OS_INT_MAPLE_1_CH_26 + 1)
- #define OS_INT_MAPLE_1_CH_28 (OS_INT_MAPLE_1_CH_27 + 1)
- #define OS_INT_MAPLE_1_CH_29 (OS_INT_MAPLE_1_CH_28 + 1)
- #define OS_INT_MAPLE_1_CH_30 (OS_INT_MAPLE_1_CH_29 + 1)
- #define OS_INT_MAPLE_1_CH_31 (OS_INT_MAPLE_1_CH_30 + 1)
- #define OS_INT_MAPLE_2_CH_0 (EPIC_INTERRUPTS_OFFSET + 242)
- #define OS_INT_MAPLE_2_CH_1 (OS_INT_MAPLE_2_CH_0 + 1)
- #define OS_INT_MAPLE_2_CH_2 (OS_INT_MAPLE_2_CH_1 + 1)
- #define OS_INT_MAPLE_2_CH_3 (OS_INT_MAPLE_2_CH_2 + 1)
- #define OS_INT_MAPLE_2_CH_4 (OS_INT_MAPLE_2_CH_3 + 1)
- #define OS_INT_MAPLE_2_CH_5 (OS_INT_MAPLE_2_CH_4 + 1)
- #define OS_INT_MAPLE_2_CH_6 (OS_INT_MAPLE_2_CH_5 + 1)
- #define OS_INT_MAPLE_2_CH_7 (OS_INT_MAPLE_2_CH_6 + 1)
- #define OS_INT_MAPLE_2_CH_8 (OS_INT_MAPLE_2_CH_7 + 1)
- #define OS_INT_MAPLE_2_CH_9 (OS_INT_MAPLE_2_CH_8 + 1)
- #define OS_INT_MAPLE_2_CH_10 (OS_INT_MAPLE_2_CH_9 + 1)
- #define OS_INT_MAPLE_2_CH_11 (OS_INT_MAPLE_2_CH_10 + 1)
- #define OS_INT_MAPLE_2_CH_12 (OS_INT_MAPLE_2_CH_11 + 1)
- #define OS_INT_MAPLE_2_CH_13 (OS_INT_MAPLE_2_CH_12 + 1)
- #define OS_INT_MAPLE_2_CH_14 (OS_INT_MAPLE_2_CH_13 + 1)
- #define OS_INT_MAPLE_2_CH_15 (OS_INT_MAPLE_2_CH_14 + 1)
- #define OS_INT_MAPLE_2_CH_16 (OS_INT_MAPLE_2_CH_15 + 1)
- #define OS_INT_MAPLE_2_CH_17 (OS_INT_MAPLE_2_CH_16 + 1)
- #define OS_INT_MAPLE_2_CH_18 (OS_INT_MAPLE_2_CH_17 + 1)
- #define OS_INT_MAPLE_2_CH_19 (OS_INT_MAPLE_2_CH_18 + 1)
- #define OS_INT_MAPLE_2_CH_20 (OS_INT_MAPLE_2_CH_19 + 1)
- #define OS_INT_MAPLE_2_CH_21 (OS_INT_MAPLE_2_CH_20 + 1)
- #define OS_INT_MAPLE_2_CH_22 (OS_INT_MAPLE_2_CH_21 + 1)

- #define OS_INT_MAPLE_2_CH_23 (OS_INT_MAPLE_2_CH_22 + 1)
- #define OS_INT_MAPLE_2_CH_24 (OS_INT_MAPLE_2_CH_23 + 1)
- #define OS_INT_MAPLE_2_CH_25 (OS_INT_MAPLE_2_CH_24 + 1)
- #define OS_INT_MAPLE_2_CH_26 (OS_INT_MAPLE_2_CH_25 + 1)
- #define OS_INT_MAPLE_2_CH_27 (OS_INT_MAPLE_2_CH_26 + 1)
- #define OS_INT_MAPLE_2_CH_28 (OS_INT_MAPLE_2_CH_27 + 1)
- #define OS_INT_MAPLE_2_CH_29 (OS_INT_MAPLE_2_CH_28 + 1)
- #define OS_INT_MAPLE_2_CH_30 (OS_INT_MAPLE_2_CH_29 + 1)
- #define OS_INT_MAPLE_2_CH_31 (OS_INT_MAPLE_2_CH_30 + 1)

General interrupts - MapleB3

- #define OS_INT_MAPLE_2_SYS_ERR (EPIC_INTERRUPTS_OFFSET + 240)
- #define OS_INT_MAPLE_2_ECC_ERR (EPIC_INTERRUPTS_OFFSET + 241)
- #define OS_INT_MAPLE_0_SYS_ERR (EPIC_INTERRUPTS_OFFSET + 280)
- #define OS_INT_MAPLE_0_ECC_ERR (EPIC_INTERRUPTS_OFFSET + 281)
- #define OS_INT_MAPLE_1_SYS_ERR (EPIC_INTERRUPTS_OFFSET + 314)
- #define OS_INT_MAPLE_1_ECC_ERR (EPIC_INTERRUPTS_OFFSET + 315)

EPIC OCN DMA interrupts

- #define **OS_INT_OCN_DMA0_CH0** (EPIC_INTERRUPTS_OFFSET + 96)
- #define **OS_INT_OCN_DMA0_CH1** (EPIC_INTERRUPTS_OFFSET + 97)
- #define **OS_INT_OCN_DMA0_CH2** (EPIC_INTERRUPTS_OFFSET + 98)
- #define **OS_INT_OCN_DMA0_CH3** (EPIC_INTERRUPTS_OFFSET + 99)
- #define **OS_INT_OCN_DMA0_CH4** (EPIC_INTERRUPTS_OFFSET + 100)
- #define **OS_INT_OCN_DMA0_CH5** (EPIC_INTERRUPTS_OFFSET + 101)
- #define **OS_INT_OCN_DMA0_CH6** (EPIC_INTERRUPTS_OFFSET + 102)
- #define **OS_INT_OCN_DMA0_CH7** (EPIC_INTERRUPTS_OFFSET + 103)
- #define **OS_INT_OCN_DMA1_CH0** (EPIC_INTERRUPTS_OFFSET + 104)
- #define **OS_INT_OCN_DMA1_CH1** (EPIC_INTERRUPTS_OFFSET + 105)
- #define **OS_INT_OCN_DMA1_CH2** (EPIC_INTERRUPTS_OFFSET + 106)
- #define **OS_INT_OCN_DMA1_CH3** (EPIC_INTERRUPTS_OFFSET + 107)
- #define **OS_INT_OCN_DMA1_CH4** (EPIC_INTERRUPTS_OFFSET + 108)
- #define **OS_INT_OCN_DMA1_CH5** (EPIC_INTERRUPTS_OFFSET + 109)
- #define **OS_INT_OCN_DMA1_CH6** (EPIC_INTERRUPTS_OFFSET + 110)
- #define **OS_INT_OCN_DMA1_CH7** (EPIC_INTERRUPTS_OFFSET + 111)
- #define **OS_INT_PAMU_OCN_ERROR** (EPIC_INTERRUPTS_OFFSET + 480)

EPIC CPRI interrupts

- #define OS_INT_CPRI_0 (EPIC_INTERRUPTS_OFFSET + 146)
- #define OS_INT_CPRI_1 (EPIC_INTERRUPTS_OFFSET + 147)
- #define OS_INT_CPRI_2 (EPIC_INTERRUPTS_OFFSET + 148)
- #define OS_INT_CPRI_3 (EPIC_INTERRUPTS_OFFSET + 149)
- #define OS_INT_CPRI_4 (EPIC_INTERRUPTS_OFFSET + 150)
- #define OS_INT_CPRI_5 (EPIC_INTERRUPTS_OFFSET + 151)
- #define OS_INT_CPRI_6 (EPIC_INTERRUPTS_OFFSET + 152)
- #define OS_INT_CPRI_7 (EPIC_INTERRUPTS_OFFSET + 153)
- #define OS_INT_CPRI_8 (EPIC_INTERRUPTS_OFFSET + 154)
- #define OS_INT_CPRI_9 (EPIC_INTERRUPTS_OFFSET + 155)
- #define OS_INT_CPRI_10 (EPIC_INTERRUPTS_OFFSET + 156)
- #define OS_INT_CPRI_11 (EPIC_INTERRUPTS_OFFSET + 157)

- #define OS_INT_CPRI_12 (EPIC_INTERRUPTS_OFFSET + 158)
- #define OS_INT_CPRI_13 (EPIC_INTERRUPTS_OFFSET + 159)
- #define OS_INT_CPRI_14 (EPIC_INTERRUPTS_OFFSET + 160)
- #define OS_INT_CPRI_15 (EPIC_INTERRUPTS_OFFSET + 161)
- #define OS_INT_CPRI_0_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 130)
- #define OS_INT_CPRI_1_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 131)
- #define OS_INT_CPRI_2_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 132)
- #define OS_INT_CPRI_3_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 133)
- #define OS_INT_CPRI_4_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 134)
- #define OS_INT_CPRI_5_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 135)
- #define OS_INT_CPRI_6_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 136)
- #define OS_INT_CPRI_7_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 137)
- #define OS_INT_CPRI_0_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 138)
- #define OS_INT_CPRI_1_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 139)
- #define OS_INT_CPRI_2_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 140)
- #define OS_INT_CPRI_3_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 141)
- #define OS_INT_CPRI_4_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 142)
- #define OS_INT_CPRI_5_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 143)
- #define OS_INT_CPRI_6_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 144)
- #define OS_INT_CPRI_7_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 145)
- #define OS_INT_CPRI_0_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 114)
- #define OS_INT_CPRI_1_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 115)
- #define OS_INT_CPRI_2_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 116)
- #define OS_INT_CPRI_3_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 117)
- #define OS_INT_CPRI_4_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 118)
- #define OS_INT_CPRI_5_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 119)
- #define OS_INT_CPRI_6_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 120)
- #define OS_INT_CPRI_7_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 121)
- #define OS_INT_CPRI_0_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 122)
- #define OS_INT_CPRI_1_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 123)
- #define OS_INT_CPRI_2_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 124)
- #define OS_INT_CPRI_3_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 125)
- #define OS_INT_CPRI_4_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 126)
- #define OS_INT_CPRI_5_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 127)
- #define OS_INT_CPRI_6_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 128)
- #define OS_INT_CPRI_7_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 129)
- #define OS_INT_CPRI_GEN_ERR (EPIC_INTERRUPTS_OFFSET + 472)

EPIC EPU interrupts

- #define OS_INT_EPU1 (EPIC_INTERRUPTS_OFFSET + 434)
- #define OS_INT_EPU2 (EPIC_INTERRUPTS_OFFSET + 435)

EPIC SOC_TIMER interrupts

- #define OS_INT_SOC_TIMER_0_CH_0 (EPIC_INTERRUPTS_OFFSET + 176)
- #define OS_INT_SOC_TIMER_0_CH_1 (EPIC_INTERRUPTS_OFFSET + 177)
- #define OS_INT_SOC_TIMER_0_CH_2 (EPIC_INTERRUPTS_OFFSET + 178)
- #define OS_INT_SOC_TIMER_0_CH_3 (EPIC_INTERRUPTS_OFFSET + 179)
- #define OS_INT_SOC_TIMER_1_CH_0 (EPIC_INTERRUPTS_OFFSET + 180)
- #define OS_INT_SOC_TIMER_1_CH_1 (EPIC_INTERRUPTS_OFFSET + 181)
- #define OS_INT_SOC_TIMER_1_CH_2 (EPIC_INTERRUPTS_OFFSET + 182)
- #define OS_INT_SOC_TIMER_1_CH_3 (EPIC_INTERRUPTS_OFFSET + 183)
- #define OS_INT_SOC_TIMER_2_CH_0 (EPIC_INTERRUPTS_OFFSET + 184)

- #define OS_INT_SOC_TIMER_2_CH_1 (EPIC_INTERRUPTS_OFFSET + 185)
- #define OS_INT_SOC_TIMER_2_CH_2 (EPIC_INTERRUPTS_OFFSET + 186)
- #define OS_INT_SOC_TIMER_2_CH_3 (EPIC_INTERRUPTS_OFFSET + 187)
- #define OS_INT_SOC_TIMER_3_CH_0 (EPIC_INTERRUPTS_OFFSET + 188)
- #define OS_INT_SOC_TIMER_3_CH_1 (EPIC_INTERRUPTS_OFFSET + 189)
- #define OS_INT_SOC_TIMER_3_CH_2 (EPIC_INTERRUPTS_OFFSET + 190)
- #define OS_INT_SOC_TIMER_3_CH_3 (EPIC_INTERRUPTS_OFFSET + 191)
- #define OS_INT_SOC_TIMER_4_CH_0 (EPIC_INTERRUPTS_OFFSET + 192)
- #define OS_INT_SOC_TIMER_4_CH_1 (EPIC_INTERRUPTS_OFFSET + 193)
- #define OS_INT_SOC_TIMER_4_CH_2 (EPIC_INTERRUPTS_OFFSET + 194)
- #define OS_INT_SOC_TIMER_4_CH_3 (EPIC_INTERRUPTS_OFFSET + 195)
- #define OS_INT_SOC_TIMER_5_CH_0 (EPIC_INTERRUPTS_OFFSET + 196)
- #define OS_INT_SOC_TIMER_5_CH_1 (EPIC_INTERRUPTS_OFFSET + 197)
- #define OS_INT_SOC_TIMER_5_CH_2 (EPIC_INTERRUPTS_OFFSET + 198)
- #define OS_INT_SOC_TIMER_5_CH_3 (EPIC_INTERRUPTS_OFFSET + 199)
- #define OS_INT_SOC_TIMER_6_CH_0 (EPIC_INTERRUPTS_OFFSET + 200)
- #define OS_INT_SOC_TIMER_6_CH_1 (EPIC_INTERRUPTS_OFFSET + 201)
- #define OS_INT_SOC_TIMER_6_CH_2 (EPIC_INTERRUPTS_OFFSET + 202)
- #define OS_INT_SOC_TIMER_6_CH_3 (EPIC_INTERRUPTS_OFFSET + 203)
- #define OS_INT_SOC_TIMER_7_CH_0 (EPIC_INTERRUPTS_OFFSET + 204)
- #define OS_INT_SOC_TIMER_7_CH_1 (EPIC_INTERRUPTS_OFFSET + 205)
- #define OS_INT_SOC_TIMER_7_CH_2 (EPIC_INTERRUPTS_OFFSET + 206)
- #define OS_INT_SOC_TIMER_7_CH_3 (EPIC_INTERRUPTS_OFFSET + 207)
- #define OS_INT_SOC_TIMER_8_CH_0 (EPIC_INTERRUPTS_OFFSET + 208)
- #define OS_INT_SOC_TIMER_8_CH_1 (EPIC_INTERRUPTS_OFFSET + 209)
- #define OS_INT_SOC_TIMER_8_CH_2 (EPIC_INTERRUPTS_OFFSET + 210)
- #define OS_INT_SOC_TIMER_8_CH_3 (EPIC_INTERRUPTS_OFFSET + 211)
- #define OS_INT_SOC_TIMER_9_CH_0 (EPIC_INTERRUPTS_OFFSET + 212)
- #define OS_INT_SOC_TIMER_9_CH_1 (EPIC_INTERRUPTS_OFFSET + 213)
- #define OS_INT_SOC_TIMER_9_CH_2 (EPIC_INTERRUPTS_OFFSET + 214)
- #define OS_INT_SOC_TIMER_9_CH_3 (EPIC_INTERRUPTS_OFFSET + 215)
- #define OS_INT_SOC_TIMER_10_CH_0 (EPIC_INTERRUPTS_OFFSET + 216)
- #define OS_INT_SOC_TIMER_10_CH_1 (EPIC_INTERRUPTS_OFFSET + 217)
- #define OS_INT_SOC_TIMER_10_CH_2 (EPIC_INTERRUPTS_OFFSET + 218)
- #define OS_INT_SOC_TIMER_10_CH_3 (EPIC_INTERRUPTS_OFFSET + 219)
- #define OS_INT_SOC_TIMER_11_CH_0 (EPIC_INTERRUPTS_OFFSET + 220)
- #define OS_INT_SOC_TIMER_11_CH_1 (EPIC_INTERRUPTS_OFFSET + 221)
- #define OS_INT_SOC_TIMER_11_CH_2 (EPIC_INTERRUPTS_OFFSET + 222)
- #define OS_INT_SOC_TIMER_11_CH_3 (EPIC_INTERRUPTS_OFFSET + 223)
- #define OS_INT_SOC_TIMER_12_CH_0 (EPIC_INTERRUPTS_OFFSET + 224)
- #define OS_INT_SOC_TIMER_12_CH_1 (EPIC_INTERRUPTS_OFFSET + 225)
- #define OS_INT_SOC_TIMER_12_CH_2 (EPIC_INTERRUPTS_OFFSET + 226)
- #define OS_INT_SOC_TIMER_12_CH_3 (EPIC_INTERRUPTS_OFFSET + 227)
- #define OS_INT_SOC_TIMER_13_CH_0 (EPIC_INTERRUPTS_OFFSET + 228)
- #define OS_INT_SOC_TIMER_13_CH_1 (EPIC_INTERRUPTS_OFFSET + 229)
- #define OS_INT_SOC_TIMER_13_CH_2 (EPIC_INTERRUPTS_OFFSET + 230)
- #define OS_INT_SOC_TIMER_13_CH_3 (EPIC_INTERRUPTS_OFFSET + 231)
- #define OS_INT_SOC_TIMER_14_CH_0 (EPIC_INTERRUPTS_OFFSET + 232)
- #define OS_INT_SOC_TIMER_14_CH_1 (EPIC_INTERRUPTS_OFFSET + 233)
- #define OS_INT_SOC_TIMER_14_CH_2 (EPIC_INTERRUPTS_OFFSET + 234)
- #define OS_INT_SOC_TIMER_14_CH_3 (EPIC_INTERRUPTS_OFFSET + 235)
- #define OS_INT_SOC_TIMER_15_CH_0 (EPIC_INTERRUPTS_OFFSET + 236)
- #define OS_INT_SOC_TIMER_15_CH_1 (EPIC_INTERRUPTS_OFFSET + 237)
- #define OS_INT_SOC_TIMER_15_CH_2 (EPIC_INTERRUPTS_OFFSET + 238)
- #define OS_INT_SOC_TIMER_15_CH_3 (EPIC_INTERRUPTS_OFFSET + 239)

EPIC GCR_TIMER interrupts

- #define OS_INT_GCR_TIMER_16 (EPIC_INTERRUPTS_OFFSET + 352)
- #define OS_INT_GCR_TIMER_17 (EPIC_INTERRUPTS_OFFSET + 353)
- #define OS_INT_GCR_TIMER_18 (EPIC_INTERRUPTS_OFFSET + 354)
- #define OS_INT_GCR_TIMER_19 (EPIC_INTERRUPTS_OFFSET + 355)
- #define OS_INT_GCR_TIMER_20 (EPIC_INTERRUPTS_OFFSET + 356)
- #define OS_INT_GCR_TIMER_21 (EPIC_INTERRUPTS_OFFSET + 357)
- #define OS_INT_GCR_TIMER_22 (EPIC_INTERRUPTS_OFFSET + 358)
- #define OS_INT_GCR_TIMER_23 (EPIC_INTERRUPTS_OFFSET + 359)
- #define OS_INT_GCR_TIMER_24 (EPIC_INTERRUPTS_OFFSET + 360)
- #define OS_INT_GCR_TIMER_25 (EPIC_INTERRUPTS_OFFSET + 361)
- #define OS_INT_GCR_TIMER_26 (EPIC_INTERRUPTS_OFFSET + 362)
- #define OS_INT_GCR_TIMER_27 (EPIC_INTERRUPTS_OFFSET + 363)
- #define OS_INT_GCR_TIMER_28 (EPIC_INTERRUPTS_OFFSET + 364)
- #define OS_INT_GCR_TIMER_29 (EPIC_INTERRUPTS_OFFSET + 365)
- #define OS_INT_GCR_TIMER_30 (EPIC_INTERRUPTS_OFFSET + 366)
- #define OS_INT_GCR_TIMER_31 (EPIC_INTERRUPTS_OFFSET + 367)

QMAN/BMAN interrupts

- #define **OS_INT_QMAN_PORTAL_0** (EPIC_INTERRUPTS_OFFSET + 368)
- #define **OS_INT_BMAN_PORTAL_0** (EPIC_INTERRUPTS_OFFSET + 369)
- #define **OS_INT_QMAN_PORTAL_1** (EPIC_INTERRUPTS_OFFSET + 370)
- #define **OS_INT_BMAN_PORTAL_1** (EPIC_INTERRUPTS_OFFSET + 371)
- #define **OS_INT_QMAN_PORTAL_2** (EPIC_INTERRUPTS_OFFSET + 372)
- #define **OS_INT_BMAN_PORTAL_2** (EPIC_INTERRUPTS_OFFSET + 373)
- #define **OS_INT_QMAN_PORTAL_3** (EPIC_INTERRUPTS_OFFSET + 374)
- #define **OS_INT_BMAN_PORTAL_3** (EPIC_INTERRUPTS_OFFSET + 375)
- #define **OS_INT_QMAN_PORTAL_4** (EPIC_INTERRUPTS_OFFSET + 376)
- #define **OS_INT_BMAN_PORTAL_4** (EPIC_INTERRUPTS_OFFSET + 377)
- #define **OS_INT_QMAN_PORTAL_5** (EPIC_INTERRUPTS_OFFSET + 378)
- #define **OS_INT_BMAN_PORTAL_5** (EPIC_INTERRUPTS_OFFSET + 379)
- #define **OS_INT_QMAN_PORTAL_6** (EPIC_INTERRUPTS_OFFSET + 380)
- #define **OS_INT_BMAN_PORTAL_6** (EPIC_INTERRUPTS_OFFSET + 381)
- #define **OS_INT_QMAN_PORTAL_7** (EPIC_INTERRUPTS_OFFSET + 382)
- #define **OS_INT_BMAN_PORTAL_7** (EPIC_INTERRUPTS_OFFSET + 383)
- #define **OS_INT_QMAN_PORTAL_8** (EPIC_INTERRUPTS_OFFSET + 384)
- #define **OS_INT_BMAN_PORTAL_8** (EPIC_INTERRUPTS_OFFSET + 385)
- #define **OS_INT_QMAN_PORTAL_9** (EPIC_INTERRUPTS_OFFSET + 386)
- #define **OS_INT_BMAN_PORTAL_9** (EPIC_INTERRUPTS_OFFSET + 387)
- #define **OS_INT_QMAN_PORTAL_10** (EPIC_INTERRUPTS_OFFSET + 388)
- #define **OS_INT_BMAN_PORTAL_10** (EPIC_INTERRUPTS_OFFSET + 389)
- #define **OS_INT_QMAN_PORTAL_11** (EPIC_INTERRUPTS_OFFSET + 390)
- #define **OS_INT_BMAN_PORTAL_11** (EPIC_INTERRUPTS_OFFSET + 391)
- #define **OS_INT_QMAN_PORTAL_12** (EPIC_INTERRUPTS_OFFSET + 392)
- #define **OS_INT_BMAN_PORTAL_12** (EPIC_INTERRUPTS_OFFSET + 393)
- #define **OS_INT_QMAN_PORTAL_13** (EPIC_INTERRUPTS_OFFSET + 394)
- #define **OS_INT_BMAN_PORTAL_13** (EPIC_INTERRUPTS_OFFSET + 395)
- #define **OS_INT_QMAN_PORTAL_14** (EPIC_INTERRUPTS_OFFSET + 396)
- #define **OS_INT_BMAN_PORTAL_14** (EPIC_INTERRUPTS_OFFSET + 397)
- #define **OS_INT_QMAN_PORTAL_15** (EPIC_INTERRUPTS_OFFSET + 398)
- #define **OS_INT_BMAN_PORTAL_15** (EPIC_INTERRUPTS_OFFSET + 399)
- #define **OS_INT_QMAN_PORTAL_16** (EPIC_INTERRUPTS_OFFSET + 400)
- #define **OS_INT_BMAN_PORTAL_16** (EPIC_INTERRUPTS_OFFSET + 401)

- #define **OS_INT_QMAN_PORTAL_17** (EPIC_INTERRUPTS_OFFSET + 402)
- #define **OS_INT_BMAN_PORTAL_17** (EPIC_INTERRUPTS_OFFSET + 403)
- #define **OS_INT_QMAN_PORTAL_18** (EPIC_INTERRUPTS_OFFSET + 404)
- #define **OS_INT_BMAN_PORTAL_18** (EPIC_INTERRUPTS_OFFSET + 405)
- #define **OS_INT_QMAN_PORTAL_19** (EPIC_INTERRUPTS_OFFSET + 406)
- #define **OS_INT_BMAN_PORTAL_19** (EPIC_INTERRUPTS_OFFSET + 407)
- #define **OS_INT_QMAN_PORTAL_20** (EPIC_INTERRUPTS_OFFSET + 408)
- #define **OS_INT_BMAN_PORTAL_20** (EPIC_INTERRUPTS_OFFSET + 409)
- #define **OS_INT_QMAN_PORTAL_21** (EPIC_INTERRUPTS_OFFSET + 410)
- #define **OS_INT_BMAN_PORTAL_21** (EPIC_INTERRUPTS_OFFSET + 411)
- #define **OS_INT_QMAN_PORTAL_22** (EPIC_INTERRUPTS_OFFSET + 412)
- #define **OS_INT_BMAN_PORTAL_22** (EPIC_INTERRUPTS_OFFSET + 413)
- #define **OS_INT_QMAN_PORTAL_23** (EPIC_INTERRUPTS_OFFSET + 414)
- #define **OS_INT_BMAN_PORTAL_23** (EPIC_INTERRUPTS_OFFSET + 415)
- #define **OS_INT_QMAN_PORTAL_24** (EPIC_INTERRUPTS_OFFSET + 416)
- #define **OS_INT_BMAN_PORTAL_24** (EPIC_INTERRUPTS_OFFSET + 417)
- #define **OS_INT_BMAN_ERROR** (EPIC_INTERRUPTS_OFFSET + 475)
- #define **OS_INT_QMAN_ERROR** (EPIC_INTERRUPTS_OFFSET + 476)

Core Interrupts (not in EPIC)

- #define OS_INT_TRAP0 0
- #define OS_INT_TRAP1 1
- #define OS_INT_ILLEGAL 2
- #define OS_INT_IRRECOVERABLE 3
- #define OS_INT_IMMUAE 4
- #define OS_INT_DMMUAE 5
- #define OS_INT_DEBUG0 6
- #define OS_INT_DEBUG1 7
- #define OS_INT_DEBUG2 8
- #define OS_INT_DEBUG3 9
- #define OS_INT_SWBE 10
- #define OS_INT_RTEST 11
- #define OS_INT_HWDE 12
- #define OS_INT_PHWDE 13

Extended Core Interrupts

- #define OS_INT_CBE (EPIC_INTERRUPTS_OFFSET + 0x0)
- #define OS_INT_WE (EPIC_INTERRUPTS_OFFSET + 0x1)
- #define OS_INT_ICGE (EPIC_INTERRUPTS_OFFSET + 0x2)
- #define OS_INT_L2E (EPIC_INTERRUPTS_OFFSET + 0x4)
- #define OS_INT_AXI2ELINK0_ERR (EPIC_INTERRUPTS_OFFSET + 0x5)
- #define OS_INT_AXI2ELINK1_ERR (EPIC_INTERRUPTS_OFFSET + 0x6)
- #define OS_INT_CMS (EPIC_INTERRUPTS_OFFSET + 0x8)
- #define OS_INT_CGMS (EPIC_INTERRUPTS_OFFSET + 0x9)
- #define OS_INT_WDTI (EPIC_INTERRUPTS_OFFSET + 0xA)
- #define OS_INT_MS (EPIC_INTERRUPTS_OFFSET + 0x10)
- #define OS_INT_GMS (EPIC_INTERRUPTS_OFFSET + 0x11)
- #define OS_INT_MMS (EPIC_INTERRUPTS_OFFSET + 0x12)
- #define OS_INT_I_TM0 (EPIC_INTERRUPTS_OFFSET + 0x14)
- #define OS_INT_I_TM1 (EPIC_INTERRUPTS_OFFSET + 0x15)
- #define OS_INT_I_TM2 (EPIC_INTERRUPTS_OFFSET + 0x16)
- #define OS_INT_I_TM3 (EPIC_INTERRUPTS_OFFSET + 0x17)

- #define OS_INT_CBC (EPIC_INTERRUPTS_OFFSET + 0x18)
- #define OS_INT_AXI2ELINK0_MON (EPIC_INTERRUPTS_OFFSET + 0x1A)
- #define OS_INT_AXI2ELINK1_MON (EPIC_INTERRUPTS_OFFSET + 0x1B)
- #define OS_INT_IRQ_0 (EPIC_INTERRUPTS_OFFSET + 0x1F0)
- #define OS_INT_IRQ_1 (EPIC_INTERRUPTS_OFFSET + 0x1F1)
- #define OS_INT_IRQ_2 (EPIC_INTERRUPTS_OFFSET + 0x1F2)
- #define OS_INT_IRQ_3 (EPIC_INTERRUPTS_OFFSET + 0x1F3)
- #define OS_INT_IRQ_4 (EPIC_INTERRUPTS_OFFSET + 0x1F4)
- #define OS_INT_IRQ_5 (EPIC_INTERRUPTS_OFFSET + 0x1F5)
- #define OS_INT_IRQ_6 (EPIC_INTERRUPTS_OFFSET + 0x1F6)
- #define OS_INT_IRQ_7 (EPIC_INTERRUPTS_OFFSET + 0x1F7)
- #define OS_INT_IRQ_8 (EPIC_INTERRUPTS_OFFSET + 0x1F8)
- #define OS_INT_IRQ_9 (EPIC_INTERRUPTS_OFFSET + 0x1F9)
- #define OS_INT_IRQ_10 (EPIC_INTERRUPTS_OFFSET + 0x1FA)
- #define OS_INT_IRQ_11 (EPIC_INTERRUPTS_OFFSET + 0x1FB)
- #define OS_INT_MPIC_CI (EPIC_INTERRUPTS_OFFSET + 0x1FD)
- #define OS_INT_MPIC (EPIC_INTERRUPTS_OFFSET + 0x1FF)

2.13.12.2.2 Macro Definition Documentation

2.13.12.2.2.1 #define OS_INT_FROM_CORE0_0 (EPIC_INTERRUPTS_OFFSET + 32)

intercore interrupts from core 0 to core 1

2.13.12.2.2.2 #define OS_INT_FROM_CORE0_1 (EPIC_INTERRUPTS_OFFSET + 33)

intercore interrupts from core 0 to core 1

2.13.12.2.2.3 #define OS_INT_FROM_CORE1_0 (EPIC_INTERRUPTS_OFFSET + 34)

intercore interrupts from core 1 to core 0

2.13.12.2.2.4 #define OS_INT_FROM_CORE1_1 (EPIC_INTERRUPTS_OFFSET + 35)

intercore interrupts from core to core 1

2.13.12.2.2.5 #define OS_INT_FROM_CORE2_0 (EPIC_INTERRUPTS_OFFSET + 36)

intercore interrupts from core 2 to core 0

2.13.12.2.2.6 #define OS_INT_FROM_CORE2_1 (EPIC_INTERRUPTS_OFFSET + 37)

intercore interrupts from core 2 to core 1

2.13.12.2.2.7 #define OS_INT_FROM_CORE3_0 (EPIC_INTERRUPTS_OFFSET + 38)

intercore interrupts from core 3 to core 0

2.13.12.2.2.8 #define OS_INT_FROM_CORE3_1 (EPIC_INTERRUPTS_OFFSET + 39)

intercore interrupts from core 3 to core 1

2.13.12.2.2.9 #define OS_INT_FROM_CORE4_0 (EPIC_INTERRUPTS_OFFSET + 40)

intercore interrupts from core 4 to core 0

2.13.12.2.2.10 #define OS_INT_FROM_CORE4_1 (EPIC_INTERRUPTS_OFFSET + 41)

intercore interrupts from core 4 to core 1

2.13.12.2.2.11 #define OS_INT_FROM_CORE5_0 (EPIC_INTERRUPTS_OFFSET + 42)

intercore interrupts from core 5 to core 0

2.13.12.2.2.12 #define OS_INT_FROM_CORE5_1 (EPIC_INTERRUPTS_OFFSET + 43)

intercore interrupts from core 5 to core 1

2.13.12.2.2.13 #define OS_INT_VIRQ0 (EPIC_INTERRUPTS_OFFSET + 64)

virtaul interrupt number 0

2.13.12.2.2.14 #define OS_INT_VIRQ1 (EPIC_INTERRUPTS_OFFSET + 65)

virtaul interrupt number 1

2.13.12.2.2.15 #define OS_INT_VIRQ2 (EPIC_INTERRUPTS_OFFSET + 66)

virtaul interrupt number 2

2.13.12.2.2.16 #define OS_INT_VIRQ3 (EPIC_INTERRUPTS_OFFSET + 67)

virtaul interrupt number 3

2.13.12.2.2.17 #define OS_INT_VIRQ4 (EPIC_INTERRUPTS_OFFSET + 68)

virtaul interrupt number 4

2.13.12.2.2.18 #define OS_INT_VIRQ5 (EPIC_INTERRUPTS_OFFSET + 69)

virtaul interrupt number 5

2.13.12.2.2.19 #define OS_INT_VIRQ6 (EPIC_INTERRUPTS_OFFSET + 70)

virtaul interrupt number 6

2.13.12.2.2.20 #define OS_INT_VIRQ7 (EPIC_INTERRUPTS_OFFSET + 71)

virtaul interrupt number 7

2.13.12.2.2.21 #define OS_INT_VIRQ8 (EPIC_INTERRUPTS_OFFSET + 72)

virtaul interrupt number 8

2.13.12.2.2.22 #define OS_INT_VIRQ9 (EPIC_INTERRUPTS_OFFSET + 73)

virtaul interrupt number 9

2.13.12.2.2.23 #define OS_INT_VIRQ10 (EPIC_INTERRUPTS_OFFSET + 74)

virtaul interrupt number 10

2.13.12.2.2.24 #define OS_INT_VIRQ11 (EPIC_INTERRUPTS_OFFSET + 75)

virtaul interrupt number 11

2.13.12.2.2.25 #define OS_INT_VIRQ12 (EPIC_INTERRUPTS_OFFSET + 76)

virtaul interrupt number 12

2.13.12.2.2.26 #define OS_INT_VIRQ13 (EPIC_INTERRUPTS_OFFSET + 77)

virtaul interrupt number 13

2.13.12.2.2.27 #define OS_INT_VIRQ14 (EPIC_INTERRUPTS_OFFSET + 78)

virtaul interrupt number 14

2.13.12.2.2.28 #define OS_INT_VIRQ15 (EPIC_INTERRUPTS_OFFSET + 79)

virtaul interrupt number 15

2.13.12.2.2.29 #define OS_INT_VIRQ16 (EPIC_INTERRUPTS_OFFSET + 80)

virtaul interrupt number 16

2.13.12.2.2.30 #define OS_INT_VIRQ17 (EPIC_INTERRUPTS_OFFSET + 81)

virtaul interrupt number 17

2.13.12.2.2.31 #define OS_INT_VIRQ18 (EPIC_INTERRUPTS_OFFSET + 82)

virtaul interrupt number 18

2.13.12.2.2.32 #define OS_INT_VIRQ19 (EPIC_INTERRUPTS_OFFSET + 83)

virtaul interrupt number 19

2.13.12.2.2.33 #define OS_INT_VIRQ20 (EPIC_INTERRUPTS_OFFSET + 84)

virtaul interrupt number 20

2.13.12.2.2.34 #define OS_INT_VIRQ21 (EPIC_INTERRUPTS_OFFSET + 85)

virtaul interrupt number 21

2.13.12.2.2.35 #define OS_INT_VIRQ22 (EPIC_INTERRUPTS_OFFSET + 86)

virtaul interrupt number 22

2.13.12.2.2.36 #define OS_INT_VIRQ23 (EPIC_INTERRUPTS_OFFSET + 87)

virtaul interrupt number 23

2.13.12.2.2.37 #define OS_INT_VIRQ24 (EPIC_INTERRUPTS_OFFSET + 88)

virtaul interrupt number 24

2.13.12.2.2.38 #define OS_INT_VIRQ25 (EPIC_INTERRUPTS_OFFSET + 89)

virtaul interrupt number 25

2.13.12.2.2.39 #define OS_INT_VIRQ26 (EPIC_INTERRUPTS_OFFSET + 90)

virtaul interrupt number 26

2.13.12.2.2.40 #define OS_INT_VIRQ27 (EPIC_INTERRUPTS_OFFSET + 91)

virtaul interrupt number 27

2.13.12.2.2.41 #define OS_INT_VIRQ28 (EPIC_INTERRUPTS_OFFSET + 92)

virtaul interrupt number 28

2.13.12.2.2.42 #define OS_INT_VIRQ29 (EPIC_INTERRUPTS_OFFSET + 93)

virtaul interrupt number 29

2.13.12.2.2.43 #define OS_INT_VIRQ30 (EPIC_INTERRUPTS_OFFSET + 94)

virtaul interrupt number 30

2.13.12.2.2.44 #define OS_INT_VIRQ31 (EPIC_INTERRUPTS_OFFSET + 95)

virtaul interrupt number 31

2.13.12.2.2.45 #define OS_INT_MAPLE_0_CH_0 (EPIC_INTERRUPTS_OFFSET + 282)

MAPLE 0 Channel 0 Interrupt.

2.13.12.2.2.46 #define OS_INT_MAPLE_0_CH_1 (OS_INT_MAPLE_0_CH_0 + 1)

MAPLE 0 Channel 1 Interrupt.

2.13.12.2.2.47 #define OS_INT_MAPLE_0_CH_2 (OS_INT_MAPLE_0_CH_1 + 1)

MAPLE 0 Channel 2 Interrupt.

2.13.12.2.2.48 #define OS_INT_MAPLE_0_CH_3 (OS_INT_MAPLE_0_CH_2 + 1)

MAPLE 0 Channel 3 Interrupt.

2.13.12.2.2.49 #define OS_INT_MAPLE_0_CH_4 (OS_INT_MAPLE_0_CH_3 + 1)

MAPLE 0 Channel 4 Interrupt.

2.13.12.2.2.50 #define OS_INT_MAPLE_0_CH_5 (OS_INT_MAPLE_0_CH_4 + 1)

MAPLE 0 Channel 5 Interrupt.

2.13.12.2.2.51 #define OS_INT_MAPLE_0_CH_6 (OS_INT_MAPLE_0_CH_5 + 1)

MAPLE 0 Channel 6 Interrupt.

2.13.12.2.2.52 #define OS_INT_MAPLE_0_CH_7 (OS_INT_MAPLE_0_CH_6 + 1)

MAPLE 0 Channel 7 Interrupt.

2.13.12.2.2.53 #define OS_INT_MAPLE_0_CH_8 (OS_INT_MAPLE_0_CH_7 + 1)

MAPLE 0 Channel 8 Interrupt.

2.13.12.2.2.54 #define OS_INT_MAPLE_0_CH_9 (OS_INT_MAPLE_0_CH_8 + 1)

MAPLE 0 Channel 9 Interrupt.

2.13.12.2.2.55 #define OS_INT_MAPLE_0_CH_10 (OS_INT_MAPLE_0_CH_9 + 1)

MAPLE 0 Channel 10 Interrupt.

2.13.12.2.2.56 #define OS_INT_MAPLE_0_CH_11 (OS_INT_MAPLE_0_CH_10 + 1)

MAPLE 0 Channel 11 Interrupt.

2.13.12.2.2.57 #define OS_INT_MAPLE_0_CH_12 (OS_INT_MAPLE_0_CH_11 + 1)

MAPLE 0 Channel 12 Interrupt.

2.13.12.2.2.58 #define OS_INT_MAPLE_0_CH_13 (OS_INT_MAPLE_0_CH_12 + 1)

MAPLE 0 Channel 13 Interrupt.

2.13.12.2.2.59 #define OS_INT_MAPLE_0_CH_14 (OS_INT_MAPLE_0_CH_13 + 1)

MAPLE 0 Channel 14 Interrupt.

2.13.12.2.2.60 #define OS_INT_MAPLE_0_CH_15 (OS_INT_MAPLE_0_CH_14 + 1)

MAPLE 0 Channel 15 Interrupt.

2.13.12.2.2.61 #define OS_INT_MAPLE_0_CH_16 (OS_INT_MAPLE_0_CH_15 + 1)

MAPLE 0 Channel 16 Interrupt.

2.13.12.2.2.62 #define OS_INT_MAPLE_0_CH_17 (OS_INT_MAPLE_0_CH_16 + 1)

MAPLE 0 Channel 17 Interrupt.

2.13.12.2.2.63 #define OS_INT_MAPLE_0_CH_18 (OS_INT_MAPLE_0_CH_17 + 1)

MAPLE 0 Channel 18 Interrupt.

2.13.12.2.2.64 #define OS_INT_MAPLE_0_CH_19 (OS_INT_MAPLE_0_CH_18 + 1)

MAPLE 0 Channel 19 Interrupt.

2.13.12.2.2.65 #define OS_INT_MAPLE_0_CH_20 (OS_INT_MAPLE_0_CH_19 + 1)

MAPLE 0 Channel 20 Interrupt.

2.13.12.2.2.66 #define OS_INT_MAPLE_0_CH_21 (OS_INT_MAPLE_0_CH_20 + 1)

MAPLE 0 Channel 21 Interrupt.

2.13.12.2.2.67 #define OS_INT_MAPLE_0_CH_22 (OS_INT_MAPLE_0_CH_21 + 1)

MAPLE 0 Channel 22 Interrupt.

2.13.12.2.2.68 #define OS_INT_MAPLE_0_CH_23 (OS_INT_MAPLE_0_CH_22 + 1)

MAPLE 0 Channel 23 Interrupt.

2.13.12.2.2.69 #define OS_INT_MAPLE_0_CH_24 (OS_INT_MAPLE_0_CH_23 + 1)

MAPLE 0 Channel 24 Interrupt.

2.13.12.2.2.70 #define OS_INT_MAPLE_0_CH_25 (OS_INT_MAPLE_0_CH_24 + 1)

MAPLE 0 Channel 25 Interrupt.

2.13.12.2.2.71 #define OS_INT_MAPLE_0_CH_26 (OS_INT_MAPLE_0_CH_25 + 1)

MAPLE 0 Channel 26 Interrupt.

2.13.12.2.2.72 #define OS_INT_MAPLE_0_CH_27 (OS_INT_MAPLE_0_CH_26 + 1)

MAPLE 0 Channel 27 Interrupt.

2.13.12.2.2.73 #define OS_INT_MAPLE_0_CH_28 (OS_INT_MAPLE_0_CH_27 + 1)

MAPLE 0 Channel 28 Interrupt.

2.13.12.2.2.74 #define OS_INT_MAPLE_0_CH_29 (OS_INT_MAPLE_0_CH_28 + 1)

MAPLE 0 Channel 29 Interrupt.

2.13.12.2.2.75 #define OS_INT_MAPLE_0_CH_30 (OS_INT_MAPLE_0_CH_29 + 1)

MAPLE 0 Channel 30 Interrupt.

2.13.12.2.2.76 #define OS_INT_MAPLE_0_CH_31 (OS_INT_MAPLE_0_CH_30 + 1)

MAPLE 0 Channel 31 Interrupt.

2.13.12.2.2.77 #define OS_INT_MAPLE_1_CH_0 (EPIC_INTERRUPTS_OFFSET + 316)

MAPLE 1 Channel 0 Interrupt.

2.13.12.2.2.78 #define OS_INT_MAPLE_1_CH_1 (OS_INT_MAPLE_1_CH_0 + 1)

MAPLE 1 Channel 1 Interrupt.

2.13.12.2.2.79 #define OS_INT_MAPLE_1_CH_2 (OS_INT_MAPLE_1_CH_1 + 1)

MAPLE 1 Channel 2 Interrupt.

2.13.12.2.2.80 #define OS_INT_MAPLE_1_CH_3 (OS_INT_MAPLE_1_CH_2 + 1)

MAPLE 1 Channel 3 Interrupt.

2.13.12.2.2.81 #define OS_INT_MAPLE_1_CH_4 (OS_INT_MAPLE_1_CH_3 + 1)

MAPLE 1 Channel 4 Interrupt.

2.13.12.2.2.82 #define OS_INT_MAPLE_1_CH_5 (OS_INT_MAPLE_1_CH_4 + 1)

MAPLE 1 Channel 5 Interrupt.

2.13.12.2.2.83 #define OS_INT_MAPLE_1_CH_6 (OS_INT_MAPLE_1_CH_5 + 1)

MAPLE 1 Channel 6 Interrupt.

2.13.12.2.2.84 #define OS_INT_MAPLE_1_CH_7 (OS_INT_MAPLE_1_CH_6 + 1)

MAPLE 1 Channel 7 Interrupt.

2.13.12.2.2.85 #define OS_INT_MAPLE_1_CH_8 (OS_INT_MAPLE_1_CH_7 + 1)

MAPLE 1 Channel 8 Interrupt.

2.13.12.2.2.86 #define OS_INT_MAPLE_1_CH_9 (OS_INT_MAPLE_1_CH_8 + 1)

MAPLE 1 Channel 9 Interrupt.

2.13.12.2.2.87 #define OS_INT_MAPLE_1_CH_10 (OS_INT_MAPLE_1_CH_9 + 1)

MAPLE 1 Channel 10 Interrupt.

2.13.12.2.2.88 #define OS_INT_MAPLE_1_CH_11 (OS_INT_MAPLE_1_CH_10 + 1)

MAPLE 1 Channel 11 Interrupt.

2.13.12.2.2.89 #define OS_INT_MAPLE_1_CH_12 (OS_INT_MAPLE_1_CH_11 + 1)

MAPLE 1 Channel 12 Interrupt.

2.13.12.2.2.90 #define OS_INT_MAPLE_1_CH_13 (OS_INT_MAPLE_1_CH_12 + 1)

MAPLE 1 Channel 13 Interrupt.

2.13.12.2.2.91 #define OS_INT_MAPLE_1_CH_14 (OS_INT_MAPLE_1_CH_13 + 1)

MAPLE 1 Channel 14 Interrupt.

2.13.12.2.2.92 #define OS_INT_MAPLE_1_CH_15 (OS_INT_MAPLE_1_CH_14 + 1)

MAPLE 1 Channel 15 Interrupt.

2.13.12.2.2.93 #define OS_INT_MAPLE_1_CH_16 (OS_INT_MAPLE_1_CH_15 + 1)

MAPLE 1 Channel 16 Interrupt.

2.13.12.2.2.94 #define OS_INT_MAPLE_1_CH_17 (OS_INT_MAPLE_1_CH_16 + 1)

MAPLE 1 Channel 17 Interrupt.

2.13.12.2.2.95 #define OS_INT_MAPLE_1_CH_18 (OS_INT_MAPLE_1_CH_17 + 1)

MAPLE 1 Channel 18 Interrupt.

2.13.12.2.2.96 #define OS_INT_MAPLE_1_CH_19 (OS_INT_MAPLE_1_CH_18 + 1)

MAPLE 1 Channel 19 Interrupt.

2.13.12.2.2.97 #define OS_INT_MAPLE_1_CH_20 (OS_INT_MAPLE_1_CH_19 + 1)

MAPLE 1 Channel 20 Interrupt.

2.13.12.2.2.98 #define OS_INT_MAPLE_1_CH_21 (OS_INT_MAPLE_1_CH_20 + 1)

MAPLE 1 Channel 21 Interrupt.

2.13.12.2.2.99 #define OS_INT_MAPLE_1_CH_22 (OS_INT_MAPLE_1_CH_21 + 1)

MAPLE 1 Channel 22 Interrupt.

2.13.12.2.2.100 #define OS_INT_MAPLE_1_CH_23 (OS_INT_MAPLE_1_CH_22 + 1)

MAPLE 1 Channel 23 Interrupt.

2.13.12.2.2.101 #define OS_INT_MAPLE_1_CH_24 (OS_INT_MAPLE_1_CH_23 + 1)

MAPLE 1 Channel 24 Interrupt.

2.13.12.2.2.102 #define OS_INT_MAPLE_1_CH_25 (OS_INT_MAPLE_1_CH_24 + 1)

MAPLE 1 Channel 25 Interrupt.

2.13.12.2.2.103 #define OS_INT_MAPLE_1_CH_26 (OS_INT_MAPLE_1_CH_25 + 1)

MAPLE 1 Channel 26 Interrupt.

2.13.12.2.2.104 #define OS_INT_MAPLE_1_CH_27 (OS_INT_MAPLE_1_CH_26 + 1)

MAPLE 1 Channel 27 Interrupt.

2.13.12.2.2.105 #define OS_INT_MAPLE_1_CH_28 (OS_INT_MAPLE_1_CH_27 + 1)

MAPLE 1 Channel 28 Interrupt.

2.13.12.2.2.106 #define OS_INT_MAPLE_1_CH_29 (OS_INT_MAPLE_1_CH_28 + 1)

MAPLE 1 Channel 29 Interrupt.

2.13.12.2.2.107 #define OS_INT_MAPLE_1_CH_30 (OS_INT_MAPLE_1_CH_29 + 1)

MAPLE 1 Channel 30 Interrupt.

2.13.12.2.2.108 #define OS_INT_MAPLE_1_CH_31 (OS_INT_MAPLE_1_CH_30 + 1)

MAPLE 1 Channel 31 Interrupt.

2.13.12.2.2.109 #define OS_INT_MAPLE_2_CH_0 (EPIC_INTERRUPTS_OFFSET + 242)

MAPLE 2 Channel 0 Interrupt.

2.13.12.2.2.110 #define OS_INT_MAPLE_2_CH_1 (OS_INT_MAPLE_2_CH_0 + 1)

MAPLE 2 Channel 1 Interrupt.

2.13.12.2.2.111 #define OS_INT_MAPLE_2_CH_2 (OS_INT_MAPLE_2_CH_1 + 1)

MAPLE 2 Channel 2 Interrupt.

2.13.12.2.2.112 #define OS_INT_MAPLE_2_CH_3 (OS_INT_MAPLE_2_CH_2 + 1)

MAPLE 2 Channel 3 Interrupt.

2.13.12.2.2.113 #define OS_INT_MAPLE_2_CH_4 (OS_INT_MAPLE_2_CH_3 + 1)

MAPLE 2 Channel 4 Interrupt.

2.13.12.2.2.114 #define OS_INT_MAPLE_2_CH_5 (OS_INT_MAPLE_2_CH_4 + 1)

MAPLE 2 Channel 5 Interrupt.

2.13.12.2.2.115 #define OS_INT_MAPLE_2_CH_6 (OS_INT_MAPLE_2_CH_5 + 1)

MAPLE 2 Channel 6 Interrupt.

2.13.12.2.2.116 #define OS_INT_MAPLE_2_CH_7 (OS_INT_MAPLE_2_CH_6 + 1)

MAPLE 2 Channel 7 Interrupt.

2.13.12.2.2.117 #define OS_INT_MAPLE_2_CH_8 (OS_INT_MAPLE_2_CH_7 + 1)

MAPLE 2 Channel 8 Interrupt.

2.13.12.2.2.118 #define OS_INT_MAPLE_2_CH_9 (OS_INT_MAPLE_2_CH_8 + 1)

MAPLE 2 Channel 9 Interrupt.

2.13.12.2.2.119 #define OS_INT_MAPLE_2_CH_10 (OS_INT_MAPLE_2_CH_9 + 1)

MAPLE 2 Channel 10 Interrupt.

2.13.12.2.2.120 #define OS_INT_MAPLE_2_CH_11 (OS_INT_MAPLE_2_CH_10 + 1)

MAPLE 2 Channel 11 Interrupt.

2.13.12.2.2.121 #define OS_INT_MAPLE_2_CH_12 (OS_INT_MAPLE_2_CH_11 + 1)

MAPLE 2 Channel 12 Interrupt.

2.13.12.2.2.122 #define OS_INT_MAPLE_2_CH_13 (OS_INT_MAPLE_2_CH_12 + 1)

MAPLE 2 Channel 13 Interrupt.

2.13.12.2.2.123 #define OS_INT_MAPLE_2_CH_14 (OS_INT_MAPLE_2_CH_13 + 1)

MAPLE 2 Channel 14 Interrupt.

2.13.12.2.2.124 #define OS_INT_MAPLE_2_CH_15 (OS_INT_MAPLE_2_CH_14 + 1)

MAPLE 2 Channel 15 Interrupt.

2.13.12.2.2.125 #define OS_INT_MAPLE_2_CH_16 (OS_INT_MAPLE_2_CH_15 + 1)

MAPLE 2 Channel 16 Interrupt.

2.13.12.2.2.126 #define OS_INT_MAPLE_2_CH_17 (OS_INT_MAPLE_2_CH_16 + 1)

MAPLE 2 Channel 17 Interrupt.

2.13.12.2.2.127 #define OS_INT_MAPLE_2_CH_18 (OS_INT_MAPLE_2_CH_17 + 1)

MAPLE 2 Channel 18 Interrupt.

2.13.12.2.2.128 #define OS_INT_MAPLE_2_CH_19 (OS_INT_MAPLE_2_CH_18 + 1)

MAPLE 2 Channel 19 Interrupt.

2.13.12.2.2.129 #define OS_INT_MAPLE_2_CH_20 (OS_INT_MAPLE_2_CH_19 + 1)

MAPLE 2 Channel 20 Interrupt.

2.13.12.2.2.130 #define OS_INT_MAPLE_2_CH_21 (OS_INT_MAPLE_2_CH_20 + 1)

MAPLE 2 Channel 21 Interrupt.

2.13.12.2.2.131 #define OS_INT_MAPLE_2_CH_22 (OS_INT_MAPLE_2_CH_21 + 1)

MAPLE 2 Channel 22 Interrupt.

2.13.12.2.2.132 #define OS_INT_MAPLE_2_CH_23 (OS_INT_MAPLE_2_CH_22 + 1)

MAPLE 2 Channel 23 Interrupt.

2.13.12.2.2.133 #define OS_INT_MAPLE_2_CH_24 (OS_INT_MAPLE_2_CH_23 + 1)

MAPLE 2 Channel 24 Interrupt.

2.13.12.2.2.134 #define OS_INT_MAPLE_2_CH_25 (OS_INT_MAPLE_2_CH_24 + 1)

MAPLE 2 Channel 25 Interrupt.

2.13.12.2.2.135 #define OS_INT_MAPLE_2_CH_26 (OS_INT_MAPLE_2_CH_25 + 1)

MAPLE 2 Channel 26 Interrupt.

2.13.12.2.2.136 #define OS_INT_MAPLE_2_CH_27 (OS_INT_MAPLE_2_CH_26 + 1)

MAPLE 2 Channel 27 Interrupt.

2.13.12.2.2.137 #define OS_INT_MAPLE_2_CH_28 (OS_INT_MAPLE_2_CH_27 + 1)

MAPLE 2 Channel 28 Interrupt.

2.13.12.2.2.138 #define OS_INT_MAPLE_2_CH_29 (OS_INT_MAPLE_2_CH_28 + 1)

MAPLE 2 Channel 29 Interrupt.

2.13.12.2.2.139 #define OS_INT_MAPLE_2_CH_30 (OS_INT_MAPLE_2_CH_29 + 1)

MAPLE 2 Channel 30 Interrupt.

2.13.12.2.2.140 #define OS_INT_MAPLE_2_CH_31 (OS_INT_MAPLE_2_CH_30 + 1)

MAPLE 2 Channel 31 Interrupt.

2.13.12.2.2.141 #define OS_INT_MAPLE_2_SYS_ERR (EPIC_INTERRUPTS_OFFSET + 240)

MAPLE general error.

2.13.12.2.2.142 #define OS_INT_MAPLE_2_ECC_ERR (EPIC_INTERRUPTS_OFFSET + 241)

ECC error interrupt of the MAPLE.

2.13.12.2.2.143 #define OS_INT_MAPLE_0_SYS_ERR (EPIC_INTERRUPTS_OFFSET + 280)

MAPLE general error.

2.13.12.2.2.144 #define OS_INT_MAPLE_0_ECC_ERR (EPIC_INTERRUPTS_OFFSET + 281)

ECC error interrupt of the MAPLE.

2.13.12.2.2.145 #define OS_INT_MAPLE_1_SYS_ERR (EPIC_INTERRUPTS_OFFSET + 314)

MAPLE general error.

2.13.12.2.2.146 #define OS_INT_MAPLE_1_ECC_ERR (EPIC_INTERRUPTS_OFFSET + 315)

ECC error interrupt of the MAPLE.

2.13.12.2.2.147 #define OS_INT_CPRI_0 (EPIC_INTERRUPTS_OFFSET + 146)

CPRI interrupt 0.

2.13.12.2.2.148 #define OS_INT_CPRI_1 (EPIC_INTERRUPTS_OFFSET + 147)

CPRI interrupt 1.

2.13.12.2.2.149 #define OS_INT_CPRI_2 (EPIC_INTERRUPTS_OFFSET + 148)

CPRI interrupt 2.

2.13.12.2.2.150 #define OS_INT_CPRI_3 (EPIC_INTERRUPTS_OFFSET + 149)

CPRI interrupt 3.

2.13.12.2.2.151 #define OS_INT_CPRI_4 (EPIC_INTERRUPTS_OFFSET + 150)

CPRI interrupt 4.

2.13.12.2.2.152 #define OS_INT_CPRI_5 (EPIC_INTERRUPTS_OFFSET + 151)

CPRI interrupt 5.

2.13.12.2.2.153 #define OS_INT_CPRI_6 (EPIC_INTERRUPTS_OFFSET + 152)

CPRI interrupt 6.

2.13.12.2.2.154 #define OS_INT_CPRI_7 (EPIC_INTERRUPTS_OFFSET + 153)

CPRI interrupt 7.

2.13.12.2.2.155 #define OS_INT_CPRI_8 (EPIC_INTERRUPTS_OFFSET + 154)

CPRI interrupt 8.

2.13.12.2.2.156 #define OS_INT_CPRI_9 (EPIC_INTERRUPTS_OFFSET + 155)

CPRI interrupt 9.

2.13.12.2.2.157 #define OS_INT_CPRI_10 (EPIC_INTERRUPTS_OFFSET + 156)

CPRI interrupt 10.

2.13.12.2.2.158 #define OS_INT_CPRI_11 (EPIC_INTERRUPTS_OFFSET + 157)

CPRI interrupt 11.

2.13.12.2.2.159 #define OS_INT_CPRI_12 (EPIC_INTERRUPTS_OFFSET + 158)

CPRI interrupt 12.

2.13.12.2.2.160 #define OS_INT_CPRI_13 (EPIC_INTERRUPTS_OFFSET + 159)

CPRI interrupt 13.

2.13.12.2.2.161 #define OS_INT_CPRI_14 (EPIC_INTERRUPTS_OFFSET + 160)

CPRI interrupt 14.

2.13.12.2.2.162 #define OS_INT_CPRI_15 (EPIC_INTERRUPTS_OFFSET + 161)

CPRI interrupt 15.

2.13.12.2.2.163 #define OS_INT_CPRI_0_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 130)

CPRI receive control interrupt of CPRI unit 1.

2.13.12.2.2.164 #define OS_INT_CPRI_1_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 131)

CPRI receive control interrupt of CPRI unit 2.

2.13.12.2.2.165 #define OS_INT_CPRI_2_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 132)

CPRI receive control interrupt of CPRI unit 3.

2.13.12.2.2.166 #define OS_INT_CPRI_3_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 133)

CPRI receive control interrupt of CPRI unit 4.

2.13.12.2.2.167 #define OS_INT_CPRI_4_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 134)

CPRI receive control interrupt of CPRI unit 5.

2.13.12.2.2.168 #define OS_INT_CPRI_5_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 135)

CPRI receive control interrupt of CPRI unit 6.

2.13.12.2.2.169 #define OS_INT_CPRI_6_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 136)

CPRI receive control interrupt of CPRI unit 7.

2.13.12.2.2.170 #define OS_INT_CPRI_7_RX_CTRL (EPIC_INTERRUPTS_OFFSET + 137)

CPRI receive control interrupt of CPRI unit 8.

2.13.12.2.2.171 #define OS_INT_CPRI_0_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 138)

CPRI transmit control interrupt of CPRI unit 1.

2.13.12.2.2.172 #define OS_INT_CPRI_1_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 139)

CPRI transmit control interrupt of CPRI unit 2.

2.13.12.2.2.173 #define OS_INT_CPRI_2_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 140)

CPRI transmit control interrupt of CPRI unit 3.

2.13.12.2.2.174 #define OS_INT_CPRI_3_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 141)

CPRI transmit control interrupt of CPRI unit 4.

2.13.12.2.2.175 #define OS_INT_CPRI_4_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 142)

CPRI transmit control interrupt of CPRI unit 5.

2.13.12.2.2.176 #define OS_INT_CPRI_5_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 143)

CPRI transmit control interrupt of CPRI unit 6.

2.13.12.2.2.177 #define OS_INT_CPRI_6_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 144)

CPRI transmit control interrupt of CPRI unit 7.

2.13.12.2.2.178 #define OS_INT_CPRI_7_TX_CTRL (EPIC_INTERRUPTS_OFFSET + 145)

CPRI transmit control interrupt of CPRI unit 8.

2.13.12.2.2.179 #define OS_INT_CPRI_0_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 114)

CPRI receive control interrupt of CPRI unit 1.

2.13.12.2.2.180 #define OS_INT_CPRI_1_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 115)

CPRI receive control interrupt of CPRI unit 2.

2.13.12.2.2.181 #define OS_INT_CPRI_2_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 116)

CPRI receive control interrupt of CPRI unit 3.

2.13.12.2.2.182 #define OS_INT_CPRI_3_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 117)

CPRI receive control interrupt of CPRI unit 4.

2.13.12.2.2.183 #define OS_INT_CPRI_4_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 118)

CPRI receive control interrupt of CPRI unit 5.

2.13.12.2.2.184 #define OS_INT_CPRI_5_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 119)

CPRI receive control interrupt of CPRI unit 7.

2.13.12.2.2.185 #define OS_INT_CPRI_6_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 120)

CPRI receive control interrupt of CPRI unit 8.

2.13.12.2.2.186 #define OS_INT_CPRI_7_RX_TIMING (EPIC_INTERRUPTS_OFFSET + 121)

CPRI receive control interrupt of CPRI unit 9.

2.13.12.2.2.187 #define OS_INT_CPRI_0_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 122)

CPRI transmit control interrupt of CPRI unit 1.

2.13.12.2.2.188 #define OS_INT_CPRI_1_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 123)

CPRI transmit control interrupt of CPRI unit 2.

2.13.12.2.2.189 #define OS_INT_CPRI_2_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 124)

CPRI transmit control interrupt of CPRI unit 3.

2.13.12.2.2.190 #define OS_INT_CPRI_3_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 125)

CPRI transmit control interrupt of CPRI unit 4.

2.13.12.2.2.191 #define OS_INT_CPRI_4_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 126)

CPRI transmit control interrupt of CPRI unit 5.

2.13.12.2.2.192 #define OS_INT_CPRI_5_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 127)

CPRI transmit control interrupt of CPRI unit 6.

2.13.12.2.2.193 #define OS_INT_CPRI_6_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 128)

CPRI transmit control interrupt of CPRI unit 7.

2.13.12.2.2.194 #define OS_INT_CPRI_7_TX_TIMING (EPIC_INTERRUPTS_OFFSET + 129)

CPRI transmit control interrupt of CPRI unit 8.

2.13.12.2.2.195 #define OS_INT_CPRI_GEN_ERR (EPIC_INTERRUPTS_OFFSET + 472)

CPRI general error interrupt.

2.13.12.2.2.196 #define OS_INT_EPU1 (EPIC_INTERRUPTS_OFFSET + 434)

EPU1 interrupt number.

2.13.12.2.2.197 #define OS_INT_EPU2 (EPIC_INTERRUPTS_OFFSET + 435)

EPU2 interrupt number.

2.13.12.2.2.198 #define OS_INT_SOC_TIMER_0_CH_0 (EPIC_INTERRUPTS_OFFSET + 176)

TIMER module 0 channel 0.

2.13.12.2.2.199 #define OS_INT_SOC_TIMER_0_CH_1 (EPIC_INTERRUPTS_OFFSET + 177)

TIMER module 0 channel 1.

2.13.12.2.2.200 #define OS_INT_SOC_TIMER_0_CH_2 (EPIC_INTERRUPTS_OFFSET + 178)

TIMER module 0 channel 2.

2.13.12.2.2.201 #define OS_INT_SOC_TIMER_0_CH_3 (EPIC_INTERRUPTS_OFFSET + 179)

TIMER module 0 channel 3.

2.13.12.2.2.202 #define OS_INT_SOC_TIMER_1_CH_0 (EPIC_INTERRUPTS_OFFSET + 180)

TIMER module 1 channel 0.

2.13.12.2.2.203 #define OS_INT_SOC_TIMER_1_CH_1 (EPIC_INTERRUPTS_OFFSET + 181)

TIMER module 1 channel 1.

2.13.12.2.2.204 #define OS_INT_SOC_TIMER_1_CH_2 (EPIC_INTERRUPTS_OFFSET + 182)

TIMER module 1 channel 2.

2.13.12.2.2.205 #define OS_INT_SOC_TIMER_1_CH_3 (EPIC_INTERRUPTS_OFFSET + 183)

TIMER module 1 channel 3.

2.13.12.2.2.206 #define OS_INT_SOC_TIMER_2_CH_0 (EPIC_INTERRUPTS_OFFSET + 184)

TIMER module 2 channel 0.

2.13.12.2.2.207 #define OS_INT_SOC_TIMER_2_CH_1 (EPIC_INTERRUPTS_OFFSET + 185)

TIMER module 2 channel 1.

2.13.12.2.2.208 #define OS_INT_SOC_TIMER_2_CH_2 (EPIC_INTERRUPTS_OFFSET + 186)

TIMER module 2 channel 2.

2.13.12.2.2.209 #define OS_INT_SOC_TIMER_2_CH_3 (EPIC_INTERRUPTS_OFFSET + 187)

TIMER module 2 channel 3.

2.13.12.2.2.210 #define OS_INT_SOC_TIMER_3_CH_0 (EPIC_INTERRUPTS_OFFSET + 188)

TIMER module 3 channel 0.

2.13.12.2.2.211 #define OS_INT_SOC_TIMER_3_CH_1 (EPIC_INTERRUPTS_OFFSET + 189)

TIMER module 3 channel 1.

2.13.12.2.2.212 #define OS_INT_SOC_TIMER_3_CH_2 (EPIC_INTERRUPTS_OFFSET + 190)

TIMER module 3 channel 2.

2.13.12.2.2.213 #define OS_INT_SOC_TIMER_3_CH_3 (EPIC_INTERRUPTS_OFFSET + 191)

TIMER module 3 channel 3.

2.13.12.2.2.214 #define OS_INT_SOC_TIMER_4_CH_0 (EPIC_INTERRUPTS_OFFSET + 192)

TIMER module 4 channel 0.

2.13.12.2.2.215 #define OS_INT_SOC_TIMER_4_CH_1 (EPIC_INTERRUPTS_OFFSET + 193)

TIMER module 4 channel 1.

2.13.12.2.2.216 #define OS_INT_SOC_TIMER_4_CH_2 (EPIC_INTERRUPTS_OFFSET + 194)

TIMER module 4 channel 2.

2.13.12.2.2.217 #define OS_INT_SOC_TIMER_4_CH_3 (EPIC_INTERRUPTS_OFFSET + 195)

TIMER module 4 channel 3.

2.13.12.2.2.218 #define OS_INT_SOC_TIMER_5_CH_0 (EPIC_INTERRUPTS_OFFSET + 196)

TIMER module 5 channel 0.

2.13.12.2.2.219 #define OS_INT_SOC_TIMER_5_CH_1 (EPIC_INTERRUPTS_OFFSET + 197)

TIMER module 5 channel 1.

2.13.12.2.2.220 #define OS_INT_SOC_TIMER_5_CH_2 (EPIC_INTERRUPTS_OFFSET + 198)

TIMER module 5 channel 2.

2.13.12.2.2.221 #define OS_INT_SOC_TIMER_5_CH_3 (EPIC_INTERRUPTS_OFFSET + 199)

TIMER module 5 channel 3.

2.13.12.2.2.222 #define OS_INT_SOC_TIMER_6_CH_0 (EPIC_INTERRUPTS_OFFSET + 200)

TIMER module 6 channel 0.

2.13.12.2.2.223 #define OS_INT_SOC_TIMER_6_CH_1 (EPIC_INTERRUPTS_OFFSET + 201)

TIMER module 6 channel 1.

2.13.12.2.2.224 #define OS_INT_SOC_TIMER_6_CH_2 (EPIC_INTERRUPTS_OFFSET + 202)

TIMER module 6 channel 2.

2.13.12.2.2.225 #define OS_INT_SOC_TIMER_6_CH_3 (EPIC_INTERRUPTS_OFFSET + 203)

TIMER module 6 channel 3.

2.13.12.2.2.226 #define OS_INT_SOC_TIMER_7_CH_0 (EPIC_INTERRUPTS_OFFSET + 204)

TIMER module 7 channel 0.

2.13.12.2.2.227 #define OS_INT_SOC_TIMER_7_CH_1 (EPIC_INTERRUPTS_OFFSET + 205)

TIMER module 7 channel 1.

2.13.12.2.2.228 #define OS_INT_SOC_TIMER_7_CH_2 (EPIC_INTERRUPTS_OFFSET + 206)

TIMER module 7 channel 2.

2.13.12.2.2.229 #define OS_INT_SOC_TIMER_7_CH_3 (EPIC_INTERRUPTS_OFFSET + 207)

TIMER module 7 channel 3.

2.13.12.2.2.230 #define OS_INT_SOC_TIMER_8_CH_0 (EPIC_INTERRUPTS_OFFSET + 208)

TIMER module 8 channel 0.

2.13.12.2.2.231 #define OS_INT_SOC_TIMER_8_CH_1 (EPIC_INTERRUPTS_OFFSET + 209)

TIMER module 8 channel 1.

2.13.12.2.2.232 #define OS_INT_SOC_TIMER_8_CH_2 (EPIC_INTERRUPTS_OFFSET + 210)

TIMER module 8 channel 2.

2.13.12.2.2.233 #define OS_INT_SOC_TIMER_8_CH_3 (EPIC_INTERRUPTS_OFFSET + 211)

TIMER module 8 channel 3.

2.13.12.2.2.234 #define OS_INT_SOC_TIMER_9_CH_0 (EPIC_INTERRUPTS_OFFSET + 212)

TIMER module 9 channel 0.

2.13.12.2.2.235 #define OS_INT_SOC_TIMER_9_CH_1 (EPIC_INTERRUPTS_OFFSET + 213)

TIMER module 9 channel 1.

2.13.12.2.2.236 #define OS_INT_SOC_TIMER_9_CH_2 (EPIC_INTERRUPTS_OFFSET + 214)

TIMER module 9 channel 2.

2.13.12.2.2.237 #define OS_INT_SOC_TIMER_9_CH_3 (EPIC_INTERRUPTS_OFFSET + 215)

TIMER module 9 channel 3.

2.13.12.2.2.238 #define OS_INT_SOC_TIMER_10_CH_0 (EPIC_INTERRUPTS_OFFSET + 216)

TIMER module 10 channel 0.

2.13.12.2.2.239 #define OS_INT_SOC_TIMER_10_CH_1 (EPIC_INTERRUPTS_OFFSET + 217)

TIMER module 10 channel 1.

2.13.12.2.2.240 #define OS_INT_SOC_TIMER_10_CH_2 (EPIC_INTERRUPTS_OFFSET + 218)

TIMER module 10 channel 2.

2.13.12.2.2.241 #define OS_INT_SOC_TIMER_10_CH_3 (EPIC_INTERRUPTS_OFFSET + 219)

TIMER module 10 channel 3.

2.13.12.2.2.242 #define OS_INT_SOC_TIMER_11_CH_0 (EPIC_INTERRUPTS_OFFSET + 220)

TIMER module 11 channel 0.

2.13.12.2.2.243 #define OS_INT_SOC_TIMER_11_CH_1 (EPIC_INTERRUPTS_OFFSET + 221)

TIMER module 11 channel 1.

2.13.12.2.2.244 #define OS_INT_SOC_TIMER_11_CH_2 (EPIC_INTERRUPTS_OFFSET + 222)

TIMER module 11 channel 2.

2.13.12.2.2.245 #define OS_INT_SOC_TIMER_11_CH_3 (EPIC_INTERRUPTS_OFFSET + 223)

TIMER module 11 channel 3.

2.13.12.2.2.246 #define OS_INT_SOC_TIMER_12_CH_0 (EPIC_INTERRUPTS_OFFSET + 224)

TIMER module 12 channel 0.

2.13.12.2.2.247 #define OS_INT_SOC_TIMER_12_CH_1 (EPIC_INTERRUPTS_OFFSET + 225)

TIMER module 12 channel 1.

2.13.12.2.2.248 #define OS_INT_SOC_TIMER_12_CH_2 (EPIC_INTERRUPTS_OFFSET + 226)

TIMER module 12 channel 2.

2.13.12.2.2.249 #define OS_INT_SOC_TIMER_12_CH_3 (EPIC_INTERRUPTS_OFFSET + 227)

TIMER module 12 channel 3.

2.13.12.2.2.250 #define OS_INT_SOC_TIMER_13_CH_0 (EPIC_INTERRUPTS_OFFSET + 228)

TIMER module 13 channel 0.

2.13.12.2.2.251 #define OS_INT_SOC_TIMER_13_CH_1 (EPIC_INTERRUPTS_OFFSET + 229)

TIMER module 13 channel 1.

2.13.12.2.2.252 #define OS_INT_SOC_TIMER_13_CH_2 (EPIC_INTERRUPTS_OFFSET + 230)

TIMER module 13 channel 2.

2.13.12.2.2.253 #define OS_INT_SOC_TIMER_13_CH_3 (EPIC_INTERRUPTS_OFFSET + 231)

TIMER module 13 channel 3.

2.13.12.2.2.254 #define OS_INT_SOC_TIMER_14_CH_0 (EPIC_INTERRUPTS_OFFSET + 232)

TIMER module 14 channel 0.

2.13.12.2.2.255 #define OS_INT_SOC_TIMER_14_CH_1 (EPIC_INTERRUPTS_OFFSET + 233)

TIMER module 14 channel 1.

2.13.12.2.2.256 #define OS_INT_SOC_TIMER_14_CH_2 (EPIC_INTERRUPTS_OFFSET + 234)

TIMER module 14 channel 2.

2.13.12.2.2.257 #define OS_INT_SOC_TIMER_14_CH_3 (EPIC_INTERRUPTS_OFFSET + 235)

TIMER module 14 channel 3.

2.13.12.2.2.258 #define OS_INT_SOC_TIMER_15_CH_0 (EPIC_INTERRUPTS_OFFSET + 236)

TIMER module 15 channel 0.

2.13.12.2.2.259 #define OS_INT_SOC_TIMER_15_CH_1 (EPIC_INTERRUPTS_OFFSET + 237)

TIMER module 15 channel 1.

2.13.12.2.2.260 #define OS_INT_SOC_TIMER_15_CH_2 (EPIC_INTERRUPTS_OFFSET + 238)

TIMER module 15 channel 2.

2.13.12.2.2.261 #define OS_INT_SOC_TIMER_15_CH_3 (EPIC_INTERRUPTS_OFFSET + 239)

TIMER module 15 channel 3.

2.13.12.2.2.262 #define OS_INT_GCR_TIMER_16 (EPIC_INTERRUPTS_OFFSET + 352)

TIMER module 16 channels 0-3.

2.13.12.2.2.263 #define OS_INT_GCR_TIMER_17 (EPIC_INTERRUPTS_OFFSET + 353)

TIMER module 17 channels 0-3.

2.13.12.2.2.264 #define OS_INT_GCR_TIMER_18 (EPIC_INTERRUPTS_OFFSET + 354)

TIMER module 18 channels 0-3.

2.13.12.2.2.265 #define OS_INT_GCR_TIMER_19 (EPIC_INTERRUPTS_OFFSET + 355)

TIMER module 19 channels 0-3.

2.13.12.2.2.266 #define OS_INT_GCR_TIMER_20 (EPIC_INTERRUPTS_OFFSET + 356)

TIMER module 20 channels 0-3.

2.13.12.2.2.267 #define OS_INT_GCR_TIMER_21 (EPIC_INTERRUPTS_OFFSET + 357)

TIMER module 21 channels 0-3.

2.13.12.2.2.268 #define OS_INT_GCR_TIMER_22 (EPIC_INTERRUPTS_OFFSET + 358)

TIMER module 22 channels 0-3.

2.13.12.2.2.269 #define OS_INT_GCR_TIMER_23 (EPIC_INTERRUPTS_OFFSET + 359)

TIMER module 23 channels 0-3.

2.13.12.2.2.270 #define OS_INT_GCR_TIMER_24 (EPIC_INTERRUPTS_OFFSET + 360)

TIMER module 24 channels 0-3.

2.13.12.2.2.271 #define OS_INT_GCR_TIMER_25 (EPIC_INTERRUPTS_OFFSET + 361)

TIMER module 25 channels 0-3.

2.13.12.2.2.272 #define OS_INT_GCR_TIMER_26 (EPIC_INTERRUPTS_OFFSET + 362)

TIMER module 26 channels 0-3.

2.13.12.2.2.273 #define OS_INT_GCR_TIMER_27 (EPIC_INTERRUPTS_OFFSET + 363)

TIMER module 27 channels 0-3.

2.13.12.2.2.274 #define OS_INT_GCR_TIMER_28 (EPIC_INTERRUPTS_OFFSET + 364)

TIMER module 28 channels 0-3.

2.13.12.2.2.275 #define OS_INT_GCR_TIMER_29 (EPIC_INTERRUPTS_OFFSET + 365)

TIMER module 29 channels 0-3.

2.13.12.2.2.276 #define OS_INT_GCR_TIMER_30 (EPIC_INTERRUPTS_OFFSET + 366)

TIMER module 30 channels 0-3.

2.13.12.2.2.277 #define OS_INT_GCR_TIMER_31 (EPIC_INTERRUPTS_OFFSET + 367)

TIMER module 31 channels 0-3.

2.13.12.2.2.278 #define OS_INT_TRAP0 0

Internal exception (generated by a trap.0 instruction).

2.13.12.2.2.279 #define OS_INT_TRAP1 1

Internal exception (generated by a trap.1 instruction).

2.13.12.2.2.280 #define OS_INT_ILLEGAL 2

Illegal event - encoded in EIDR.EID.

2.13.12.2.2.281 #define OS_INT_IRRECOVERABLE 3

Irrecoverable event.

2.13.12.2.2.282 #define OS_INT_IMMUAE 4

Instruction MMU Error - encoded in EIDR.EID.

2.13.12.2.2.283 #define OS_INT_DMMUAE 5

Data MMU Error - encoded in EIDR.EID.

2.13.12.2.2.284 #define OS_INT_DEBUG0 6

Debug exception (generated by debug.0 instruction).

2.13.12.2.2.285 #define OS_INT_DEBUG1 7

Debug exception (generated by debug.1 instruction).

2.13.12.2.2.286 #define OS_INT_DEBUG2 8

Debug exception (generated by debug.2 instruction).

2.13.12.2.2.287 #define OS_INT_DEBUG3 9

Debug exception (generated by debug.3 instruction).

2.13.12.2.2.288 #define OS_INT_SWBE 10

Software Breakpoint Event.

2.13.12.2.2.289 #define OS_INT_RTEST 11

Return after single step.

2.13.12.2.2.290 #define OS_INT_HWDE 12

Imprecise debugger HW events.

2.13.12.2.2.291 #define OS_INT_PHWDE 13

Precise debugger HW events.

2.13.12.2.2.292 #define EPIC_INTERRUPTS_OFFSET 14

The EPIC interrupts offset.

2.13.12.2.2.293 #define OS_INT_CBE (EPIC_INTERRUPTS_OFFSET + 0x0)

CME block error.

2.13.12.2.2.294 #define OS_INT_WE (EPIC_INTERRUPTS_OFFSET + 0x1)

Write error on SB or non cacheable write hit.

2.13.12.2.2.295 #define OS_INT_ICGE (EPIC_INTERRUPTS_OFFSET + 0x2)

Instruction granular cache command error.

2.13.12.2.2.296 #define OS_INT_L2E (EPIC_INTERRUPTS_OFFSET + 0x4)

L2 cache error.

2.13.12.2.2.297 #define OS_INT_AXI2ELINK0_ERR (EPIC_INTERRUPTS_OFFSET + 0x5)

AXI2ELINK0 error.

2.13.12.2.2.298 #define OS_INT_AXI2ELINK1_ERR (EPIC_INTERRUPTS_OFFSET + 0x6)

AXI2ELINK1 error.

2.13.12.2.2.299 #define OS_INT_CMS (EPIC_INTERRUPTS_OFFSET + 0x8)

Critical message.

2.13.12.2.2.300 #define OS_INT_CGMS (EPIC_INTERRUPTS_OFFSET + 0x9)

Critical guest message.

2.13.12.2.2.301 #define OS_INT_WDTI (EPIC_INTERRUPTS_OFFSET + 0xA)

WDT interrupt.

2.13.12.2.2.302 #define OS_INT_MS (EPIC_INTERRUPTS_OFFSET + 0x10)

Message.

2.13.12.2.2.303 #define OS_INT_GMS (EPIC_INTERRUPTS_OFFSET + 0x11)

Guest message.

2.13.12.2.2.304 #define OS_INT_MMS (EPIC_INTERRUPTS_OFFSET + 0x12)

Machine check message.

2.13.12.2.2.305 #define OS_INT_I_TM0 (EPIC_INTERRUPTS_OFFSET + 0x14)

Timer 0 interrupt.

2.13.12.2.2.306 #define OS_INT_I_TM1 (EPIC_INTERRUPTS_OFFSET + 0x15)

Timer 1 interrupt.

2.13.12.2.2.307 #define OS_INT_I_TM2 (EPIC_INTERRUPTS_OFFSET + 0x16)

Timer 2 interrupt.

2.13.12.2.2.308 #define OS_INT_I_TM3 (EPIC_INTERRUPTS_OFFSET + 0x17)

Timer 3 interrupt.

2.13.12.2.2.309 #define OS_INT_CBC (EPIC_INTERRUPTS_OFFSET + 0x18)

CME block completion.

2.13.12.2.2.310 #define OS_INT_AXI2ELINK0_MON (EPIC_INTERRUPTS_OFFSET + 0x1A)

AXI2ELINK0 monitor.

2.13.12.2.2.311 #define OS_INT_AXI2ELINK1_MON (EPIC_INTERRUPTS_OFFSET + 0x1B)

AXI2ELINK1 monitor.

2.13.12.2.2.312 #define OS_INT_IRQ_0 (EPIC_INTERRUPTS_OFFSET + 0x1F0)

IRQ pin 0 interrupt port.

2.13.12.2.2.313 #define OS_INT_IRQ_1 (EPIC_INTERRUPTS_OFFSET + 0x1F1)

IRQ pin 1 interrupt port.

2.13.12.2.2.314 #define OS_INT_IRQ_2 (EPIC_INTERRUPTS_OFFSET + 0x1F2)

IRQ pin 2 interrupt port.

2.13.12.2.2.315 #define OS_INT_IRQ_3 (EPIC_INTERRUPTS_OFFSET + 0x1F3)

IRQ pin 3 interrupt port.

2.13.12.2.2.316 #define OS_INT_IRQ_4 (EPIC_INTERRUPTS_OFFSET + 0x1F4)

IRQ pin 4 interrupt port.

2.13.12.2.2.317 #define OS_INT_IRQ_5 (EPIC_INTERRUPTS_OFFSET + 0x1F5)

IRQ pin 5 interrupt port.

2.13.12.2.2.318 #define OS_INT_IRQ_6 (EPIC_INTERRUPTS_OFFSET + 0x1F6)

IRQ pin 6 interrupt port.

2.13.12.2.2.319 #define OS_INT_IRQ_7 (EPIC_INTERRUPTS_OFFSET + 0x1F7)

IRQ pin 7 interrupt port.

2.13.12.2.2.320 #define OS_INT_IRQ_8 (EPIC_INTERRUPTS_OFFSET + 0x1F8)

IRQ pin 8 interrupt port.

2.13.12.2.2.321 #define OS_INT_IRQ_9 (EPIC_INTERRUPTS_OFFSET + 0x1F9)

IRQ pin 9 interrupt port.

2.13.12.2.2.322 #define OS_INT_IRQ_10 (EPIC_INTERRUPTS_OFFSET + 0x1FA)

IRQ pin 10 interrupt port.

2.13.12.2.2.323 #define OS_INT_IRQ_11 (EPIC_INTERRUPTS_OFFSET + 0x1FB)

IRQ pin 11 interrupt port.

2.13.12.2.2.324 #define OS_INT_MPIC_CI (EPIC_INTERRUPTS_OFFSET + 0x1FD)

mpic critical interrupt port

2.13.12.2.2.325 #define OS_INT_MPIC (EPIC_INTERRUPTS_OFFSET + 0x1FF)

mpic interrupt port

2.13.12.2.2.326 #define EPIC_EDGE_INTERRUPT_START (EPIC_INTERRUPTS_OFFSET + 48)

the first epic edge interrupt id

2.13.12.2.2.327 #define EPIC_EDGE_INTERRUPT_END (EPIC_INTERRUPTS_OFFSET + 95)

the LAST epic edge interrupt id

2.13.12.2.2.328 #define EPIC_INTERNAL_INTERRUPT_END (EPIC_INTERRUPTS_OFFSET + 0x1F)

Last internal interrupt - reserved.

2.13.12.2.2.329 #define OS_HWI_DEFAULT_DISPATCHER osHwiFullScheduler

dispatcher to call when no other was defined

2.13.12.2.2.330 #define OS_SWI_DEFALT_DISPATCHER osSwiTaskDispatcher

dispatcher to call when SWI is invoked

2.13.12.2.2.331 #define OS_TASK_DEFALT_DISPATCHER osHwiTaskScheduler

dispatcher to call when a task switch is requested

2.13.13 B4860 L1 Defense

2.13.13.1 Overview

L1 Defense initialization.

Data Structures

- struct maple_allocated_buffers_t
- struct maple_data_t
- struct l1d_data_t
- struct l1d_init_params_t
- struct l1d_clean_params_t
- struct maple3_clean_param_t

Typedefs

- typedef long(∗ maple3_config_func_t)(void ∗addr, void ∗data, unsigned long int num_bytes, unsigned long int dev_id, unsigned long int config_param)

Functions

- os_status osL1dResetFlowSet (l1d_init_params_t ∗soc_l1d_init_params, os_het_l1_defense_t ∗os↩ _het_l1d, l1d_clean_params_t ∗clean_params)
- os_status osL1dInitialize (l1d_init_params_t ∗soc_l1d_init_params)
- void osL1dResetRequest (os_hwi_arg arg)
- os_status osL1dHardwareClean (l1d_clean_params_t ∗clean_params)
- os_het_l1d_reset_maple_t osL1dGetMapleResetStatus ()

2.13.13.2 Data Structure Documentation

2.13.13.2.1 struct maple_allocated_buffers_t

maple buffer to be freed in case of L1 defense warm reset

Data Fields

- uint8_t core_id
- void ∗ buffer

2.13.13.2.1.1 Field Documentation

2.13.13.2.1.1.1 uint8_t maple_allocated_buffers_t::core_id

ID of allocating core (owner)

2.13.13.2.1.1.2 void* maple_allocated_buffers_t::buffer

pointer to allocated buffer

2.13.13.2.2 struct maple_data_t

Saved MAPLE data for L1 defense use.

2.13.13.2.3 struct l1d_data_t

data structure of L1 defense.

Data Fields

- uint32_t start_validation_value
- maple_data_t maple_data
- uint32_t end_validation_value

2.13.13.2.3.1 Field Documentation

2.13.13.2.3.1.1 uint32_t l1d_data_t::start_validation_value

validation value for checking for corruption in case of reset

2.13.13.2.3.1.2 maple_data_t l1d_data_t::maple_data

L1 defense maple data.

2.13.13.2.3.1.3 uint32_t l1d_data_t::end_validation_value

validation value for checking for corruption in case of reset

2.13.13.2.4 struct l1d_init_params_t

initialization structure of L1 defense.

This structure is passed once for a device init and include parameters for initialization of the L1 defense on the device.

Data Fields

- uint32_t enabled_modes
- os_hwi_handle hwi_num [OS_SOC_MAX_NUM_OF_CORES]

2.13.13.2.4.1 Field Documentation

2.13.13.2.4.1.1 uint32_t l1d_init_params_t::enabled_modes

enabled L1 defense modes

2.13.13.2.4.1.2 os_hwi_handle l1d_init_params_t::hwi_num[OS_SOC_MAX_NUM_OF_CORES]

virtual interrupt IDs to be used for invoking of L1 defense

2.13.13.2.5 struct l1d_clean_params_t

Structure for hardware clean parameters of L1 defense.

This structure is used by osL1dHardwareClean() function. Application which supports the L1 defense should fill this structure before call of osL1dHardwareClean().

Data Fields

- bool l2_cache
- bool maple [NUM_OF_MAPLES]
- bool cpri [NUM_OF_MEMMAP_CPRI_UNITS]
- bool mpic_dsp
- bool debug_print
- void * ocn_dma [NUMBER_OF_OCN_DMA]

2.13.13.2.5.1 Field Documentation

2.13.13.2.5.1.1 bool l1d_clean_params_t::l2_cache

status of L2 cache: enable = TRUE, disable = FALSE

2.13.13.2.5.1.2 bool l1d_clean_params_t::maple[NUM_OF_MAPLES]

status of MAPLE units: enable = TRUE, disable = FALSE

2.13.13.2.5.1.3 bool l1d_clean_params_t::cpri[NUM_OF_MEMMAP_CPRI_UNITS]

status of CPRI units: enable = TRUE, disable = FALSE

2.13.13.2.5.1.4 bool l1d_clean_params_t::mpic_dsp

status of MPIC: enable = TRUE, disable = FALSE

2.13.13.2.5.1.5 bool l1d_clean_params_t::debug_print

NOT IN USE - place holder for future compatibility.

value has no impact

2.13.13.2.5.1.6 void* l1d_clean_params_t::ocn_dma[NUMBER_OF_OCN_DMA]

pointers to OCN DMA unit

2.13.13.2.6 struct maple3_clean_param_t

Structure for hardware clean MAPLE parameters of L1 defense.

This structure is used by osL1dHardwareClean() function.

2.13.13.3 Typedef Documentation

2.13.13.3.1 typedef long(* maple3_config_func_t)(void *addr, void *data, unsigned long int num_bytes, unsigned long int dev_id, unsigned long int config_param)

Pointer to config_read/config_write functions for clean MAPLE parameters.

2.13.13.4 Function Documentation

2.13.13.4.1 os_status osL1dResetFlowSet (l1d_init_params_t * *soc_l1d_init_params*, os_het_l1_defense_t * *os_het_l1d*, l1d_clean_params_t * *clean_params*)

Check if L1 defense reset has occured. If true, enable proper reset flow

Parameters

in	*soc_l1d_init_params*	- Initialization parameters for L1 defense
in	*os_het_l1d*	- Pointer to l1 defense heterogeneous structure
in	*clean_params*	- Hardware clean parameters for L1 defense

Returns

OS status.

2.13.13.4.2 os_status osL1dInitialize (l1d_init_params_t * *soc_l1d_init_params*)

L1 Defense initialization method

Parameters

in	*soc_l1d_init_params*	- Initialization parameters for L1 defense

Returns

OS status.

2.13.13.4.3 void osL1dResetRequest (os_hwi_arg *arg*)

Prepare core for a warm reset.

2.13.13.4.4 os_status osL1dHardwareClean (l1d_clean_params_t * *clean_params*)

Clean hardware registers and return them to original state after reset.

Parameters

in	*clean_params*	- hardware clean parameters.

Returns

OS status

2.13.13.4.5 os_het_l1d_reset_maple_t osL1dGetMapleResetStatus ( )

Returns a bitmask of which MAPLEs performed soft reset (RESET_MAPLE_0, RESET_MAPLE_1 & RESET_MAPLE_2)

Returns

Which MAPLE were reset

2.14 MPIC Module API

2.14.1 Overview

MPIC Module API.

Modules

- MPIC B486x Initialization API
- MPIC runtime API

2.14.2 MPIC B486x Initialization API

2.14.2.1 Overview

B486x Initialization API.

Macros

- #define NUM_OF_MPIC_IPI_CORES (OS_SOC_MAX_NUM_OF_CORES + OS_SOC_MAX↩ _NUM_OF_PA_THREADS)
- #define MPIC_NUM_OF_EXT_INTRS 12
- #define MPIC_NUM_OF_INT_INTRS 256
- #define MPIC_NUM_OF_TIMERS 8
- #define MPIC_NUM_OF_MSG_INTRS 16
- #define MPIC_NUM_OF_SMSG_INTRS 64
- #define MPIC_NUM_OF_IPI_EVENTS 4
- #define MPIC_NUM_OF_LOW_INT_INTRS 160
- #define MPIC_MAX_NUM_OF_INTR_SRC
- #define MPIC_MULTI_SRC_INTERNAL_INTERRUPTS
- #define MPIC_MULTI_SRC_INTERNAL_INTERRUPTS_COUNT 40
- #define MPIC_INTR(type, idx, mult_src_idx) (((uint32_t)OS_MPIC_INTR_GROUP_##type << 20) | ((mult_src_idx) << 12) | (idx))
- #define MPIC_INTR_TO_IDX(mpic_intr_id) (mpic_intr_id & 0xfff)

Enumerations

- enum os_mpic_intr_groups_t {
  OS_MPIC_INTR_GROUP_INTERNAL = 0, OS_MPIC_INTR_GROUP_EXTERNAL, OS_MP↩ IC_INTR_GROUP_TIMERS,
  OS_MPIC_INTR_GROUP_MSG, OS_MPIC_INTR_GROUP_SHARED_MSG, OS_MPIC_IN↩ TR_GROUP_IPI }
- enum os_mpic_intr_id_t {
  OS_MPIC_INTR_FM_ERR = MPIC_INTR(INTERNAL , 0 , 30), OS_MPIC_INTR_BM_ERR = MPIC_INTR(INTERNAL , 0 , 29), OS_MPIC_INTR_QM_ERR = MPIC_INTR(INTERNAL , 0 , 28),
  OS_MPIC_INTR_L2_CACHE_ERR = MPIC_INTR(INTERNAL , 0 , 22), OS_MPIC_INTR_R↩ MAN_ERR = MPIC_INTR(INTERNAL , 0 , 20), OS_MPIC_INTR_DDR_1_ERR = MPIC_IN↩ TR(INTERNAL , 0 , 9),
  OS_MPIC_INTR_DDR_0_ERR = MPIC_INTR(INTERNAL , 0 , 8), OS_MPIC_INTROS_MP↩ C_1_ERR = MPIC_INTR(INTERNAL , 0 , 5), OS_MPIC_INTR_CPC_0_ERR = MPIC_INTR(I↩ NTERNAL , 0 , 4),
  OS_MPIC_INTR_MURAM_ECC_ERR = MPIC_INTR(INTERNAL , 0 , 2), OS_MPIC_INTR↩

_PAMU_ERR = MPIC_INTR(INTERNAL , 0 , 1), OS_MPIC_INTR_CCF_ERR = MPIC_INT↩
R(INTERNAL , 0 , 0),
OS_MPIC_INTR_WATCHDOG = MPIC_INTR(INTERNAL , 1 , 0), OS_MPIC_INTR_TMU_0 = MPIC_INTR(INTERNAL , 2 , 0), OS_MPIC_INTR_TMU_1 = MPIC_INTR(INTERNAL , 3 , 0),
OS_MPIC_INTR_PCI_ERR = MPIC_INTR(INTERNAL , 4 , 0), OS_MPIC_INTR_PAMU = M↩
PIC_INTR(INTERNAL , 8 , 0), OS_MPIC_INTR_IFC = MPIC_INTR(INTERNAL , 9 , 0),
OS_MPIC_INTR_DMA_0_CH_0 = MPIC_INTR(INTERNAL , 12 , 0), OS_MPIC_INTR_DM↩
A_0_CH_1 = MPIC_INTR(INTERNAL , 13 , 0), OS_MPIC_INTR_DMA_0_CH_2 = MPIC_I↩
NTR(INTERNAL , 14 , 0),
OS_MPIC_INTR_DMA_0_CH_3 = MPIC_INTR(INTERNAL , 15 , 0), OS_MPIC_INTR_DM↩
A_1_CH_0 = MPIC_INTR(INTERNAL , 16 , 0), OS_MPIC_INTR_DMA_1_CH_1 = MPIC_I↩
NTR(INTERNAL , 17 , 0),
OS_MPIC_INTR_DMA_1_CH_2 = MPIC_INTR(INTERNAL , 18 , 0), OS_MPIC_INTR_DM↩
A_1_CH_3 = MPIC_INTR(INTERNAL , 19 , 0), OS_MPIC_INTR_DUART_UART_0 = MPIC↩
_INTR(INTERNAL , 20 , 0),
OS_MPIC_INTR_DUART_UART_1 = MPIC_INTR(INTERNAL , 20 , 1), OS_MPIC_INTR_↩
DUART_UART_2 = MPIC_INTR(INTERNAL , 21 , 0), OS_MPIC_INTR_DUART_UART_3 = MPIC_INTR(INTERNAL , 21 , 1),
OS_MPIC_INTR_I2C_0 = MPIC_INTR(INTERNAL , 22 , 0), OS_MPIC_INTR_I2C_1 = MPI↩
C_INTR(INTERNAL , 22 , 1), OS_MPIC_INTR_I2C_2 = MPIC_INTR(INTERNAL , 23 , 0),
OS_MPIC_INTR_I2C_3 = MPIC_INTR(INTERNAL , 23 , 1), OS_MPIC_INTR_PCIE_INTA = MPIC_INTR(INTERNAL , 24 , 0), OS_MPIC_INTR_USB = MPIC_INTR(INTERNAL , 28 , 0),
OS_MPIC_INTR_ESDHC = MPIC_INTR(INTERNAL , 32 , 0), OS_MPIC_INTR_PM = MPIC↩
_INTR(INTERNAL , 36 , 0), OS_MPIC_INTR_ESPI = MPIC_INTR(INTERNAL , 37 , 0),
OS_MPIC_INTR_GPIO_1 = MPIC_INTR(INTERNAL , 38 , 0), OS_MPIC_INTR_GPIO_0 = M↩
PIC_INTR(INTERNAL , 39 , 0), OS_MPIC_INTR_DMA_0_CH_4 = MPIC_INTR(INTERNAL , 60 , 0),
OS_MPIC_INTR_DMA_0_CH_5 = MPIC_INTR(INTERNAL , 61 , 0), OS_MPIC_INTR_DM↩
A_0_CH_6 = MPIC_INTR(INTERNAL , 62 , 0), OS_MPIC_INTR_DMA_0_CH_7 = MPIC_I↩
NTR(INTERNAL , 63 , 0),
OS_MPIC_INTR_DMA_1_CH_4 = MPIC_INTR(INTERNAL , 64 , 0), OS_MPIC_INTR_DM↩
A_1_CH_5 = MPIC_INTR(INTERNAL , 65 , 0), OS_MPIC_INTR_DMA_1_CH_6 = MPIC_I↩
NTR(INTERNAL , 66 , 0),
OS_MPIC_INTR_DMA_1_CH_7 = MPIC_INTR(INTERNAL , 67 , 0), OS_MPIC_INTR_EPU↩
_0 = MPIC_INTR(INTERNAL , 68 , 0), OS_MPIC_INTR_EPU_1 = MPIC_INTR(INTERNAL , 69 , 0),
OS_MPIC_INTR_GPIO_2 = MPIC_INTR(INTERNAL , 70 , 0), OS_MPIC_INTR_SEC_JQ_0 = MPIC_INTR(INTERNAL , 72 , 0), OS_MPIC_INTR_SEC_JQ_1 = MPIC_INTR(INTERNAL , 73 , 0),
OS_MPIC_INTR_SEC_JQ_2 = MPIC_INTR(INTERNAL , 74 , 0), OS_MPIC_INTR_SEC_JQ_3 = MPIC_INTR(INTERNAL , 75 , 0), OS_MPIC_INTR_SEC_RTIC = MPIC_INTR(INTERNAL , 76 , 0),
OS_MPIC_INTR_SEC_MONITOR = MPIC_INTR(INTERNAL , 77 , 0), OS_MPIC_INTR_EP↩
U_2 = MPIC_INTR(INTERNAL , 78 , 0), OS_MPIC_INTR_EPU_3 = MPIC_INTR(INTERNAL

, 79 , 0),
OS_MPIC_INTR_FM = MPIC_INTR(INTERNAL , 80 , 0), OS_MPIC_INTR_MDIO_0 = MPI↩
C_INTR(INTERNAL , 84 , 0), OS_MPIC_INTR_MDIO_1 = MPIC_INTR(INTERNAL , 85 , 0),
OS_MPIC_INTR_QM_PORTAL_0 = MPIC_INTR(INTERNAL , 88 , 0), OS_MPIC_INTR_B↩
M_PORTAL_0 = MPIC_INTR(INTERNAL , 89 , 0), OS_MPIC_INTR_QM_PORTAL_1 = MP↩
IC_INTR(INTERNAL , 90 , 0),
OS_MPIC_INTR_BM_PORTAL_1 = MPIC_INTR(INTERNAL , 91 , 0), OS_MPIC_INTR_Q↩
M_PORTAL_2 = MPIC_INTR(INTERNAL , 92 , 0), OS_MPIC_INTR_BM_PORTAL_2 = MP↩
IC_INTR(INTERNAL , 93 , 0),
OS_MPIC_INTR_QM_PORTAL_3 = MPIC_INTR(INTERNAL , 94 , 0), OS_MPIC_INTR_B↩
M_PORTAL_3 = MPIC_INTR(INTERNAL , 95 , 0), OS_MPIC_INTR_QM_PORTAL_4 = MP↩
IC_INTR(INTERNAL , 96 , 0),
OS_MPIC_INTR_BM_PORTAL_4 = MPIC_INTR(INTERNAL , 97 , 0), OS_MPIC_INTR_Q↩
M_PORTAL_5 = MPIC_INTR(INTERNAL , 98 , 0), OS_MPIC_INTR_BM_PORTAL_5 = MP↩
IC_INTR(INTERNAL , 99 , 0),
OS_MPIC_INTR_QM_PORTAL_6 = MPIC_INTR(INTERNAL , 100, 0), OS_MPIC_INTR_B↩
M_PORTAL_6 = MPIC_INTR(INTERNAL , 101, 0), OS_MPIC_INTR_QM_PORTAL_7 = M↩
PIC_INTR(INTERNAL , 102, 0),
OS_MPIC_INTR_BM_PORTAL_7 = MPIC_INTR(INTERNAL , 103, 0), OS_MPIC_INTR_Q↩
M_PORTAL_8 = MPIC_INTR(INTERNAL , 104, 0), OS_MPIC_INTR_BM_PORTAL_8 = M↩
PIC_INTR(INTERNAL , 105, 0),
OS_MPIC_INTR_QM_PORTAL_9 = MPIC_INTR(INTERNAL , 106, 0), OS_MPIC_INTR_↩
BM_PORTAL_9 = MPIC_INTR(INTERNAL , 107, 0), OS_MPIC_INTR_QM_PORTAL_10 =
MPIC_INTR(INTERNAL , 108, 0),
OS_MPIC_INTR_BM_PORTAL_10 = MPIC_INTR(INTERNAL , 109, 0), OS_MPIC_INTR_↩
QM_PORTAL_11 = MPIC_INTR(INTERNAL , 110, 0), OS_MPIC_INTR_BM_PORTAL_11 =
MPIC_INTR(INTERNAL , 111, 0),
OS_MPIC_INTR_QM_PORTAL_12 = MPIC_INTR(INTERNAL , 112, 0), OS_MPIC_INTR_↩
BM_PORTAL_12 = MPIC_INTR(INTERNAL , 113, 0), OS_MPIC_INTR_QM_PORTAL_13 =
MPIC_INTR(INTERNAL , 114, 0),
OS_MPIC_INTR_BM_PORTAL_13 = MPIC_INTR(INTERNAL , 115, 0), OS_MPIC_INTR_↩
QM_PORTAL_14 = MPIC_INTR(INTERNAL , 116, 0), OS_MPIC_INTR_BM_PORTAL_14 =
MPIC_INTR(INTERNAL , 117, 0),
OS_MPIC_INTR_QM_PORTAL_15 = MPIC_INTR(INTERNAL , 118, 0), OS_MPIC_INTR_↩
BM_PORTAL_15 = MPIC_INTR(INTERNAL , 119, 0), OS_MPIC_INTR_QM_PORTAL_16 =
MPIC_INTR(INTERNAL , 120, 0),
OS_MPIC_INTR_BM_PORTAL_16 = MPIC_INTR(INTERNAL , 121, 0), OS_MPIC_INTR_↩
QM_PORTAL_17 = MPIC_INTR(INTERNAL , 122, 0), OS_MPIC_INTR_BM_PORTAL_17 =
MPIC_INTR(INTERNAL , 123, 0),
OS_MPIC_INTR_QM_PORTAL_18 = MPIC_INTR(INTERNAL , 124, 0), OS_MPIC_INTR_↩
BM_PORTAL_18 = MPIC_INTR(INTERNAL , 125, 0), OS_MPIC_INTR_QM_PORTAL_19 =
MPIC_INTR(INTERNAL , 126, 0),
OS_MPIC_INTR_BM_PORTAL_19 = MPIC_INTR(INTERNAL , 127, 0), OS_MPIC_INTR_↩
QM_PORTAL_20 = MPIC_INTR(INTERNAL , 128 , 0), OS_MPIC_INTR_BM_PORTAL_20 =

MPIC_INTR(INTERNAL , 129 , 0),
OS_MPIC_INTR_QM_PORTAL_21 = MPIC_INTR(INTERNAL , 130 , 0), OS_MPIC_INTR_↩
BM_PORTAL_21 = MPIC_INTR(INTERNAL , 131 , 0), OS_MPIC_INTR_QM_PORTAL_22 =
MPIC_INTR(INTERNAL , 132 , 0),
OS_MPIC_INTR_BM_PORTAL_22 = MPIC_INTR(INTERNAL , 133 , 0), OS_MPIC_INTR_↩
QM_PORTAL_23 = MPIC_INTR(INTERNAL , 134 , 0), OS_MPIC_INTR_BM_PORTAL_23 =
MPIC_INTR(INTERNAL , 135 , 0),
OS_MPIC_INTR_QM_PORTAL_24 = MPIC_INTR(INTERNAL , 136 , 0), OS_MPIC_INTR_↩
BM_PORTAL_24 = MPIC_INTR(INTERNAL , 137 , 0), OS_MPIC_INTR_NUM_138 = MPIC↩
_INTR(INTERNAL , 138 , 0),
OS_MPIC_INTR_NUM_139 = MPIC_INTR(INTERNAL , 139 , 0), OS_MPIC_INTR_NUM_140
= MPIC_INTR(INTERNAL , 140 , 0), OS_MPIC_INTR_NUM_141 = MPIC_INTR(INTERNAL
, 141 , 0),
OS_MPIC_INTR_NUM_142 = MPIC_INTR(INTERNAL , 142 , 0), OS_MPIC_INTR_NUM_143
= MPIC_INTR(INTERNAL , 143 , 0), OS_MPIC_INTR_NUM_144 = MPIC_INTR(INTERNAL
, 144 , 0),
OS_MPIC_INTR_NUM_145 = MPIC_INTR(INTERNAL , 145 , 0), OS_MPIC_INTR_NUM_146
= MPIC_INTR(INTERNAL , 146 , 0), OS_MPIC_INTR_NUM_147 = MPIC_INTR(INTERNAL
, 147 , 0),
OS_MPIC_INTR_NUM_148 = MPIC_INTR(INTERNAL , 148 , 0), OS_MPIC_INTR_NUM_149
= MPIC_INTR(INTERNAL , 149 , 0), OS_MPIC_INTR_NUM_150 = MPIC_INTR(INTERNAL
, 150 , 0),
OS_MPIC_INTR_NUM_151 = MPIC_INTR(INTERNAL , 151 , 0), OS_MPIC_INTR_MAPL↩
E_LW0_BD_1 = MPIC_INTR(INTERNAL , 152 , 0), OS_MPIC_INTR_MAPLE_LW0_BD_2 =
MPIC_INTR(INTERNAL , 153 , 0),
OS_MPIC_INTR_MAPLE_LW0_BD_3 = MPIC_INTR(INTERNAL , 154 , 0), OS_MPIC_IN↩
TR_MAPLE_LW0_BD_4 = MPIC_INTR(INTERNAL , 155 , 0), OS_MPIC_INTR_MAPLE_L↩
W0_BD_5 = MPIC_INTR(INTERNAL , 156 , 0),
OS_MPIC_INTR_MAPLE_LW0_BD_6 = MPIC_INTR(INTERNAL , 157 , 0), OS_MPIC_IN↩
TR_MAPLE_LW0_BD_7 = MPIC_INTR(INTERNAL , 158 , 0), OS_MPIC_INTR_MAPLE_L↩
W0_BD_8 = MPIC_INTR(INTERNAL , 159 , 0),
OS_MPIC_INTR_MAPLE_LW0_BD_9 = MPIC_INTR(INTERNAL , 160 , 0), OS_MPIC_INT↩
R_MAPLE_LW0_BD_10 = MPIC_INTR(INTERNAL , 161 , 0), OS_MPIC_INTR_MAPLE_L↩
W0_BD_11 = MPIC_INTR(INTERNAL , 162 , 0),
OS_MPIC_INTR_MAPLE_LW0_BD_12 = MPIC_INTR(INTERNAL , 163 , 0), OS_MPIC_IN↩
TR_MAPLE_LW0_BD_13 = MPIC_INTR(INTERNAL , 164 , 0), OS_MPIC_INTR_MAPLE_↩
LW0_BD_14 = MPIC_INTR(INTERNAL , 165 , 0),
OS_MPIC_INTR_MAPLE_LW0_BD_15 = MPIC_INTR(INTERNAL , 166 , 0), OS_MPIC_IN↩
TR_MAPLE_LW0_BD_16 = MPIC_INTR(INTERNAL , 167 , 0), OS_MPIC_INTR_MAPLE_↩
LW1_BD_1 = MPIC_INTR(INTERNAL , 168 , 0),
OS_MPIC_INTR_MAPLE_LW1_BD_2 = MPIC_INTR(INTERNAL , 169 , 0), OS_MPIC_IN↩
TR_MAPLE_LW1_BD_3 = MPIC_INTR(INTERNAL , 170 , 0), OS_MPIC_INTR_MAPLE_L↩
W1_BD_4 = MPIC_INTR(INTERNAL , 171 , 0),
OS_MPIC_INTR_MAPLE_LW1_BD_5 = MPIC_INTR(INTERNAL , 172 , 0), OS_MPIC_IN↩
TR_MAPLE_LW1_BD_6 = MPIC_INTR(INTERNAL , 173 , 0), OS_MPIC_INTR_MAPLE_L↩

W1_BD_7 = MPIC_INTR(INTERNAL , 174 , 0),
OS_MPIC_INTR_MAPLE_LW1_BD_8 = MPIC_INTR(INTERNAL , 175 , 0), OS_MPIC_IN↩
TR_MAPLE_LW1_BD_9 = MPIC_INTR(INTERNAL , 176 , 0), OS_MPIC_INTR_MAPLE_L↩
W1_BD_10 = MPIC_INTR(INTERNAL , 177 , 0),
OS_MPIC_INTR_MAPLE_LW1_BD_11 = MPIC_INTR(INTERNAL , 178 , 0), OS_MPIC_IN↩
TR_MAPLE_LW1_BD_12 = MPIC_INTR(INTERNAL , 179 , 0), OS_MPIC_INTR_MAPLE_↩
LW1_BD_13 = MPIC_INTR(INTERNAL , 180 , 0),
OS_MPIC_INTR_MAPLE_LW1_BD_14 = MPIC_INTR(INTERNAL , 181 , 0), OS_MPIC_IN↩
TR_MAPLE_LW1_BD_15 = MPIC_INTR(INTERNAL , 182 , 0), OS_MPIC_INTR_MAPLE_↩
LW1_BD_16 = MPIC_INTR(INTERNAL , 183 , 0),
OS_MPIC_INTR_NUM_184 = MPIC_INTR(INTERNAL , 184 , 0), OS_MPIC_INTR_NUM_185
= MPIC_INTR(INTERNAL , 185 , 0), OS_MPIC_INTR_NUM_186 = MPIC_INTR(INTERNAL
, 186 , 0),
OS_MPIC_INTR_NUM_187 = MPIC_INTR(INTERNAL , 187 , 0), OS_MPIC_INTR_NUM_188
= MPIC_INTR(INTERNAL , 188 , 0), OS_MPIC_INTR_NUM_189 = MPIC_INTR(INTERNAL
, 189 , 0),
OS_MPIC_INTR_QTIMER_BLK_17 = MPIC_INTR(INTERNAL , 190 , 0), OS_MPIC_INTR_↩
QTIMER_BLK_18 = MPIC_INTR(INTERNAL , 191 , 0), OS_MPIC_INTR_QTIMER_BLK_19
= MPIC_INTR(INTERNAL , 192 , 0),
OS_MPIC_INTR_QTIMER_BLK_20 = MPIC_INTR(INTERNAL , 193 , 0), OS_MPIC_INTR_↩
QTIMER_BLK_21 = MPIC_INTR(INTERNAL , 194 , 0), OS_MPIC_INTR_QTIMER_BLK_22
= MPIC_INTR(INTERNAL , 195 , 0),
OS_MPIC_INTR_QTIMER_BLK_23 = MPIC_INTR(INTERNAL , 196 , 0), OS_MPIC_INTR_↩
QTIMER_BLK_24 = MPIC_INTR(INTERNAL , 197 , 0), OS_MPIC_INTR_QTIMER_BLK_25
= MPIC_INTR(INTERNAL , 198 , 0),
OS_MPIC_INTR_QTIMER_BLK_26 = MPIC_INTR(INTERNAL , 199 , 0), OS_MPIC_INTR_↩
QTIMER_BLK_27 = MPIC_INTR(INTERNAL , 200 , 0), OS_MPIC_INTR_QTIMER_BLK_28
= MPIC_INTR(INTERNAL , 201 , 0),
OS_MPIC_INTR_QTIMER_BLK_29 = MPIC_INTR(INTERNAL , 202 , 0), OS_MPIC_INTR_↩
QTIMER_BLK_30 = MPIC_INTR(INTERNAL , 203 , 0), OS_MPIC_INTR_QTIMER_BLK_31
= MPIC_INTR(INTERNAL , 204 , 0),
OS_MPIC_INTR_QTIMER_BLK_32 = MPIC_INTR(INTERNAL , 205 , 0), OS_MPIC_INTR↩
_NUM_206 = MPIC_INTR(INTERNAL , 206 , 0), OS_MPIC_INTR_NUM_207 = MPIC_INT↩
R(INTERNAL , 207 , 0),
OS_MPIC_INTR_NUM_208 = MPIC_INTR(INTERNAL , 208 , 0), OS_MPIC_INTR_NUM_209
= MPIC_INTR(INTERNAL , 209 , 0), OS_MPIC_INTR_NUM_210 = MPIC_INTR(INTERNAL
, 210 , 0),
OS_MPIC_INTR_NUM_211 = MPIC_INTR(INTERNAL , 211 , 0), OS_MPIC_INTR_NUM_212
= MPIC_INTR(INTERNAL , 212 , 0), OS_MPIC_INTR_NUM_213 = MPIC_INTR(INTERNAL
, 213 , 0),
OS_MPIC_INTR_NUM_214 = MPIC_INTR(INTERNAL , 214 , 0), OS_MPIC_INTR_NUM_215
= MPIC_INTR(INTERNAL , 215 , 0), OS_MPIC_INTR_NUM_216 = MPIC_INTR(INTERNAL
, 216 , 0),
OS_MPIC_INTR_NUM_217 = MPIC_INTR(INTERNAL , 217 , 0), OS_MPIC_INTR_NUM_218
= MPIC_INTR(INTERNAL , 218 , 0), OS_MPIC_INTR_NUM_219 = MPIC_INTR(INTERNAL

, 219 , 0),
OS_MPIC_INTR_NUM_220 = MPIC_INTR(INTERNAL , 220 , 0), OS_MPIC_INTR_NUM_221 = MPIC_INTR(INTERNAL , 221 , 0), OS_MPIC_INTR_NUM_222 = MPIC_INTR(INTERNAL , 222 , 0),
OS_MPIC_INTR_NUM_223 = MPIC_INTR(INTERNAL , 223 , 0), OS_MPIC_INTR_NUM_224 = MPIC_INTR(INTERNAL , 224 , 0), OS_MPIC_INTR_NUM_225 = MPIC_INTR(INTERNAL , 225 , 0),
OS_MPIC_INTR_NUM_226 = MPIC_INTR(INTERNAL , 226 , 0), OS_MPIC_INTR_NUM_227 = MPIC_INTR(INTERNAL , 227 , 0), OS_MPIC_INTR_NUM_228 = MPIC_INTR(INTERNAL , 228 , 0),
OS_MPIC_INTR_NUM_229 = MPIC_INTR(INTERNAL , 229 , 0), OS_MPIC_INTR_NUM_230 = MPIC_INTR(INTERNAL , 230 , 0), OS_MPIC_INTR_NUM_231 = MPIC_INTR(INTERNAL , 231 , 0),
OS_MPIC_INTR_NUM_232 = MPIC_INTR(INTERNAL , 232 , 0), OS_MPIC_INTR_NUM_233 = MPIC_INTR(INTERNAL , 233 , 0), OS_MPIC_INTR_NUM_234 = MPIC_INTR(INTERNAL , 234 , 0),
OS_MPIC_INTR_NUM_235 = MPIC_INTR(INTERNAL , 235 , 0), OS_MPIC_INTR_NUM_236 = MPIC_INTR(INTERNAL , 236 , 0), OS_MPIC_INTR_NUM_237 = MPIC_INTR(INTERNAL , 237 , 0),
OS_MPIC_INTR_NUM_238 = MPIC_INTR(INTERNAL , 238 , 0), OS_MPIC_INTR_NUM_239 = MPIC_INTR(INTERNAL , 239 , 0), OS_MPIC_INTR_NUM_240 = MPIC_INTR(INTERNAL , 240 , 0),
OS_MPIC_INTR_NUM_241 = MPIC_INTR(INTERNAL , 241 , 0), OS_MPIC_INTR_NUM_242 = MPIC_INTR(INTERNAL , 242 , 0), OS_MPIC_INTR_NUM_243 = MPIC_INTR(INTERNAL , 243 , 0),
OS_MPIC_INTR_NUM_244 = MPIC_INTR(INTERNAL , 244 , 0), OS_MPIC_INTR_NUM_245 = MPIC_INTR(INTERNAL , 245 , 0), OS_MPIC_INTR_NUM_246 = MPIC_INTR(INTERNAL , 246 , 0),
OS_MPIC_INTR_NUM_247 = MPIC_INTR(INTERNAL , 247 , 0), OS_MPIC_INTR_NUM_248 = MPIC_INTR(INTERNAL , 248 , 0), OS_MPIC_INTR_NUM_249 = MPIC_INTR(INTERNAL , 249 , 0),
OS_MPIC_INTR_NUM_250 = MPIC_INTR(INTERNAL , 250 , 0), OS_MPIC_INTR_NUM_251 = MPIC_INTR(INTERNAL , 251 , 0), OS_MPIC_INTR_NUM_252 = MPIC_INTR(INTERNAL , 252 , 0),
OS_MPIC_INTR_NUM_253 = MPIC_INTR(INTERNAL , 253 , 0), OS_MPIC_INTR_NUM_254 = MPIC_INTR(INTERNAL , 254 , 0), OS_MPIC_INTR_NUM_255 = MPIC_INTR(INTERNAL , 255 , 0),
OS_MPIC_INTR_IRQ_0 = MPIC_INTR(EXTERNAL , 0 , 0), OS_MPIC_INTR_IRQ_1 = MPIC_INTR(EXTERNAL , 1 , 0), OS_MPIC_INTR_IRQ_2 = MPIC_INTR(EXTERNAL , 2 , 0),
OS_MPIC_INTR_IRQ_3 = MPIC_INTR(EXTERNAL , 3 , 0), OS_MPIC_INTR_IRQ_4 = MPIC_INTR(EXTERNAL , 4 , 0), OS_MPIC_INTR_IRQ_5 = MPIC_INTR(EXTERNAL , 5 , 0),
OS_MPIC_INTR_IRQ_6 = MPIC_INTR(EXTERNAL , 6 , 0), OS_MPIC_INTR_IRQ_7 = MPIC_INTR(EXTERNAL , 7 , 0), OS_MPIC_INTR_IRQ_8 = MPIC_INTR(EXTERNAL , 8 , 0),
OS_MPIC_INTR_IRQ_9 = MPIC_INTR(EXTERNAL , 9 , 0), OS_MPIC_INTR_IRQ_10 = MPIC_INTR(EXTERNAL , 10 , 0), OS_MPIC_INTR_IRQ_11 = MPIC_INTR(EXTERNAL , 11 , 0

),
OS_MPIC_INTR_TIMER_0 = MPIC_INTR(TIMERS , 0 , 0), OS_MPIC_INTR_TIMER_1 = M↩
PIC_INTR(TIMERS , 1 , 0), OS_MPIC_INTR_TIMER_2 = MPIC_INTR(TIMERS , 2 , 0),
OS_MPIC_INTR_TIMER_3 = MPIC_INTR(TIMERS , 3 , 0), OS_MPIC_INTR_TIMER_4 = M↩
PIC_INTR(TIMERS , 4 , 0), OS_MPIC_INTR_TIMER_5 = MPIC_INTR(TIMERS , 5 , 0),
OS_MPIC_INTR_TIMER_6 = MPIC_INTR(TIMERS , 6 , 0), OS_MPIC_INTR_TIMER_7 = M↩
PIC_INTR(TIMERS , 7 , 0), OS_MPIC_INTR_MSG_0 = MPIC_INTR(MSG , 0 , 0),
OS_MPIC_INTR_MSG_1 = MPIC_INTR(MSG , 1 , 0), OS_MPIC_INTR_MSG_2 = MPIC_IN↩
TR(MSG , 2 , 0), OS_MPIC_INTR_MSG_3 = MPIC_INTR(MSG , 3 , 0),
OS_MPIC_INTR_MSG_4 = MPIC_INTR(MSG , 4 , 0), OS_MPIC_INTR_MSG_5 = MPIC_IN↩
TR(MSG , 5 , 0), OS_MPIC_INTR_MSG_6 = MPIC_INTR(MSG , 6 , 0),
OS_MPIC_INTR_MSG_7 = MPIC_INTR(MSG , 7 , 0), OS_MPIC_INTR_MSG_8 = MPIC_IN↩
TR(MSG , 8 , 0), OS_MPIC_INTR_MSG_9 = MPIC_INTR(MSG , 9 , 0),
OS_MPIC_INTR_MSG_10 = MPIC_INTR(MSG , 10 , 0), OS_MPIC_INTR_MSG_11 = MPIC↩
_INTR(MSG , 11 , 0), OS_MPIC_INTR_MSG_12 = MPIC_INTR(MSG , 12 , 0),
OS_MPIC_INTR_MSG_13 = MPIC_INTR(MSG , 13 , 0), OS_MPIC_INTR_MSG_14 = MPIC↩
_INTR(MSG , 14 , 0), OS_MPIC_INTR_MSG_15 = MPIC_INTR(MSG , 15 , 0),
OS_MPIC_INTR_SHARED_MSG_0 = MPIC_INTR(SHARED_MSG , 0 , 0), OS_MPIC_INT↩
R_SHARED_MSG_1 = MPIC_INTR(SHARED_MSG , 1 , 0), OS_MPIC_INTR_SHARED_M↩
SG_2 = MPIC_INTR(SHARED_MSG , 2 , 0),
OS_MPIC_INTR_SHARED_MSG_3 = MPIC_INTR(SHARED_MSG , 3 , 0), OS_MPIC_INT↩
R_SHARED_MSG_4 = MPIC_INTR(SHARED_MSG , 4 , 0), OS_MPIC_INTR_SHARED_M↩
SG_5 = MPIC_INTR(SHARED_MSG , 5 , 0),
OS_MPIC_INTR_SHARED_MSG_6 = MPIC_INTR(SHARED_MSG , 6 , 0), OS_MPIC_INT↩
R_SHARED_MSG_7 = MPIC_INTR(SHARED_MSG , 7 , 0), OS_MPIC_INTR_SHARED_M↩
SG_8 = MPIC_INTR(SHARED_MSG , 8 , 0),
OS_MPIC_INTR_SHARED_MSG_9 = MPIC_INTR(SHARED_MSG , 9 , 0), OS_MPIC_INT↩
R_SHARED_MSG_10 = MPIC_INTR(SHARED_MSG , 10 , 0), OS_MPIC_INTR_SHARED_↩
MSG_11 = MPIC_INTR(SHARED_MSG , 11 , 0),
OS_MPIC_INTR_SHARED_MSG_12 = MPIC_INTR(SHARED_MSG , 12 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_13 = MPIC_INTR(SHARED_MSG , 13 , 0), OS_MPIC_INTR_SHARED↩
_MSG_14 = MPIC_INTR(SHARED_MSG , 14 , 0),
OS_MPIC_INTR_SHARED_MSG_15 = MPIC_INTR(SHARED_MSG , 15 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_16 = MPIC_INTR(SHARED_MSG , 16 , 0), OS_MPIC_INTR_SHARED↩
_MSG_17 = MPIC_INTR(SHARED_MSG , 17 , 0),
OS_MPIC_INTR_SHARED_MSG_18 = MPIC_INTR(SHARED_MSG , 18 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_19 = MPIC_INTR(SHARED_MSG , 19 , 0), OS_MPIC_INTR_SHARED↩
_MSG_20 = MPIC_INTR(SHARED_MSG , 20 , 0),
OS_MPIC_INTR_SHARED_MSG_21 = MPIC_INTR(SHARED_MSG , 21 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_22 = MPIC_INTR(SHARED_MSG , 22 , 0), OS_MPIC_INTR_SHARED↩
_MSG_23 = MPIC_INTR(SHARED_MSG , 23 , 0),
OS_MPIC_INTR_SHARED_MSG_24 = MPIC_INTR(SHARED_MSG , 24 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_25 = MPIC_INTR(SHARED_MSG , 25 , 0), OS_MPIC_INTR_SHARED↩
_MSG_26 = MPIC_INTR(SHARED_MSG , 26 , 0),
OS_MPIC_INTR_SHARED_MSG_27 = MPIC_INTR(SHARED_MSG , 27 , 0), OS_MPIC_IN↩

TR_SHARED_MSG_28 = MPIC_INTR(SHARED_MSG , 28 , 0), OS_MPIC_INTR_SHARED↩
_MSG_29 = MPIC_INTR(SHARED_MSG , 29 , 0),
OS_MPIC_INTR_SHARED_MSG_30 = MPIC_INTR(SHARED_MSG , 30 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_31 = MPIC_INTR(SHARED_MSG , 31 , 0), OS_MPIC_INTR_SHARED↩
_MSG_32 = MPIC_INTR(SHARED_MSG , 32 , 0),
OS_MPIC_INTR_SHARED_MSG_33 = MPIC_INTR(SHARED_MSG , 33 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_34 = MPIC_INTR(SHARED_MSG , 34 , 0), OS_MPIC_INTR_SHARED↩
_MSG_35 = MPIC_INTR(SHARED_MSG , 35 , 0),
OS_MPIC_INTR_SHARED_MSG_36 = MPIC_INTR(SHARED_MSG , 36 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_37 = MPIC_INTR(SHARED_MSG , 37 , 0), OS_MPIC_INTR_SHARED↩
_MSG_38 = MPIC_INTR(SHARED_MSG , 38 , 0),
OS_MPIC_INTR_SHARED_MSG_39 = MPIC_INTR(SHARED_MSG , 39 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_40 = MPIC_INTR(SHARED_MSG , 40 , 0), OS_MPIC_INTR_SHARED↩
_MSG_41 = MPIC_INTR(SHARED_MSG , 41 , 0),
OS_MPIC_INTR_SHARED_MSG_42 = MPIC_INTR(SHARED_MSG , 42 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_43 = MPIC_INTR(SHARED_MSG , 43 , 0), OS_MPIC_INTR_SHARED↩
_MSG_44 = MPIC_INTR(SHARED_MSG , 44 , 0),
OS_MPIC_INTR_SHARED_MSG_45 = MPIC_INTR(SHARED_MSG , 45 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_46 = MPIC_INTR(SHARED_MSG , 46 , 0), OS_MPIC_INTR_SHARED↩
_MSG_47 = MPIC_INTR(SHARED_MSG , 47 , 0),
OS_MPIC_INTR_SHARED_MSG_48 = MPIC_INTR(SHARED_MSG , 48 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_49 = MPIC_INTR(SHARED_MSG , 49 , 0), OS_MPIC_INTR_SHARED↩
_MSG_50 = MPIC_INTR(SHARED_MSG , 50 , 0),
OS_MPIC_INTR_SHARED_MSG_51 = MPIC_INTR(SHARED_MSG , 51 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_52 = MPIC_INTR(SHARED_MSG , 52 , 0), OS_MPIC_INTR_SHARED↩
_MSG_53 = MPIC_INTR(SHARED_MSG , 53 , 0),
OS_MPIC_INTR_SHARED_MSG_54 = MPIC_INTR(SHARED_MSG , 54 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_55 = MPIC_INTR(SHARED_MSG , 55 , 0), OS_MPIC_INTR_SHARED↩
_MSG_56 = MPIC_INTR(SHARED_MSG , 56 , 0),
OS_MPIC_INTR_SHARED_MSG_57 = MPIC_INTR(SHARED_MSG , 57 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_58 = MPIC_INTR(SHARED_MSG , 58 , 0), OS_MPIC_INTR_SHARED↩
_MSG_59 = MPIC_INTR(SHARED_MSG , 59 , 0),
OS_MPIC_INTR_SHARED_MSG_60 = MPIC_INTR(SHARED_MSG , 60 , 0), OS_MPIC_IN↩
TR_SHARED_MSG_61 = MPIC_INTR(SHARED_MSG , 61 , 0), OS_MPIC_INTR_SHARED↩
_MSG_62 = MPIC_INTR(SHARED_MSG , 62 , 0),
OS_MPIC_INTR_SHARED_MSG_63 = MPIC_INTR(SHARED_MSG , 63 , 0), OS_MPIC_IN↩
TR_IPI_0 = MPIC_INTR(IPI , 0 , 0), OS_MPIC_INTR_IPI_1 = MPIC_INTR(IPI , 1 , 0),
OS_MPIC_INTR_IPI_2 = MPIC_INTR(IPI , 2 , 0), OS_MPIC_INTR_IPI_3 = MPIC_INTR(IPI ,
3 , 0) }

2.14.2.2 Macro Definition Documentation

2.14.2.2.1 #define NUM_OF_MPIC_IPI_CORES (OS_SOC_MAX_NUM_OF_CORES + OS_SOC_MAX_NUM_OF_PA_THREADS)

The amount of mpic IPI supported cores.

2.14.2.2.2 #define MPIC_NUM_OF_EXT_INTRS 12

The amount of External interrupts.

2.14.2.2.3 #define MPIC_NUM_OF_INT_INTRS 256

The amount of internal interrupts.

2.14.2.2.4 #define MPIC_NUM_OF_TIMERS 8

The amount of Timers interrupts.

2.14.2.2.5 #define MPIC_NUM_OF_MSG_INTRS 16

The amount of Massage interrupts.

2.14.2.2.6 #define MPIC_NUM_OF_SMSG_INTRS 64

The amount of MSI interrupts.

2.14.2.2.7 #define MPIC_NUM_OF_IPI_EVENTS 4

The amount of IPI interrupts.

2.14.2.2.8 #define MPIC_NUM_OF_LOW_INT_INTRS 160

The amount of low internal interrupts.

2.14.2.2.9 #define MPIC_MAX_NUM_OF_INTR_SRC

Maximum number of interrupt source, IPI events takes into account all chip's core, both PowerPC and StarCore.

2.14.2.2.10 #define MPIC_MULTI_SRC_INTERNAL_INTERRUPTS

number of interrupt source per interrupt index

2.14.2.2.11 #define MPIC_MULTI_SRC_INTERNAL_INTERRUPTS_COUNT 40

number of multi-source interrupts

2.14.2.2.12 #define MPIC_INTR(*type, idx, mult_src_idx*) (((uint32_t)O↩ S_MPIC_INTR_GROUP_##type << 20) | ((mult_src_idx) << 12) | (idx))

interrupt id [group | multi-source index | interrupt index]

2.14.2.2.13 #define MPIC_INTR_TO_IDX(*mpic_intr_id*) (mpic_intr_id & 0xfff)

returns the interrupt id in the group from the interrupt general id

2.14.2.3 Enumeration Type Documentation

2.14.2.3.1 enum os_mpic_intr_groups_t

MPIC interrupt groups.

This enum is not directly used in the MPIC API, but represents the internal enumeration of interrupt groups and thus needed for the definition of MPIC interrupt IDs (see mpic_intr_id).

Enumerator

OS_MPIC_INTR_GROUP_INTERNAL Internal interrupts group.
OS_MPIC_INTR_GROUP_EXTERNAL External interrupts group.
OS_MPIC_INTR_GROUP_TIMERS Timers interrupts group.
OS_MPIC_INTR_GROUP_MSG Massage interrupts group.
OS_MPIC_INTR_GROUP_SHARED_MSG MSI interrupts group.
OS_MPIC_INTR_GROUP_IPI IPI interrupts group.

2.14.2.3.2 enum os_mpic_intr_id_t

Enumeration of MPIC interrupts.

Enumerator

OS_MPIC_INTR_FM_ERR Frame manager error.

OS_MPIC_INTR_BM_ERR Buffer manager error.
OS_MPIC_INTR_QM_ERR Queue manager error.
OS_MPIC_INTR_L2_CACHE_ERR L2 Cache error.
OS_MPIC_INTR_RMAN_ERR RIO message manager error.
OS_MPIC_INTR_DDR_1_ERR Memory controller 1 error.
OS_MPIC_INTR_DDR_0_ERR Memory controller 0 error.
OS_MPIC_INTROS_MPC_1_ERR CoreNet platform cache 1 (CPC1) error.
OS_MPIC_INTR_CPC_0_ERR CoreNet platform cache 0 (CPC0) error.
OS_MPIC_INTR_MURAM_ECC_ERR Internal RAM multi-bit ECC error.
OS_MPIC_INTR_PAMU_ERR PAMU hardware error.
OS_MPIC_INTR_CCF_ERR CoreNet coherency fabric (CCF) error.
OS_MPIC_INTR_WATCHDOG Watchdog interrupts.
OS_MPIC_INTR_TMU_0 Termal monitor unit 0.
OS_MPIC_INTR_TMU_1 Termal monitor unit 1.
OS_MPIC_INTR_PCI_ERR PCI error.
OS_MPIC_INTR_PAMU PAMU (access violations)
OS_MPIC_INTR_IFC IFC.
OS_MPIC_INTR_DMA_0_CH_0 DMA0 channel 0.
OS_MPIC_INTR_DMA_0_CH_1 DMA0 channel 1.
OS_MPIC_INTR_DMA_0_CH_2 DMA0 channel 2.
OS_MPIC_INTR_DMA_0_CH_3 DMA0 channel 3.
OS_MPIC_INTR_DMA_1_CH_0 DMA1 channel 0.
OS_MPIC_INTR_DMA_1_CH_1 DMA1 channel 1.
OS_MPIC_INTR_DMA_1_CH_2 DMA1 channel 2.
OS_MPIC_INTR_DMA_1_CH_3 DMA1 channel 3.
OS_MPIC_INTR_DUART_UART_0 DUART0 UART0.
OS_MPIC_INTR_DUART_UART_1 DUART0 UART1.
OS_MPIC_INTR_DUART_UART_2 DUART1 UART2.
OS_MPIC_INTR_DUART_UART_3 DUART1 UART3.
OS_MPIC_INTR_I2C_0 I2C0 controller 0.
OS_MPIC_INTR_I2C_1 I2C0 controller 1.
OS_MPIC_INTR_I2C_2 I2C1 controller 2.
OS_MPIC_INTR_I2C_3 I2C1 controller 3.
OS_MPIC_INTR_PCIE_INTA PCI Express INTA.
OS_MPIC_INTR_USB USB controller.
OS_MPIC_INTR_ESDHC eSDHC
OS_MPIC_INTR_PM Performance monitor.
OS_MPIC_INTR_ESPI eSPI
OS_MPIC_INTR_GPIO_1 GPIO 1.
OS_MPIC_INTR_GPIO_0 GPIO 0.
OS_MPIC_INTR_DMA_0_CH_4 DMA0 channel 4.
OS_MPIC_INTR_DMA_0_CH_5 DMA0 channel 5.
OS_MPIC_INTR_DMA_0_CH_6 DMA0 channel 6.
OS_MPIC_INTR_DMA_0_CH_7 DMA0 channel 7.
OS_MPIC_INTR_DMA_1_CH_4 DMA1 channel 4.

OS_MPIC_INTR_DMA_1_CH_5 DMA1 channel 5.
OS_MPIC_INTR_DMA_1_CH_6 DMA1 channel 6.
OS_MPIC_INTR_DMA_1_CH_7 DMA1 channel 7.
OS_MPIC_INTR_EPU_0 Event processing unit 0.
OS_MPIC_INTR_EPU_1 Event processing unit 1.
OS_MPIC_INTR_GPIO_2 GPIO 2.
OS_MPIC_INTR_SEC_JQ_0 SEC JQ-0.
OS_MPIC_INTR_SEC_JQ_1 SEC JQ-1.
OS_MPIC_INTR_SEC_JQ_2 SEC JQ-2.
OS_MPIC_INTR_SEC_JQ_3 SEC JQ-3.
OS_MPIC_INTR_SEC_RTIC SEC global error.
OS_MPIC_INTR_SEC_MONITOR Security monitor.
OS_MPIC_INTR_EPU_2 Event processing unit 2.
OS_MPIC_INTR_EPU_3 Event processing unit 3.
OS_MPIC_INTR_FM Frame Manager.
OS_MPIC_INTR_MDIO_0 MDIO 0.
OS_MPIC_INTR_MDIO_1 MDIO 1.
OS_MPIC_INTR_QM_PORTAL_0 Queue Manager Portal 0.
OS_MPIC_INTR_BM_PORTAL_0 Buffer Manager Portal 0.
OS_MPIC_INTR_QM_PORTAL_1 Queue Manager Portal 1.
OS_MPIC_INTR_BM_PORTAL_1 Buffer Manager Portal 1.
OS_MPIC_INTR_QM_PORTAL_2 Queue Manager Portal 2.
OS_MPIC_INTR_BM_PORTAL_2 Buffer Manager Portal 2.
OS_MPIC_INTR_QM_PORTAL_3 Queue Manager Portal 3.
OS_MPIC_INTR_BM_PORTAL_3 Buffer Manager Portal 3.
OS_MPIC_INTR_QM_PORTAL_4 Queue Manager Portal 4.
OS_MPIC_INTR_BM_PORTAL_4 Buffer Manager Portal 4.
OS_MPIC_INTR_QM_PORTAL_5 Queue Manager Portal 5.
OS_MPIC_INTR_BM_PORTAL_5 Buffer Manager Portal 5.
OS_MPIC_INTR_QM_PORTAL_6 Queue Manager Portal 6.
OS_MPIC_INTR_BM_PORTAL_6 Buffer Manager Portal 6.
OS_MPIC_INTR_QM_PORTAL_7 Queue Manager Portal 7.
OS_MPIC_INTR_BM_PORTAL_7 Buffer Manager Portal 7.
OS_MPIC_INTR_QM_PORTAL_8 Queue Manager Portal 8.
OS_MPIC_INTR_BM_PORTAL_8 Buffer Manager Portal 8.
OS_MPIC_INTR_QM_PORTAL_9 Queue Manager Portal 9.
OS_MPIC_INTR_BM_PORTAL_9 Buffer Manager Portal 9.
OS_MPIC_INTR_QM_PORTAL_10 Queue Manager Portal 10.
OS_MPIC_INTR_BM_PORTAL_10 Buffer Manager Portal 10.
OS_MPIC_INTR_QM_PORTAL_11 Queue Manager Portal 11.
OS_MPIC_INTR_BM_PORTAL_11 Buffer Manager Portal 11.
OS_MPIC_INTR_QM_PORTAL_12 Queue Manager Portal 12.
OS_MPIC_INTR_BM_PORTAL_12 Buffer Manager Portal 12.
OS_MPIC_INTR_QM_PORTAL_13 Queue Manager Portal 13.
OS_MPIC_INTR_BM_PORTAL_13 Buffer Manager Portal 13.

OS_MPIC_INTR_QM_PORTAL_14 Queue Manager Portal 14.
OS_MPIC_INTR_BM_PORTAL_14 Buffer Manager Portal 14.
OS_MPIC_INTR_QM_PORTAL_15 Queue Manager Portal 15.
OS_MPIC_INTR_BM_PORTAL_15 Buffer Manager Portal 15.
OS_MPIC_INTR_QM_PORTAL_16 Queue Manager Portal 16.
OS_MPIC_INTR_BM_PORTAL_16 Buffer Manager Portal 16.
OS_MPIC_INTR_QM_PORTAL_17 Queue Manager Portal 17.
OS_MPIC_INTR_BM_PORTAL_17 Buffer Manager Portal 17.
OS_MPIC_INTR_QM_PORTAL_18 Queue Manager Portal 18.
OS_MPIC_INTR_BM_PORTAL_18 Buffer Manager Portal 18.
OS_MPIC_INTR_QM_PORTAL_19 Queue Manager Portal 19.
OS_MPIC_INTR_BM_PORTAL_19 Buffer Manager Portal 19.
OS_MPIC_INTR_QM_PORTAL_20 Queue Manager Portal 20.
OS_MPIC_INTR_BM_PORTAL_20 Buffer Manager Portal 20.
OS_MPIC_INTR_QM_PORTAL_21 Queue Manager Portal 21.
OS_MPIC_INTR_BM_PORTAL_21 Buffer Manager Portal 21.
OS_MPIC_INTR_QM_PORTAL_22 Queue Manager Portal 22.
OS_MPIC_INTR_BM_PORTAL_22 Buffer Manager Portal 22.
OS_MPIC_INTR_QM_PORTAL_23 Queue Manager Portal 23.
OS_MPIC_INTR_BM_PORTAL_23 Buffer Manager Portal 23.
OS_MPIC_INTR_QM_PORTAL_24 Queue Manager Portal 24.
OS_MPIC_INTR_BM_PORTAL_24 Buffer Manager Portal 24.
OS_MPIC_INTR_NUM_138 internal interrupt 138
OS_MPIC_INTR_NUM_139 internal interrupt 139
OS_MPIC_INTR_NUM_140 internal interrupt 140
OS_MPIC_INTR_NUM_141 internal interrupt 141
OS_MPIC_INTR_NUM_142 internal interrupt 142
OS_MPIC_INTR_NUM_143 internal interrupt 143
OS_MPIC_INTR_NUM_144 internal interrupt 144
OS_MPIC_INTR_NUM_145 internal interrupt 145
OS_MPIC_INTR_NUM_146 internal interrupt 146
OS_MPIC_INTR_NUM_147 internal interrupt 147
OS_MPIC_INTR_NUM_148 internal interrupt 148
OS_MPIC_INTR_NUM_149 internal interrupt 149
OS_MPIC_INTR_NUM_150 internal interrupt 150
OS_MPIC_INTR_NUM_151 internal interrupt 151
OS_MPIC_INTR_MAPLE_LW0_BD_1 MAPLE-LW0 BD 1.
OS_MPIC_INTR_MAPLE_LW0_BD_2 MAPLE-LW0 BD 2.
OS_MPIC_INTR_MAPLE_LW0_BD_3 MAPLE-LW0 BD 3.
OS_MPIC_INTR_MAPLE_LW0_BD_4 MAPLE-LW0 BD 4.
OS_MPIC_INTR_MAPLE_LW0_BD_5 MAPLE-LW0 BD 5.
OS_MPIC_INTR_MAPLE_LW0_BD_6 MAPLE-LW0 BD 6.
OS_MPIC_INTR_MAPLE_LW0_BD_7 MAPLE-LW0 BD 7.
OS_MPIC_INTR_MAPLE_LW0_BD_8 MAPLE-LW0 BD 8.
OS_MPIC_INTR_MAPLE_LW0_BD_9 MAPLE-LW0 BD 9.

OS_MPIC_INTR_MAPLE_LW0_BD_10 MAPLE-LW0 BD 10.
OS_MPIC_INTR_MAPLE_LW0_BD_11 MAPLE-LW0 BD 11.
OS_MPIC_INTR_MAPLE_LW0_BD_12 MAPLE-LW0 BD 12.
OS_MPIC_INTR_MAPLE_LW0_BD_13 MAPLE-LW0 BD 13.
OS_MPIC_INTR_MAPLE_LW0_BD_14 MAPLE-LW0 BD 14.
OS_MPIC_INTR_MAPLE_LW0_BD_15 MAPLE-LW0 BD 15.
OS_MPIC_INTR_MAPLE_LW0_BD_16 MAPLE-LW0 BD 16.
OS_MPIC_INTR_MAPLE_LW1_BD_1 MAPLE-LW1 BD 1.
OS_MPIC_INTR_MAPLE_LW1_BD_2 MAPLE-LW1 BD 2.
OS_MPIC_INTR_MAPLE_LW1_BD_3 MAPLE-LW1 BD 3.
OS_MPIC_INTR_MAPLE_LW1_BD_4 MAPLE-LW1 BD 4.
OS_MPIC_INTR_MAPLE_LW1_BD_5 MAPLE-LW1 BD 5.
OS_MPIC_INTR_MAPLE_LW1_BD_6 MAPLE-LW1 BD 6.
OS_MPIC_INTR_MAPLE_LW1_BD_7 MAPLE-LW1 BD 7.
OS_MPIC_INTR_MAPLE_LW1_BD_8 MAPLE-LW1 BD 8.
OS_MPIC_INTR_MAPLE_LW1_BD_9 MAPLE-LW1 BD 9.
OS_MPIC_INTR_MAPLE_LW1_BD_10 MAPLE-LW1 BD 10.
OS_MPIC_INTR_MAPLE_LW1_BD_11 MAPLE-LW1 BD 11.
OS_MPIC_INTR_MAPLE_LW1_BD_12 MAPLE-LW1 BD 12.
OS_MPIC_INTR_MAPLE_LW1_BD_13 MAPLE-LW1 BD 13.
OS_MPIC_INTR_MAPLE_LW1_BD_14 MAPLE-LW1 BD 14.
OS_MPIC_INTR_MAPLE_LW1_BD_15 MAPLE-LW1 BD 15.
OS_MPIC_INTR_MAPLE_LW1_BD_16 MAPLE-LW1 BD 16.
OS_MPIC_INTR_NUM_184 internal interrupt 184
OS_MPIC_INTR_NUM_185 internal interrupt 185
OS_MPIC_INTR_NUM_186 internal interrupt 186
OS_MPIC_INTR_NUM_187 internal interrupt 187
OS_MPIC_INTR_NUM_188 internal interrupt 188
OS_MPIC_INTR_NUM_189 internal interrupt 189
OS_MPIC_INTR_QTIMER_BLK_17 Quad timer block 17.
OS_MPIC_INTR_QTIMER_BLK_18 Quad timer block 18.
OS_MPIC_INTR_QTIMER_BLK_19 Quad timer block 19.
OS_MPIC_INTR_QTIMER_BLK_20 Quad timer block 20.
OS_MPIC_INTR_QTIMER_BLK_21 Quad timer block 21.
OS_MPIC_INTR_QTIMER_BLK_22 Quad timer block 22.
OS_MPIC_INTR_QTIMER_BLK_23 Quad timer block 23.
OS_MPIC_INTR_QTIMER_BLK_24 Quad timer block 24.
OS_MPIC_INTR_QTIMER_BLK_25 Quad timer block 25.
OS_MPIC_INTR_QTIMER_BLK_26 Quad timer block 26.
OS_MPIC_INTR_QTIMER_BLK_27 Quad timer block 27.
OS_MPIC_INTR_QTIMER_BLK_28 Quad timer block 28.
OS_MPIC_INTR_QTIMER_BLK_29 Quad timer block 29.
OS_MPIC_INTR_QTIMER_BLK_30 Quad timer block 30.
OS_MPIC_INTR_QTIMER_BLK_31 Quad timer block 31.
OS_MPIC_INTR_QTIMER_BLK_32 Quad timer block 32.

OS_MPIC_INTR_NUM_206 internal interrupt 206
OS_MPIC_INTR_NUM_207 internal interrupt 207
OS_MPIC_INTR_NUM_208 internal interrupt 208
OS_MPIC_INTR_NUM_209 internal interrupt 209
OS_MPIC_INTR_NUM_210 internal interrupt 210
OS_MPIC_INTR_NUM_211 internal interrupt 211
OS_MPIC_INTR_NUM_212 internal interrupt 212
OS_MPIC_INTR_NUM_213 internal interrupt 213
OS_MPIC_INTR_NUM_214 internal interrupt 214
OS_MPIC_INTR_NUM_215 internal interrupt 215
OS_MPIC_INTR_NUM_216 internal interrupt 216
OS_MPIC_INTR_NUM_217 internal interrupt 217
OS_MPIC_INTR_NUM_218 internal interrupt 218
OS_MPIC_INTR_NUM_219 internal interrupt 219
OS_MPIC_INTR_NUM_220 internal interrupt 220
OS_MPIC_INTR_NUM_221 internal interrupt 221
OS_MPIC_INTR_NUM_222 internal interrupt 222
OS_MPIC_INTR_NUM_223 internal interrupt 223
OS_MPIC_INTR_NUM_224 internal interrupt 224
OS_MPIC_INTR_NUM_225 internal interrupt 225
OS_MPIC_INTR_NUM_226 internal interrupt 226
OS_MPIC_INTR_NUM_227 internal interrupt 227
OS_MPIC_INTR_NUM_228 internal interrupt 228
OS_MPIC_INTR_NUM_229 internal interrupt 229
OS_MPIC_INTR_NUM_230 internal interrupt 230
OS_MPIC_INTR_NUM_231 internal interrupt 231
OS_MPIC_INTR_NUM_232 internal interrupt 232
OS_MPIC_INTR_NUM_233 internal interrupt 233
OS_MPIC_INTR_NUM_234 internal interrupt 234
OS_MPIC_INTR_NUM_235 internal interrupt 235
OS_MPIC_INTR_NUM_236 internal interrupt 236
OS_MPIC_INTR_NUM_237 internal interrupt 237
OS_MPIC_INTR_NUM_238 internal interrupt 238
OS_MPIC_INTR_NUM_239 internal interrupt 239
OS_MPIC_INTR_NUM_240 internal interrupt 240
OS_MPIC_INTR_NUM_241 internal interrupt 241
OS_MPIC_INTR_NUM_242 internal interrupt 242
OS_MPIC_INTR_NUM_243 internal interrupt 243
OS_MPIC_INTR_NUM_244 internal interrupt 244
OS_MPIC_INTR_NUM_245 internal interrupt 245
OS_MPIC_INTR_NUM_246 internal interrupt 246
OS_MPIC_INTR_NUM_247 internal interrupt 247
OS_MPIC_INTR_NUM_248 internal interrupt 248
OS_MPIC_INTR_NUM_249 internal interrupt 249
OS_MPIC_INTR_NUM_250 internal interrupt 250

OS_MPIC_INTR_NUM_251 internal interrupt 251
OS_MPIC_INTR_NUM_252 internal interrupt 252
OS_MPIC_INTR_NUM_253 internal interrupt 253
OS_MPIC_INTR_NUM_254 internal interrupt 254
OS_MPIC_INTR_NUM_255 internal interrupt 255
OS_MPIC_INTR_IRQ_0 External interrupt 0.
OS_MPIC_INTR_IRQ_1 External interrupt 1.
OS_MPIC_INTR_IRQ_2 External interrupt 2.
OS_MPIC_INTR_IRQ_3 External interrupt 3.
OS_MPIC_INTR_IRQ_4 External interrupt 4.
OS_MPIC_INTR_IRQ_5 External interrupt 5.
OS_MPIC_INTR_IRQ_6 External interrupt 6.
OS_MPIC_INTR_IRQ_7 External interrupt 7.
OS_MPIC_INTR_IRQ_8 External interrupt 8.
OS_MPIC_INTR_IRQ_9 External interrupt 9.
OS_MPIC_INTR_IRQ_10 External interrupt 10.
OS_MPIC_INTR_IRQ_11 External interrupt 11.
OS_MPIC_INTR_TIMER_0 Timer interrupt 0.
OS_MPIC_INTR_TIMER_1 Timer interrupt 1.
OS_MPIC_INTR_TIMER_2 Timer interrupt 2.
OS_MPIC_INTR_TIMER_3 Timer interrupt 3.
OS_MPIC_INTR_TIMER_4 Timer interrupt 4.
OS_MPIC_INTR_TIMER_5 Timer interrupt 5.
OS_MPIC_INTR_TIMER_6 Timer interrupt 6.
OS_MPIC_INTR_TIMER_7 Timer interrupt 7.
OS_MPIC_INTR_MSG_0 Message interrupt 0.
OS_MPIC_INTR_MSG_1 Message interrupt 1.
OS_MPIC_INTR_MSG_2 Message interrupt 2.
OS_MPIC_INTR_MSG_3 Message interrupt 3.
OS_MPIC_INTR_MSG_4 Message interrupt 4.
OS_MPIC_INTR_MSG_5 Message interrupt 5.
OS_MPIC_INTR_MSG_6 Message interrupt 6.
OS_MPIC_INTR_MSG_7 Message interrupt 7.
OS_MPIC_INTR_MSG_8 Message interrupt 8.
OS_MPIC_INTR_MSG_9 Message interrupt 9.
OS_MPIC_INTR_MSG_10 Message interrupt 10.
OS_MPIC_INTR_MSG_11 Message interrupt 11.
OS_MPIC_INTR_MSG_12 Message interrupt 12.
OS_MPIC_INTR_MSG_13 Message interrupt 13.
OS_MPIC_INTR_MSG_14 Message interrupt 14.
OS_MPIC_INTR_MSG_15 Message interrupt 15.
OS_MPIC_INTR_SHARED_MSG_0 Shared message interrupt 0.
OS_MPIC_INTR_SHARED_MSG_1 Shared message interrupt 1.
OS_MPIC_INTR_SHARED_MSG_2 Shared message interrupt 2.
OS_MPIC_INTR_SHARED_MSG_3 Shared message interrupt 3.

OS_MPIC_INTR_SHARED_MSG_4 Shared message interrupt 4.
OS_MPIC_INTR_SHARED_MSG_5 Shared message interrupt 5.
OS_MPIC_INTR_SHARED_MSG_6 Shared message interrupt 6.
OS_MPIC_INTR_SHARED_MSG_7 Shared message interrupt 7.
OS_MPIC_INTR_SHARED_MSG_8 Shared message interrupt 8.
OS_MPIC_INTR_SHARED_MSG_9 Shared message interrupt 9.
OS_MPIC_INTR_SHARED_MSG_10 Shared message interrupt 10.
OS_MPIC_INTR_SHARED_MSG_11 Shared message interrupt 11.
OS_MPIC_INTR_SHARED_MSG_12 Shared message interrupt 12.
OS_MPIC_INTR_SHARED_MSG_13 Shared message interrupt 13.
OS_MPIC_INTR_SHARED_MSG_14 Shared message interrupt 14.
OS_MPIC_INTR_SHARED_MSG_15 Shared message interrupt 15.
OS_MPIC_INTR_SHARED_MSG_16 Shared message interrupt 16.
OS_MPIC_INTR_SHARED_MSG_17 Shared message interrupt 17.
OS_MPIC_INTR_SHARED_MSG_18 Shared message interrupt 18.
OS_MPIC_INTR_SHARED_MSG_19 Shared message interrupt 19.
OS_MPIC_INTR_SHARED_MSG_20 Shared message interrupt 20.
OS_MPIC_INTR_SHARED_MSG_21 Shared message interrupt 21.
OS_MPIC_INTR_SHARED_MSG_22 Shared message interrupt 22.
OS_MPIC_INTR_SHARED_MSG_23 Shared message interrupt 23.
OS_MPIC_INTR_SHARED_MSG_24 Shared message interrupt 24.
OS_MPIC_INTR_SHARED_MSG_25 Shared message interrupt 25.
OS_MPIC_INTR_SHARED_MSG_26 Shared message interrupt 26.
OS_MPIC_INTR_SHARED_MSG_27 Shared message interrupt 27.
OS_MPIC_INTR_SHARED_MSG_28 Shared message interrupt 28.
OS_MPIC_INTR_SHARED_MSG_29 Shared message interrupt 29.
OS_MPIC_INTR_SHARED_MSG_30 Shared message interrupt 30.
OS_MPIC_INTR_SHARED_MSG_31 Shared message interrupt 31.
OS_MPIC_INTR_SHARED_MSG_32 Shared message interrupt 32.
OS_MPIC_INTR_SHARED_MSG_33 Shared message interrupt 33.
OS_MPIC_INTR_SHARED_MSG_34 Shared message interrupt 34.
OS_MPIC_INTR_SHARED_MSG_35 Shared message interrupt 35.
OS_MPIC_INTR_SHARED_MSG_36 Shared message interrupt 36.
OS_MPIC_INTR_SHARED_MSG_37 Shared message interrupt 37.
OS_MPIC_INTR_SHARED_MSG_38 Shared message interrupt 38.
OS_MPIC_INTR_SHARED_MSG_39 Shared message interrupt 39.
OS_MPIC_INTR_SHARED_MSG_40 Shared message interrupt 40.
OS_MPIC_INTR_SHARED_MSG_41 Shared message interrupt 41.
OS_MPIC_INTR_SHARED_MSG_42 Shared message interrupt 42.
OS_MPIC_INTR_SHARED_MSG_43 Shared message interrupt 43.
OS_MPIC_INTR_SHARED_MSG_44 Shared message interrupt 44.
OS_MPIC_INTR_SHARED_MSG_45 Shared message interrupt 45.
OS_MPIC_INTR_SHARED_MSG_46 Shared message interrupt 46.
OS_MPIC_INTR_SHARED_MSG_47 Shared message interrupt 47.
OS_MPIC_INTR_SHARED_MSG_48 Shared message interrupt 48.

OS_MPIC_INTR_SHARED_MSG_49 Shared message interrupt 49.
OS_MPIC_INTR_SHARED_MSG_50 Shared message interrupt 50.
OS_MPIC_INTR_SHARED_MSG_51 Shared message interrupt 51.
OS_MPIC_INTR_SHARED_MSG_52 Shared message interrupt 52.
OS_MPIC_INTR_SHARED_MSG_53 Shared message interrupt 53.
OS_MPIC_INTR_SHARED_MSG_54 Shared message interrupt 54.
OS_MPIC_INTR_SHARED_MSG_55 Shared message interrupt 55.
OS_MPIC_INTR_SHARED_MSG_56 Shared message interrupt 56.
OS_MPIC_INTR_SHARED_MSG_57 Shared message interrupt 57.
OS_MPIC_INTR_SHARED_MSG_58 Shared message interrupt 58.
OS_MPIC_INTR_SHARED_MSG_59 Shared message interrupt 59.
OS_MPIC_INTR_SHARED_MSG_60 Shared message interrupt 60.
OS_MPIC_INTR_SHARED_MSG_61 Shared message interrupt 61.
OS_MPIC_INTR_SHARED_MSG_62 Shared message interrupt 62.
OS_MPIC_INTR_SHARED_MSG_63 Shared message interrupt 63.
OS_MPIC_INTR_IPI_0 Inter-processor interrupt 0.
OS_MPIC_INTR_IPI_1 Inter-processor interrupt 1.
OS_MPIC_INTR_IPI_2 Inter-processor interrupt 2.
OS_MPIC_INTR_IPI_3 Inter-processor interrupt 3.

2.14.3 MPIC runtime API

2.14.3.1 Overview

MPIC runtime API.

Data Structures

- struct os_mpic_intr_params_t
- struct os_mpic_isr_entry_t

Macros

- #define OS_VPR_PRIORITY_SET(priority) priority << VPR_PRIORITY_SHIFT
- #define OS_MPIC_DEFAULT_PRIORITY 1
- #define OS_MPIC_DEFAULT_POLARITY OS_MPIC_INTR_POLARITY_LOW
- #define OS_MPIC_DEFAULT_SENSE OS_MPIC_INTR_SENSE_EDGE
- #define OS_MPIC_DEFAULT_TARGET OS_MPIC_INTR_TARGET_DEFAULT
- #define osMpicSetIntr(mpic_intr_id, isr, arg) osMpicSetConfigIntr(mpic_intr_id,isr,arg,NULL)

Enumerations

- enum os_mpic_intr_target_t {
  OS_MPIC_INTR_TARGET_DEFAULT, OS_MPIC_INTR_TARGET_CRITICAL, OS_MPIC_↩

INTR_TARGET_IRQ_OUT,
OS_MPIC_INTR_TARGET_MACHINE_CHECK, OS_MPIC_INTR_TARGET_SOC_INTR_0,
OS_MPIC_INTR_TARGET_SOC_INTR_1,
OS_MPIC_INTR_TARGET_SOC_INTR_2 }

- enum os_mpic_intr_sense_t { OS_MPIC_INTR_SENSE_EDGE, OS_MPIC_INTR_SENSE_LEVEL }
- enum os_mpic_intr_polarity_t { OS_MPIC_INTR_POLARITY_LOW, OS_MPIC_INTR_POLARITY_HIGH }

Functions

- os_status osMpicSetConfigIntr (uint32_t mpic_intr_id, os_mpic_isr isr, os_mpic_arg arg, os_mpic_intr_params_t *intr_params)
- os_status osMpicWriteMessage (uint32_t mpic_intr_id, uint32_t msg_data, bool blocking)
- bool osMpicReadMessage (uint32_t mpic_intr_id, uint32_t *msg_data)
- void osMpicReset ()
- os_status osMpicEnableIntr (uint32_t mpic_intr_id)
- os_status osMpicDisableIntr (uint32_t mpic_intr_id)
- os_status osMpicWriteMsi (uint32_t mpic_intr_id, uint8_t signal)
- os_status osMpicMsiCoalescingConfig (uint32_t mpic_intr_id, uint32_t coalescing_mask)
- os_status osMpicReadMsiSignalRegister (uint32_t mpic_intr_id, uint32_t *msir)
- os_status osMpicFreeIntr (uint32_t mpic_intr_id)
- os_status osMpicFindMsiIntr (uint32_t *mpic_intr_id)

2.14.3.2 Data Structure Documentation

2.14.3.2.1 struct os_mpic_intr_params_t

MPIC Interrupt Configuration Parameters.

Data Fields

- uint8_t priority
- os_mpic_intr_target_t target
- os_mpic_intr_sense_t sense
- os_mpic_intr_polarity_t polarity

2.14.3.2.1.1 Field Documentation

2.14.3.2.1.1.1 uint8_t os_mpic_intr_params_t::priority

Interrupt priority: 0 disabled, 1 lowest, ...

15 highest

2.14.3.2.1.1.2 os_mpic_intr_target_t os_mpic_intr_params_t::target

Interrupt target selection.

2.14.3.2.1.1.3 os_mpic_intr_sense_t os_mpic_intr_params_t::sense

Interrupt sense (external interrupts only)

2.14.3.2.1.1.4 os_mpic_intr_polarity_t os_mpic_intr_params_t::polarity

Interrupt polarity (internal/external interrupts only)

2.14.3.2.2 struct os_mpic_isr_entry_t

Interrupt table definition.

Data Fields

- os_mpic_isr isr
- os_mpic_arg arg

2.14.3.2.2.1 Field Documentation

2.14.3.2.2.1.1 os_mpic_isr os_mpic_isr_entry_t::isr

Interrupt service routine (ISR)

2.14.3.2.2.1.2 os_mpic_arg os_mpic_isr_entry_t::arg

Argument for the ISR.

2.14.3.3 Macro Definition Documentation

2.14.3.3.1 #define OS_VPR_PRIORITY_SET(*priority*) priority << VPR_PRIORITY_SHIFT

MPIC default interrupt parameters (internal/external interrupts only)

use to set priority in vpr register

2.14.3.3.2 #define OS_MPIC_DEFAULT_PRIORITY 1

Default interrupt priority - lowest priority 1.

2.14.3.3.3 #define OS_MPIC_DEFAULT_POLARITY OS_MPIC_INTR_POLARITY_LOW

Default interrupt polarity - active-low.

2.14.3.3.4 #define OS_MPIC_DEFAULT_SENSE OS_MPIC_INTR_SENSE_EDGE

Default interrupt sense edge-sensitive.

2.14.3.3.5 #define OS_MPIC_DEFAULT_TARGET OS_MPIC_INTR_TARGET_DEFAULT

Default interrupt line (core int)

2.14.3.3.6 #define osMpicSetIntr(*mpic_intr_id, isr, arg*) osMpicSetConfigIntr(mpic_intr_↩ id,isr,arg,NULL)

Function osMpicSetIntr - sets interrupt with default parameters.

2.14.3.4 Enumeration Type Documentation

2.14.3.4.1 enum os_mpic_intr_target_t

MPIC interrupt target selection.

Enumerator

OS_MPIC_INTR_TARGET_DEFAULT Default interrupt line (core int)
OS_MPIC_INTR_TARGET_CRITICAL Critical interrupt line (core cint)
OS_MPIC_INTR_TARGET_IRQ_OUT IRQ_OUT signal for external processing.
OS_MPIC_INTR_TARGET_MACHINE_CHECK Machine check interrupt line (core mcp)
OS_MPIC_INTR_TARGET_SOC_INTR_0 SoC-specific interrupt routing (option 0)
OS_MPIC_INTR_TARGET_SOC_INTR_1 SoC-specific interrupt routing (option 1)
OS_MPIC_INTR_TARGET_SOC_INTR_2 SoC-specific interrupt routing (option 2)

2.14.3.4.2 enum os_mpic_intr_sense_t

MPIC interrupt sense modes (external interrupts only)

Enumerator

OS_MPIC_INTR_SENSE_EDGE Interrupt is edge-sensitive.
OS_MPIC_INTR_SENSE_LEVEL Interrupt is level-sensitive.

2.14.3.4.3 enum os_mpic_intr_polarity_t

MPIC interrupt polarity modes (internal/external interrupts only)

Enumerator

OS_MPIC_INTR_POLARITY_LOW For edge-sensitive interrupts: high-to-low triggered; For level-sensitive interrupts: active-low.

OS_MPIC_INTR_POLARITY_HIGH For edge-sensitive interrupts: low-to-high triggered; For level-sensitive interrupts: active-high.

2.14.3.5 Function Documentation

2.14.3.5.1 os_status osMpicSetConfigIntr (uint32_t *mpic_intr_id,* os_mpic_isr *isr,* os_mpic_arg *arg,* os_mpic_intr_params_t * *intr_params*)

Registration of an interrupt source.

Parameters

in	*mpic_intr_id*	- Interrupt ID as defined by os_mpic_intr_id.
in	*isr*	- Callback routine to be called when interrupt occurs;
in	*arg*	- This argument will be passed in the isr callback.
in	*intr_params*	- This argument contain the requested interrupt parameters.

Returns

OS_SUCCESS on success; Error code otherwise.

2.14.3.5.2 os_status osMpicWriteMessage (uint32_t *mpic_intr_id,* uint32_t *msg_data,* bool *blocking*)

Write one of the message registers to issue a message interrupt to the assigned core.

The message interrupts can be used as a doorbell mechanism between cores. The user writes a 32-bit data to one of the message registers. When this value is written, the core assigned to that interrupt is interrupted and may read the message data using the osMpicReadMessage() routine. Cores are assigned to message interrupts as with any other interrupt, using the osMpicSetIntr() routine.

Parameters

in	*mpic_intr_id*	- Interrupt ID as defined by os_mpic_intr_id.

in	*msg_data*	- The message data to write (up to 32-bits).
in	*blocking*	- FALSE to return after the message is written; TRUE to block until the message is read; Note that blocking mode has no timeout, and it is not recommended to use blocking mode when running in interrupt context.

Returns

OS_SUCCESS on success; Error code otherwise.

2.14.3.5.3 bool osMpicReadMessage (uint32_t *mpic_intr_id,* uint32_t * *msg_data*)

Read one of the message registers to retrieve its 32-bit data.

The message interrupts can be used as a doorbell mechanism between cores. The user writes a 32-bit data to one of the message registers via osMpicWriteMessage() routine. When this value is written, the core assigned to that interrupt is interrupted and may read the message data using this routine. Cores are assigned to message interrupts as with any other interrupt, using the osMpicSetIntr() routine. This routine may be called by the message interrupt handler, or even in polling mode (without registering for the interrupt). In polling mode, the return value indicates whether a message was pending (in which case the returned message data is valid).

Parameters

in	*mpic_intr_id*	- Interrupt ID as defined by os_mpic_intr_id.
out	*msg_data*	- Returns the message data (up to 32-bits) only if a message was pending.

Returns

TRUE if a message was pending on the given message ID; FALSE otherwise.

2.14.3.5.4 void osMpicReset ( )

Reset MPIC.

Warning

Needs to be used only by the MPIC owner.

Returns

None.

2.14.3.5.5 os_status osMpicEnableIntr (uint32_t *mpic_intr_id*)

Unmask (enable) a specific interrupt source.

Parameters

in	*mpic_intr_id*	- Interrupt ID as defined by os_mpic_intr_id.

Returns

OS_SUCCESS on success; Error code otherwise.

2.14.3.5.6 os_status osMpicDisableIntr (uint32_t *mpic_intr_id*)

Mask (disable) a specific interrupt source.

Parameters

in	*mpic_intr_id*	- Interrupt ID as defined by os_mpic_intr_id.

Returns

OS_SUCCESS on success; Error code otherwise.

2.14.3.5.7 os_status osMpicWriteMsi (uint32_t *mpic_intr_id,* uint8_t *signal*)

Invoke an interrupt to the assigned signal.

Parameters

in	*mpic_intr_id*	- Interrupt ID as defined by os_mpic_intr_id.
out	*signal*	- The assigned signal (valid values: 0-31).

Returns

OS_SUCCESS on success; Error code otherwise.

2.14.3.5.8 os_status osMpicMsiCoalescingConfig (uint32_t *mpic_intr_id,* uint32_t *coalescing_mask*)

Configure the coalescing settings for one of the MSI.

Parameters

in	*mpic_intr_id*	- Interrupt ID as defined by os_mpic_intr_id.
out	*coalescing_mask*	- The signals included in coalescing. (up to 32-bits).

Returns

OS_SUCCESS on success; Error code otherwise.

2.14.3.5.9 os_status osMpicReadMsiSignalRegister (uint32_t *mpic_intr_id,* uint32_t * *msir*)

Reads the signals from the MSI message registe.

Parameters

in	*mpic_intr_id*	- Interrupt ID as defined by os_mpic_intr_id.
out	*msir*	- Pointer to the signals register.

Returns

OS_SUCCESS on success; Error code otherwise.

2.14.3.5.10 os_status osMpicFreeIntr (uint32_t *mpic_intr_id*)

Deregistration of an interrupt source.

Parameters

in	*mpic_intr_id*	- Interrupt ID as defined by os_mpic_intr_id.

Returns

OS_SUCCESS on success; Error code otherwise.

2.14.3.5.11 os_status osMpicFindMsiIntr (uint32_t * *mpic_intr_id*)

Returns the interrupt id of an unused MSI interrupt.

Parameters

in	*mpic_intr_id*	- Pointer to hold the free interrupt ID as defined by os_mpic_intr_id.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

This function is not multi-core safe!

Chapter 3
Drivers

3.1 Overview

Modules

- CPRI Module API
- sRIO Module API
- I2C Module API
- MAPLE-B3 Module API
- OCeaN DMA Module API
- DPAA Module API

3.2 CPRI Module API

3.2.1 Overview

```
CPRI runtime API


CPRI initialize.


CPRI block API
```

Modules

- CPRI initialization
- CPRI runtime
- CPRI B4860 architecture-specific definitions

3.2.2 CPRI initialization

3.2.2.1 Overview

```
CPRI initialization API.
```

Data Structures

- struct cpri_multicast_params_t
- struct mapping_table_entry_t
- struct mapping_params_t
- struct cpri_iq_init_params_t
- struct cpri_iq_additional_params_t

- struct cpri_vss_init_params_t
- struct cpri_ethernet_init_params_t
- struct cpri_hdlc_init_params_t
- struct cpri_init_params_t
- struct cpri_auxiliary_params_t

Macros

- #define CPRI_IQ_DEV_NAME0 "c_i0"
- #define CPRI_IQ_DEV_NAME1 "c_i1"
- #define CPRI_IQ_DEV_NAME2 "c_i2"
- #define CPRI_IQ_DEV_NAME3 "c_i3"
- #define CPRI_IQ_DEV_NAME4 "c_i4"
- #define CPRI_IQ_DEV_NAME5 "c_i5"
- #define CPRI_VSS_DEV_NAME0 "c_v0"
- #define CPRI_VSS_DEV_NAME1 "c_v1"
- #define CPRI_VSS_DEV_NAME2 "c_v2"
- #define CPRI_VSS_DEV_NAME3 "c_v3"
- #define CPRI_VSS_DEV_NAME4 "c_v4"
- #define CPRI_VSS_DEV_NAME5 "c_v5"
- #define CPRI_ETHERNET_DEV_NAME0 "c_e0"
- #define CPRI_ETHERNET_DEV_NAME1 "c_e1"
- #define CPRI_ETHERNET_DEV_NAME2 "c_e2"
- #define CPRI_ETHERNET_DEV_NAME3 "c_e3"
- #define CPRI_ETHERNET_DEV_NAME4 "c_e4"
- #define CPRI_ETHERNET_DEV_NAME5 "c_e5"
- #define CPRI_HDLC_DEV_NAME0 "c_h0"
- #define CPRI_HDLC_DEV_NAME1 "c_h1"
- #define CPRI_HDLC_DEV_NAME2 "c_h2"
- #define CPRI_HDLC_DEV_NAME3 "c_h3"
- #define CPRI_HDLC_DEV_NAME4 "c_h4"
- #define CPRI_HDLC_DEV_NAME5 "c_h5"
- #define CPRI_IQ_ENABLED &cpriIqInitialize
- #define CPRI_VSS_ENABLED &cpriVssInitialize
- #define CPRI_ETHERNET_ENABLED &cpriEthernetInitialize
- #define CPRI_HDLC_ENABLED &cpriHdlcInitialize
- #define CPRI_IQ_DISABLED NULL
- #define CPRI_VSS_DISABLED NULL
- #define CPRI_ETHERNET_DISABLED NULL
- #define CPRI_HDLC_DISABLED NULL
- #define CPRI_NOT_ACTIVE 0
- #define CPRI_ACTIVE 1
- #define CPRI_NOT_SYNCHONIZED 0
- #define CPRI_SYNCHONIZED 1
- #define MAXIMUM_NUMBER_OF_AXCS 24
- #define CPRI_ALL_ERRORS

Enumerations

- enum cpri_loop_mode_t
- enum cpri_sync_mode_t
- enum ext_sync_act_t
- enum shared_sync_mode_source_t
- enum reset_ack_assert_length_t

- enum reset_detection_length_t
- enum cpri_protocol_t
- enum cpri_multicast_mode_t
- enum mapping_mode_t
- enum sampling_width_select_t
- enum iq_bus_transaction_size_t
- enum vss_bus_transaction_size_t
- enum oversampling_ratio_t
- enum cpri_interrupt_type_t

Functions

- os_status cpriInitialize (cpri_global_init_params_t *global_parms, cpri_init_params_t(*init_params)[])
- os_status cpriIqInitialize (cpri_num_t cpri_num, cpri_iq_init_params_t *init_params, cpri_iq_additional_params_t *additional_init_params)
- os_status cpriVssInitialize (cpri_num_t cpri_num, cpri_vss_init_params_t *init_params, void **vss_handle)
- os_status cpriEthernetInitialize (cpri_num_t cpri_num, cpri_ethernet_init_params_t *init_params, void **ethernet_handle)
- os_status cpriHdlcInitialize (cpri_num_t cpri_num, cpri_hdlc_init_params_t *init_params, void **hdlc_handle)

CPRI active flags for RCR and TCR registers.

- #define CPRI_IQ_ACTIVE 0x00000001
- #define CPRI_ETHERNET_ACTIVE 0x00000002
- #define CPRI_HDLC_ACTIVE 0x00000004
- #define CPRI_VSS_ACTIVE 0x00000008
- #define CPRI_ALL_DATA_TYPES_ACTIVE (CPRI_IQ_ACTIVE | CPRI_ETHERNET_ACTIVE | CPRI_HDLC_ACTIVE | CPRI_VSS_ACTIVE)

Defines of the cpri device control commands

- #define CPRI_DEVICE_DELAYS_CALCULATE 0x00010000
- #define CPRI_DEVICE_STATISTICS_SET 0x00020000
- #define CPRI_DEVICE_TX_CONTROL_TABLE_WRITE 0x00040000
- #define CPRI_DEVICE_TX_CONTROL_TABLE_READ 0x00080000
- #define CPRI_DEVICE_RX_CONTROL_TABLE_READ 0x00100000
- #define CPRI_DEVICE_RX_BFN_COUNTER_READ 0x00200000
- #define CPRI_DEVICE_RX_HFN_COUNTER_READ 0x00400000
- #define CPRI_DEVICE_TX_BFN_COUNTER_READ 0x00800000
- #define CPRI_DEVICE_TX_HFN_COUNTER_READ 0x01000000
- #define CPRI_DEVICE_RESET_REQUEST 0x02000000
- #define CPRI_DEVICE_RESET_REQUEST_DISABLE 0x04000000
- #define CPRI_DEVICE_RESET_ENABLE 0x08000000
- #define CPRI_DEVICE_RESET_DISABLE 0x10000000
- #define CPRI_DEVICE_CHECK_RESET_DETECTED 0x20000000
- #define CPRI_DEVICE_TX_BFN_RESET 0x40000000
- #define CPRI_DEVICE_DMA_RESTART 0x80000000
- #define CPRI_DEVICE_RECONFIGURATION_LEVEL0 0x80010000

- #define CPRI_DEVICE_RECONFIGURATION_LEVEL1 0x80020000
- #define CPRI_DEVICE_RECONFIGURATION_LEVEL2 0x80040000
- #define CPRI_STATUS_QUERY 0x80080000

defines of CPRI AXC (channels) activation

- #define CPRI_AXC_0_ACTIVE 0x00000001
- #define CPRI_AXC_1_ACTIVE 0x00000002
- #define CPRI_AXC_2_ACTIVE 0x00000004
- #define CPRI_AXC_3_ACTIVE 0x00000008
- #define CPRI_AXC_4_ACTIVE 0x00000010
- #define CPRI_AXC_5_ACTIVE 0x00000020
- #define CPRI_AXC_6_ACTIVE 0x00000040
- #define CPRI_AXC_7_ACTIVE 0x00000080
- #define CPRI_AXC_8_ACTIVE 0x00000100
- #define CPRI_AXC_9_ACTIVE 0x00000200
- #define CPRI_AXC_10_ACTIVE 0x00000400
- #define CPRI_AXC_11_ACTIVE 0x00000800
- #define CPRI_AXC_12_ACTIVE 0x00001000
- #define CPRI_AXC_13_ACTIVE 0x00002000
- #define CPRI_AXC_14_ACTIVE 0x00004000
- #define CPRI_AXC_15_ACTIVE 0x00008000
- #define CPRI_AXC_16_ACTIVE 0x00010000
- #define CPRI_AXC_17_ACTIVE 0x00020000
- #define CPRI_AXC_18_ACTIVE 0x00040000
- #define CPRI_AXC_19_ACTIVE 0x00080000
- #define CPRI_AXC_20_ACTIVE 0x00100000
- #define CPRI_AXC_21_ACTIVE 0x00200000
- #define CPRI_AXC_22_ACTIVE 0x00400000
- #define CPRI_AXC_23_ACTIVE 0x00800000

cpri interrupts events - values according to ICR and RER/TER registers

CPRI interrupt definition data structures.

This structure should be used to define the interrutps used for CPRI and the events represented by each interrupt .

- #define CPRI_IQ_THRESHOLD_EVENT 0x00000001
- #define CPRI_IQ_FIRST_THRESHOLD_EVENT 0x00000002
- #define CPRI_IQ_SECOND_THRESHOLD_EVENT 0x00000004
- #define CPRI_VSS_EVENT 0x00000008
- #define CPRI_HDLC_EVENT 0x00000010
- #define CPRI_ETHERNET_EVENT 0x00000020
- #define CPRI_HFN_TIMING_EVENT 0x00000040
- #define CPRI_BFN_TIMING_EVENT 0x00000080
- #define CPRI_MULTICAST_RX_IQ_THRESHOLD_EVENT 0x00000100
- #define CPRI_IQ_TX_BEGIN_EVENT 0x00000100
- #define CPRI_ALL_TER_RER_EVENTS

cpri error events

- #define CPRI_RECEIVE_IQ_OVERRUN_ERROR 0x00000001
- #define CPRI_TRANSMIT_IQ_UNDERRUN_ERROR 0x00000002
- #define CPRI_RECEIVE_ETH_MEMORY_OVERRUN_ERROR 0x00000004
- #define CPRI_TRANSMIT_ETH_UNDERRUN_ERROR 0x00000008
- #define CPRI_RECEIVE_ETH_BUFFER_DESCRIPTOR_UNDERRUN_ERROR 0x00000010
- #define CPRI_RECEIVE_HDLC_OVERRUN_ERROR 0x00000020
- #define CPRI_TRANSMIT_HDLC_UNDERRUN_ERROR 0x00000040
- #define CPRI_RECEIVE_HDLC_BUFFER_DESCRIPTOR_UNDERRUN_ERROR 0x00000080
- #define CPRI_RECEIVE_VSS_OVERRUN_ERROR 0x00000100
- #define CPRI_TRANSMIT_VSS_UNDERRUN_ERROR 0x00000200
- #define CPRI_EXTERNAL_SYNC_SYNCHRONIZATION_LOST_ERROR 0x00008000
- #define CPRI_REMOTE_LOS_ERROR 0x00010000
- #define CPRI_REMOTE_LOF_ERROR 0x00020000
- #define CPRI_REMOTE_RAI_ERROR 0x00040000
- #define CPRI_REMOTE_SDI_ERROR 0x00080000
- #define CPRI_LOCAL_LOS_ERROR 0x00100000
- #define CPRI_LOCAL_LOST_OF_FRAME_ERROR 0x00200000
- #define CPRI_FREQUENCY_ALARM_ERROR 0x00800000
- #define CPRI_REMOTE_RESET_ERROR 0x00400000
- #define CPRI_ECC_CONFIGURATION_MEMORY_ERROR 0x00000400
- #define CPRI_ECC_DATA_MEMORY_ERROR 0x00000800
- #define CPRI_REMOTE_RESET_ACKNOWLEDGE 0x01000000

CPRI ECC multibit errors

- #define CPRI_ERROR_MULTIBIT_ECC_SECOND_DESTINATION_DMA_MEMORY 0↩X00000001
- #define CPRI_ERROR_MULTIBIT_ECC_COMBINER_MEMORY 0X00000002
- #define CPRI_ERROR_MULTIBIT_ECC_FRAME_ELASTIC_BUFFER_MEMORY 0X00000004
- #define CPRI_ERROR_MULTIBIT_ECC_AUX_MEMORY 0X00000008
- #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_TRANSMIT_HDLC_MEMORY 0↩X00000010
- #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_RECEIVE_HDLC_MEMORY 0X00000020
- #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_TRANSMIT_ETH_MEMORY 0X00000040
- #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_RECEIVE_ETH_MEMORY 0X00000080
- #define CPRI_ERROR_MULTIBIT_ECC_TRANSMIT_CONTROL_WORD_TABLE_MEMO↩RY 0X00000100
- #define CPRI_ERROR_MULTIBIT_ECC_RECEIVE_CONTROL_WORD_TABLE_MEMOR↩Y 0X00000200
- #define CPRI_ERROR_MULTIBIT_ECC_RECEIVE_DMA_MEMORY 0X00000800
- #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_TRANSMIT_IQ_MEMORY 0X00001000
- #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_RECEIVE_IQ_MEMORY 0X00002000
- #define CPRI_ERROR_MULTIBIT_ECC_TRANSMIT_CONFIGURATION_MEMORY 0↩X00004000
- #define CPRI_ERROR_MULTIBIT_ECC_RECEIVE_CONFIGURATION_MEMORY 0↩X00008000

3.2.2.2 Data Structure Documentation

3.2.2.2.1 struct cpri_multicast_params_t

CPRI multicast mode parameters.

This structure should be used as the CPRI LLD parameters for using multicast modes with CPRI .

Data Fields

- cpri_multicast_mode_t multicast_mode
- iq_bus_transaction_size_t unicast_receive_transaction_size
- uint32_t rx_buffer_size
- uint32_t threshold

3.2.2.2.1.1 Field Documentation

3.2.2.2.1.1.1 cpri_multicast_mode_t cpri_multicast_params_t::multicast_mode

multicast mode

3.2.2.2.1.1.2 iq_bus_transaction_size_t cpri_multicast_params_t::unicast_receive_transaction_size

only for uni multicast - uni_multicast mode transaction size

3.2.2.2.1.1.3 uint32_t cpri_multicast_params_t::rx_buffer_size

only for multicast - size of rx buffer for second destination

3.2.2.2.1.1.4 uint32_t cpri_multicast_params_t::threshold

the amount of received bytes for a IQ tx event

3.2.2.2.2 struct mapping_table_entry_t

CPRI advanced mapping mode parameters.

This structure should be used as the CPRI LLD parameters for using advanced mapping modes with CPRI .

Data Fields

- uint32_t width0:5
- uint32_t position0:5
- uint32_t axc0:5
- uint32_t enable0:1
- uint32_t width1:5
- uint32_t position1:5

- uint32_t axc1:5
- uint32_t enable1:1

3.2.2.2.2.1 Field Documentation

3.2.2.2.2.1.1 uint32_t mapping_table_entry_t::width0

30 or 32 bit sample.

3.2.2.2.2.1.2 uint32_t mapping_table_entry_t::position0

I/Q Sample Start Bit Position.

3.2.2.2.2.1.3 uint32_t mapping_table_entry_t::axc0

AxC Number.

3.2.2.2.2.1.4 uint32_t mapping_table_entry_t::enable0

Mapping of I/Q Sample Into Timeslot Enable.

3.2.2.2.2.1.5 uint32_t mapping_table_entry_t::width1

30 or 32 bit sample.

3.2.2.2.2.1.6 uint32_t mapping_table_entry_t::position1

I/Q Sample Start Bit Position.

3.2.2.2.2.1.7 uint32_t mapping_table_entry_t::axc1

AxC Number.

3.2.2.2.2.1.8 uint32_t mapping_table_entry_t::enable1

Mapping of I/Q Sample Into Timeslot Enable.

3.2.2.2.3 struct mapping_params_t

CPRI advanced mapping mode parameters.

This structure should be used as a mapping entry to the CPRI LLD parameters for using advanced mapping modes with CPRI .

Data Fields

- mapping_mode_t mapping_mode
- oversampling_ratio_t axc_oversampling_factor
- uint32_t mapping_table_configuration:7
- mapping_table_entry_t * rx_mapping_table

- mapping_table_entry_t * tx_mapping_table
- uint32_t tx_table_size
- uint32_t rx_table_size

3.2.2.2.3.1 Field Documentation

3.2.2.2.3.1.1 mapping_mode_t mapping_params_t::mapping_mode

Mapping mode used on the link.

3.2.2.2.3.1.2 oversampling_ratio_t mapping_params_t::axc_oversampling_factor

Only for basic mapping - oversampling factor for AxC's on the link.

3.2.2.2.3.1.3 uint32_t mapping_params_t::mapping_table_configuration

Only for advanced mapping - Number of Basic Frames in AxC Container Block.

3.2.2.2.3.1.4 mapping_table_entry_t* mapping_params_t::rx_mapping_table

Only for advanced mapping - pointer to beginning of the rx mapping table array - length up to 1920.

3.2.2.2.3.1.5 mapping_table_entry_t* mapping_params_t::tx_mapping_table

Only for advanced mapping - pointer to beginning of the tx mapping table array - length up to 1920.

3.2.2.2.3.1.6 uint32_t mapping_params_t::tx_table_size

Only for advanced mapping - length of tx table up to 960.

3.2.2.2.3.1.7 uint32_t mapping_params_t::rx_table_size

Only for advanced mapping - length of rx table up to 960.

3.2.2.2.4 struct cpri_iq_init_params_t

Initialization structure of CPRI IQ data.

This structure can be used for 2 purposes: 1 - Given as CPRI IQ initialization parameters in cpri_init_↩ params_t. 2 - Parameters for calling CPRI level2 and level3 reconfiguration. The following parameters must not be changed compared to existing configuration: a. Cores configuration (participating_cores and initializing_core) b. Sync (transmit_iq_sync and receive_iq_sync) 3 - Parameters for calling CPRI level0 reconfiguration. In that case only cores configuration must equal to existing configuration.

NOTE: for reconfiguration usage, participating_cores mask must contain only a single core.

Data Fields

- uint8_t tx_axc_number
- uint8_t rx_axc_number

- uint8_t participating_cores
- cpri_cores_t initializing_core
- uint32_t axc_tx_active
- uint32_t axc_rx_active
- uint32_t axc_tx_summing_active
- uint32_t axc_rx_forwarding_active
- uint32_t axc_tx_active_dma
- uint32_t axc_rx_active_dma
- uint32_t auxiliary_mode:1
- uint32_t transmit_sync_active:1
- uint32_t transmit_iq_sync:1
- uint32_t receive_iq_sync:1
- uint32_t map_tx_hf_resync:1
- uint32_t map_rx_hf_resync:1
- uint32_t start_tx_hf_resync:1
- uint32_t tx_shared_mode:1
- uint32_t dual_bandwidth_mode:1
- mapping_params_t * mapping_params
- cpri_multicast_params_t * multicast_params
- sampling_width_select_t sample_width
- iq_bus_transaction_size_t tx_transaction_size
- iq_bus_transaction_size_t rx_transaction_size
- uint32_t rx_buffer_size
- uint32_t tx_buffer_size
- uint32_t tx_first_threshold
- uint32_t tx_second_threshold
- uint32_t rx_first_threshold
- uint32_t rx_second_threshold
- uint32_t tx_threshold
- uint32_t rx_threshold
- uint8_t map_tx_offset_z
- uint8_t map_tx_offset_x
- uint8_t map_rx_offset_z
- uint8_t map_rx_offset_x
- uint8_t start_tx_offset_z
- uint8_t start_tx_offset_x
- uint32_t * auxiliary_mask
- uint32_t auxiliary_mask_size
- uint32_t rx_symbol_awareness_int:1
- uint32_t rx_2nd_dest_symbol_awareness_int:1
- uint32_t tx_symbol_awareness_int:1
- uint32_t warm_reset_enable
- uint32_t transmit_iq_sample_count:22

3.2.2.2.4.1 Field Documentation

3.2.2.2.4.1.1 uint8_t cpri_iq_init_params_t::tx_axc_number

number of tx AxC's passing on the link - 1 to 24

3.2.2.2.4.1.2 uint8_t cpri_iq_init_params_t::rx_axc_number

number of rx AxC's passing on the link - 1 to 24

3.2.2.2.4.1.3 uint8_t cpri_iq_init_params_t::participating_cores

cores that will use IQ data type

3.2.2.2.4.1.4 cpri_cores_t cpri_iq_init_params_t::initializing_core

core that will initialize the CPRI block

3.2.2.2.4.1.5 uint32_t cpri_iq_init_params_t::axc_tx_active

active tx AxC channels

3.2.2.2.4.1.6 uint32_t cpri_iq_init_params_t::axc_rx_active

active rx AxC channels

3.2.2.2.4.1.7 uint32_t cpri_iq_init_params_t::axc_tx_summing_active

AxC channels for summing.

3.2.2.2.4.1.8 uint32_t cpri_iq_init_params_t::axc_rx_forwarding_active

Forwarded AxC channels.

3.2.2.2.4.1.9 uint32_t cpri_iq_init_params_t::axc_tx_active_dma

Only if dual bandwidth mode is active - active tx AxC channels of the DMA, must be a subset of axc_tx↩ _active.

3.2.2.2.4.1.10 uint32_t cpri_iq_init_params_t::axc_rx_active_dma

Only if dual bandwidth mode is active - active rx AxC channels of the DMA, must be a subset of axc_↩ rx_active.

3.2.2.2.4.1.11 uint32_t cpri_iq_init_params_t::auxiliary_mode

auxiliary mode active

3.2.2.2.4.1.12 uint32_t cpri_iq_init_params_t::transmit_sync_active

determines if syncronization is required (to the 10ms frame)

3.2.2.2.4.1.13 uint32_t cpri_iq_init_params_t::transmit_iq_sync

determines if the transfer of transmit IQ data should be synchronized to other CPRI modules in the group.

3.2.2.2.4.1.14 uint32_t cpri_iq_init_params_t::receive_iq_sync

determines if the received IQ data should be synchronized to the other CPRIs in the group.

3.2.2.2.4.1.15 uint32_t cpri_iq_init_params_t::map_tx_hf_resync

TX Resynchronization Every Hyper Frame Enable.

3.2.2.2.4.1.16 uint32_t cpri_iq_init_params_t::map_rx_hf_resync

RX Resynchronization Every Hyper Frame Enable.

3.2.2.2.4.1.17 uint32_t cpri_iq_init_params_t::start_tx_hf_resync

enables synchronization every Hyper Frame nstead of every Radio Frame

3.2.2.2.4.1.18 uint32_t cpri_iq_init_params_t::tx_shared_mode

Selects if the CPRI pair operates in tx shared mode.

3.2.2.2.4.1.19 uint32_t cpri_iq_init_params_t::dual_bandwidth_mode

Double Bandwidth Mode active.

3.2.2.2.4.1.20 mapping_params_t* cpri_iq_init_params_t::mapping_params

parameters for the mapping used by cpri

3.2.2.2.4.1.21 cpri_multicast_params_t* cpri_iq_init_params_t::multicast_params

parameters multicast modes cpri

3.2.2.2.4.1.22 sampling_width_select_t cpri_iq_init_params_t::sample_width

15, 16 or 8 bits of i/q sample width

3.2.2.2.4.1.23 iq_bus_transaction_size_t cpri_iq_init_params_t::tx_transaction_size

size of iq bus tx transaction

3.2.2.2.4.1.24 iq_bus_transaction_size_t cpri_iq_init_params_t::rx_transaction_size

size of iq bus rx transaction

3.2.2.2.4.1.25 uint32_t cpri_iq_init_params_t::rx_buffer_size

size of rx buffer

3.2.2.2.4.1.26 uint32_t cpri_iq_init_params_t::tx_buffer_size

size of tx buffer

3.2.2.2.4.1.27 uint32_t cpri_iq_init_params_t::tx_first_threshold

the location of the first threshold in the AxC tx data buffers

3.2.2.2.4.1.28 uint32_t cpri_iq_init_params_t::tx_second_threshold

the location of the second threshold in the AxC tx data buffers

3.2.2.2.4.1.29 uint32_t cpri_iq_init_params_t::rx_first_threshold

the location of the first threshold in the AxC rx data buffers

3.2.2.2.4.1.30 uint32_t cpri_iq_init_params_t::rx_second_threshold

the location of the second threshold in the AxC rx data buffers

3.2.2.2.4.1.31 uint32_t cpri_iq_init_params_t::tx_threshold

the amount of transmitted bytes for a IQ tx event

3.2.2.2.4.1.32 uint32_t cpri_iq_init_params_t::rx_threshold

the amount of received bytes for a IQ tx event

3.2.2.2.4.1.33 uint8_t cpri_iq_init_params_t::map_tx_offset_z

hyper frame number for the start of TX side AxC Container Block

3.2.2.2.4.1.34 uint8_t cpri_iq_init_params_t::map_tx_offset_x

basic frame number for the start of TX side AxC Container Block

3.2.2.2.4.1.35 uint8_t cpri_iq_init_params_t::map_rx_offset_z

hyper frame number for the start of RX side AxC Container Block

3.2.2.2.4.1.36 uint8_t cpri_iq_init_params_t::map_rx_offset_x

basic frame number for the start of RX side AxC Container Block

3.2.2.2.4.1.37 uint8_t cpri_iq_init_params_t::start_tx_offset_z

Stores the hyper frame number for start of CPRI_TX_START synchronization output.

3.2.2.2.4.1.38 uint8_t cpri_iq_init_params_t::start_tx_offset_x

Stores the basic frame number for start of CPRI_TX_START synchronization output.

3.2.2.2.4.1.39 uint32_t* cpri_iq_init_params_t::auxiliary_mask

only if auxiliary_mode is active - pointer to the beginning of the mask registers array

3.2.2.2.4.1.40 uint32_t cpri_iq_init_params_t::auxiliary_mask_size

only if auxiliary_mode is active - mask registers array size

3.2.2.2.4.1.41 uint32_t cpri_iq_init_params_t::rx_symbol_awareness_int

Generate Receive Symbol Awareness interrupt (expected usage: toward DSP timer)

3.2.2.2.4.1.42 uint32_t cpri_iq_init_params_t::rx_2nd_dest_symbol_awareness_int

Generate 2nd destination receive Symbol Awareness interrupt (expected usage: toward DSP timer)

3.2.2.2.4.1.43 uint32_t cpri_iq_init_params_t::tx_symbol_awareness_int

Generate transmit Symbol Awareness interrupt (expected usage: toward DSP timer)

3.2.2.2.4.1.44 uint32_t cpri_iq_init_params_t::warm_reset_enable

is warm reset active - must be active if L1 defense mode 1/2 is enabled

3.2.2.2.4.1.45 uint32_t cpri_iq_init_params_t::transmit_iq_sample_count

Transmit IQ Sample Count - to be configured in TIQSC register for L1 defense stop-restart usage this value is relevant only warm_reset_enable == CPRI_ACTIVE.

3.2.2.2.5 struct cpri_iq_additional_params_t

Adittional parameters passed to IQ initialization function.

Data Fields

- uint32_t rx_transparent_mode:1
- uint32_t tx_transparent_mode:1

3.2.2.2.5.1 Field Documentation

3.2.2.2.5.1.1 uint32_t cpri_iq_additional_params_t::rx_transparent_mode

rx transparent mode active

3.2.2.2.5.1.2 uint32_t cpri_iq_additional_params_t::tx_transparent_mode

tx transparent mode active

3.2.2.2.6 struct cpri_vss_init_params_t

Initialization structure of CPRI VSS data.

Data Fields

- cpri_cores_t initializing_core

- vss_bus_transaction_size_t tx_transaction_size
- vss_bus_transaction_size_t rx_transaction_size
- uint32_t tx_buffer_size
- uint32_t rx_buffer_size
- uint32_t tx_threshold
- uint32_t rx_threshold

3.2.2.2.6.1 Field Documentation

3.2.2.2.6.1.1 cpri_cores_t cpri_vss_init_params_t::initializing_core

core that will initialize and use the CPRI block (one only)

3.2.2.2.6.1.2 vss_bus_transaction_size_t cpri_vss_init_params_t::tx_transaction_size

size of vss bus tx transaction

3.2.2.2.6.1.3 vss_bus_transaction_size_t cpri_vss_init_params_t::rx_transaction_size

size of vss bus rx transaction

3.2.2.2.6.1.4 uint32_t cpri_vss_init_params_t::tx_buffer_size

size of tx buffer

3.2.2.2.6.1.5 uint32_t cpri_vss_init_params_t::rx_buffer_size

size of rx buffer

3.2.2.2.6.1.6 uint32_t cpri_vss_init_params_t::tx_threshold

the amount of transmitted bytes for a VSS tx event

3.2.2.2.6.1.7 uint32_t cpri_vss_init_params_t::rx_threshold

the amount of received bytes for a VSS rx event

3.2.2.2.7 struct cpri_ethernet_init_params_t

Initialization structure of CPRI Ethernet data.

This structure can be used for 2 purposes: 1 - Given as Ethernet initialization parameters in cpri_init_↩ params_t. 2 - Parameters for calling CPRI level0 reconfiguration. The following parameters must not be changed compared to existing configuration: a. Initializing core b. tx_bd_ring_size and rx_bd_ring_size

Data Fields

- cpri_cores_t initializing_core
- uint32_t tx_crc_enable:1
- uint32_t rx_crc_enable:1

- uint32_t rx_long_ethernet_frame_ena:1
- uint32_t rx_illegal_preamble_dis_ena:1
- uint32_t rx_broadcast_enable:1
- uint32_t rx_multicast_enable:1
- uint32_t rx_mac_address_check:1
- uint32_t tx_store_and_forward_enable:1
- uint32_t rx_packet_length_chk_enable:1
- uint32_t little_endian_selection:1
- uint32_t tx_interrupt_en:1
- uint32_t tx_coherency_en:1
- uint32_t rx_interrupt_en:1
- uint32_t mac_address_lsb
- uint16_t mac_address_msb
- uint32_t rx_buffer_size
- uint32_t hash_filter
- uint8_t tx_bd_ring_size
- uint8_t rx_bd_ring_size
- uint8_t tx_coalescing_threshold
- uint8_t rx_coalescing_threshold
- os_mem_type tx_heap
- os_mem_type rx_heap

3.2.2.2.7.1 Field Documentation

3.2.2.2.7.1.1 cpri_cores_t cpri_ethernet_init_params_t::initializing_core

core that will initialize and use the CPRI block (one only)

3.2.2.2.7.1.2 uint32_t cpri_ethernet_init_params_t::tx_crc_enable

Enable insertion of the Ethernet FCS at the end of the frame.

3.2.2.2.7.1.3 uint32_t cpri_ethernet_init_params_t::rx_crc_enable

Enable checking of received frames CRC.

3.2.2.2.7.1.4 uint32_t cpri_ethernet_init_params_t::rx_long_ethernet_frame_ena

Enables receipt of RX frames >1536 bytes.

3.2.2.2.7.1.5 uint32_t cpri_ethernet_init_params_t::rx_illegal_preamble_dis_ena

Enables discard of RX frames with illegal preamble nibble before receipt of SFD.

3.2.2.2.7.1.6 uint32_t cpri_ethernet_init_params_t::rx_broadcast_enable

Enables RX of broadcast packets.

3.2.2.2.7.1.7 uint32_t cpri_ethernet_init_params_t::rx_multicast_enable

Enables RX of the subset of multicast Ethernet packets enabled by the hash function.

3.2.2.2.7.1.8 uint32_t cpri_ethernet_init_params_t::rx_mac_address_check

Enables RX MAC address checking.

3.2.2.2.7.1.9 uint32_t cpri_ethernet_init_params_t::tx_store_and_forward_enable

Used to select TX store and forward mode - frame length should be less than 512bytes in store and forward mode.

3.2.2.2.7.1.10 uint32_t cpri_ethernet_init_params_t::rx_packet_length_chk_enable

Enables RX packet length checking.

3.2.2.2.7.1.11 uint32_t cpri_ethernet_init_params_t::little_endian_selection

Select the endian mode to use for Ethernet RX and TX data (0 - big, 1- little)

3.2.2.2.7.1.12 uint32_t cpri_ethernet_init_params_t::tx_interrupt_en

If TRUE, enable transmitted frames interrupt for Tx confirmation for this channel.

3.2.2.2.7.1.13 uint32_t cpri_ethernet_init_params_t::tx_coherency_en

If TRUE, flushes frame from cache before transmit (buffers must be in system area only)

3.2.2.2.7.1.14 uint32_t cpri_ethernet_init_params_t::rx_interrupt_en

If TRUE, enable transmitted frames interrupt for Rx confirmation for this channel.

3.2.2.2.7.1.15 uint32_t cpri_ethernet_init_params_t::mac_address_lsb

only if rx_mac_address_check enabled - Ethernet address least significant bits

3.2.2.2.7.1.16 uint16_t cpri_ethernet_init_params_t::mac_address_msb

only if rx_mac_address_check enabled - Ethernet address most significant bits

3.2.2.2.7.1.17 uint32_t cpri_ethernet_init_params_t::rx_buffer_size

size of rx buffer

3.2.2.2.7.1.18 uint32_t cpri_ethernet_init_params_t::hash_filter

relevant only for multicast - hash table to filter multicast traffic

3.2.2.2.7.1.19 uint8_t cpri_ethernet_init_params_t::tx_bd_ring_size

receive ethernet buffer descriptor ring size

3.2.2.2.7.1.20 uint8_t cpri_ethernet_init_params_t::rx_bd_ring_size

Transmit ethernet buffer descriptor ring size.

3.2.2.2.7.1.21 uint8_t cpri_ethernet_init_params_t::tx_coalescing_threshold

tx coalescing - the number of transmitted packets for a ethernet tx event - 1

3.2.2.2.7.1.22 uint8_t cpri_ethernet_init_params_t::rx_coalescing_threshold

rx coalescing - the number of received packets for a ethernet tx event -1

3.2.2.2.7.1.23 os_mem_type cpri_ethernet_init_params_t::tx_heap

Heap to use for a channel's BD ring - NOTE: local heap should be used (cpri ethernet can be used by a single core only.

3.2.2.2.7.1.24 os_mem_type cpri_ethernet_init_params_t::rx_heap

Heap to use for a channel's BD ring - NOTE: local heap should be used (cpri ethernet can be used by a single core only.

3.2.2.2.8 struct cpri_hdlc_init_params_t

Initialization structure of CPRI HDLC data.

This structure can be used for 2 purposes: 1 - Given as HDLC initialization parameters in cpri_init_↩ params_t. 2 - Parameters for calling CPRI level0 reconfiguration. The following parameters must not be changed compared to existing configuration: a. Initializing core b. tx_bd_ring_size and rx_bd_ring_size

Data Fields

- cpri_cores_t initializing_core
- uint32_t tx_crc_enable:1
- uint32_t rx_crc_enable:1
- uint32_t tx_store_and_forward_enable:1
- uint32_t rx_packet_length_chk_enable:1
- uint32_t little_endian_selection:1
- uint32_t tx_interrupt_en:1
- uint32_t tx_coherency_en:1
- uint32_t rx_interrupt_en:1
- uint32_t rx_buffer_size
- uint8_t tx_bd_ring_size
- uint8_t rx_bd_ring_size
- os_mem_type tx_heap
- os_mem_type rx_heap

3.2.2.2.8.1 Field Documentation

3.2.2.2.8.1.1 cpri_cores_t cpri_hdlc_init_params_t::initializing_core

core that will initialize and use the CPRI block (one only)

3.2.2.2.8.1.2 uint32_t cpri_hdlc_init_params_t::tx_crc_enable

Enable insertion of the HDLC FCS at the end of the frame.

3.2.2.2.8.1.3 uint32_t cpri_hdlc_init_params_t::rx_crc_enable

Enable checking of received frames CRC.

3.2.2.2.8.1.4 uint32_t cpri_hdlc_init_params_t::tx_store_and_forward_enable

Used to select TX store and forward mode.

3.2.2.2.8.1.5 uint32_t cpri_hdlc_init_params_t::rx_packet_length_chk_enable

Enables RX packet length checking.

3.2.2.2.8.1.6 uint32_t cpri_hdlc_init_params_t::little_endian_selection

Select the endian mode to use for HDLC RX and TX data (0 - big, 1- little)

3.2.2.2.8.1.7 uint32_t cpri_hdlc_init_params_t::tx_interrupt_en

If TRUE, enable transmitted frames interrupt for Tx confirmation for this channel.

3.2.2.2.8.1.8 uint32_t cpri_hdlc_init_params_t::tx_coherency_en

If TRUE, flushes frame from cache before transmit (buffers must be in system area only)

3.2.2.2.8.1.9 uint32_t cpri_hdlc_init_params_t::rx_interrupt_en

If TRUE, enable transmitted frames interrupt for Rx confirmation for this channel.

3.2.2.2.8.1.10 uint32_t cpri_hdlc_init_params_t::rx_buffer_size

size of rx buffer

3.2.2.2.8.1.11 uint8_t cpri_hdlc_init_params_t::tx_bd_ring_size

receive HDLC buffer descriptor ring size

3.2.2.2.8.1.12 uint8_t cpri_hdlc_init_params_t::rx_bd_ring_size

Transmit HDLC buffer descriptor ring size.

3.2.2.2.8.1.13 os_mem_type cpri_hdlc_init_params_t::tx_heap

Heap to use for a channel's BD ring - NOTE: local heap should be used (cpri hdlc can be used by a single core only.

3.2.2.2.8.1.14 os_mem_type cpri_hdlc_init_params_t::rx_heap

Heap to use for a channel's BD ring - NOTE: local heap should be used (cpri hdlc can be used by a single core only.

3.2.2.2.9 struct cpri_init_params_t

Initialization structure of CPRI.

This structure can be used for 2 purposes: 1 - CPRI initialization parameters structure in system bringup phase. 2 - Passed as a parameter for calling CPRI level0 reconfiguration. The following parameters must not be changed compared to existing configuration: a. cpri_iq_init, cpri_vss_init, cpri_ethernet_↩ init and cpri_hdlc_init. 3 - Passed as a parameter for calling CPRI level1 reconfiguration. The following parameters must not be changed compared to existing configuration: a. cpri_iq_init, cpri_vss_init, cpri↩ _ethernet_init and cpri_hdlc_init. b. loop_mode. c. interrupts_table.

Data Fields

- cpri_num_t cpri_num
- cpri_loop_mode_t loop_mode
- cpri_sync_mode_t sync_mode
- ext_sync_act_t external_sync_active
- shared_sync_mode_source_t shared_sync_mode_source
- reset_ack_assert_length_t reset_ack_assert_length
- reset_detection_length_t reset_detection_length
- uint32_t rx_transparent_mode:1
- uint32_t tx_transparent_mode:1
- uint32_t transmit_sync_output:1
- uint32_t sync_state_machine_ena:1
- uint32_t shared_sync_mode:1
- uint32_t delay_direction:1
- uint32_t sync1588:1
- uint32_t remote_reset_indication:1
- uint32_t number_of_errors:2
- uint32_t tx_framer_buffer_size:7
- uint32_t vss_eth_portion_in_frame:6
- uint32_t tx_ex_delay:6
- uint32_t scarmbling_seed
- uint32_t timer_tolerance_value
- uint8_t transmit_delay
- uint8_t sync_delay
- cpri_interrupt_t * interrupts_table
- arch_specific extended_features
- os_status(* cpri_iq_init)(cpri_num_t cpri_num, cpri_iq_init_params_t *init_params, cpri_iq_↩ additional_params_t *additional_init_params)

- os_status(∗ cpri_vss_init)(cpri_num_t cpri_num, cpri_vss_init_params_t ∗init_params, void ∗∗vss_handle)
- os_status(∗ cpri_ethernet_init)(cpri_num_t cpri_num, cpri_ethernet_init_params_t ∗init_params, void ∗∗ethernet_handle)
- os_status(∗ cpri_hdlc_init)(cpri_num_t cpri_num, cpri_hdlc_init_params_t ∗init_params, void ∗∗hdlc_handle)
- cpri_iq_init_params_t ∗ iq_init_params
- cpri_vss_init_params_t ∗ vss_init_params
- cpri_ethernet_init_params_t ∗ ethernet_init_params
- cpri_hdlc_init_params_t ∗ hdlc_init_params

3.2.2.2.9.1 Field Documentation

3.2.2.2.9.1.1 cpri_num_t cpri_init_params_t::cpri_num

number of CPRI block to initialize

3.2.2.2.9.1.2 cpri_loop_mode_t cpri_init_params_t::loop_mode

loop mode to run CPRI with

3.2.2.2.9.1.3 cpri_sync_mode_t cpri_init_params_t::sync_mode

CPRI master or slave.

3.2.2.2.9.1.4 ext_sync_act_t cpri_init_params_t::external_sync_active

sync pulse source

3.2.2.2.9.1.5 shared_sync_mode_source_t cpri_init_params_t::shared_sync_mode_source

shared sync mode, used only if shared_sync_mode is active.

3.2.2.2.9.1.6 reset_ack_assert_length_t cpri_init_params_t::reset_ack_assert_length

For how many HFs reset ack is asserted?

3.2.2.2.9.1.7 reset_detection_length_t cpri_init_params_t::reset_detection_length

reset ack/req detection length

3.2.2.2.9.1.8 uint32_t cpri_init_params_t::rx_transparent_mode

rx transparent mode active

3.2.2.2.9.1.9 uint32_t cpri_init_params_t::tx_transparent_mode

tx transparent mode active

3.2.2.2.9.1.10 uint32_t cpri_init_params_t::transmit_sync_output

should transmit sync be an output (only if external_sync_active is SYNC_GENERATED_LOCALLY)

3.2.2.2.9.1.11 uint32_t cpri_init_params_t::sync_state_machine_ena

Sync state machine enable.

3.2.2.2.9.1.12 uint32_t cpri_init_params_t::shared_sync_mode

independent(0) / shared(1) sync mode

3.2.2.2.9.1.13 uint32_t cpri_init_params_t::delay_direction

When external_sync_active is set to SYNC_GENERATED_EXTERNALLY, define sync direction to positive (active) or negative (not active)

3.2.2.2.9.1.14 uint32_t cpri_init_params_t::sync1588

if active the 10ms sync pulse comes from block 1588.

3.2.2.2.9.1.15 uint32_t cpri_init_params_t::remote_reset_indication

Active if this unit participates in external remote reset indication; master unit indicates reset acknowledge and slave unit indicates reset request.

3.2.2.2.9.1.16 uint32_t cpri_init_params_t::number_of_errors

number of sync errors to hunt state

3.2.2.2.9.1.17 uint32_t cpri_init_params_t::tx_framer_buffer_size

Transmit CPRI Framer Buffer size.

0 -> default size

3.2.2.2.9.1.18 uint32_t cpri_init_params_t::vss_eth_portion_in_frame

Used to select Ethernet/VSS portion in CPRI frame, 0x14 is maximal Ethernet, 0x3f is maximal VSS.

3.2.2.2.9.1.19 uint32_t cpri_init_params_t::tx_ex_delay

Not used.

Kept for backward compatibility.

3.2.2.2.9.1.20 uint32_t cpri_init_params_t::scarmbling_seed

Seed for scrambling.

0 if scrambling is not desired

3.2.2.2.9.1.21 uint32_t cpri_init_params_t::timer_tolerance_value

Timer tolerance value in case of external sync pulse.

3.2.2.2.9.1.22 uint8_t cpri_init_params_t::transmit_delay

TX delay to configure on the link.

3.2.2.2.9.1.23 uint8_t cpri_init_params_t::sync_delay

sync delay to configure on the link

3.2.2.2.9.1.24 cpri_interrupt_t* cpri_init_params_t::interrupts_table

pointer to beginning of cpri_interrupts array

3.2.2.2.9.1.25 arch_specific cpri_init_params_t::extended_features

init extended features

3.2.2.2.9.1.26 os_status(* cpri_init_params_t::cpri_iq_init)(cpri_num_t cpri_num, cpri_iq_init_params_t *init_params, cpri_iq_additional_params_t *additional_init_params)

pointer to cpriIqInitialize function used to enable/disable IQ path

3.2.2.2.9.1.27 os_status(* cpri_init_params_t::cpri_vss_init)(cpri_num_t cpri_num, cpri_vss_init_params_t *init_params, void **vss_handle)

pointer to cpriVssInitialize function used to enable/disable VSS path

3.2.2.2.9.1.28 os_status(* cpri_init_params_t::cpri_ethernet_init)(cpri_num_t cpri_num, cpri_ethernet_init_params_t *init_params, void **ethernet_handle)

pointer to cpriEthernetInitialize function used to enable/disable Ethernet path

3.2.2.2.9.1.29 os_status(* cpri_init_params_t::cpri_hdlc_init)(cpri_num_t cpri_num, cpri_hdlc_init_params_t *init_params, void **hdlc_handle)

pointer to cpriHdlcInitialize function used to enable/disable HDLC path

3.2.2.2.9.1.30 cpri_iq_init_params_t* cpri_init_params_t::iq_init_params

pointer to iq_init_params

3.2.2.2.9.1.31 cpri_vss_init_params_t* cpri_init_params_t::vss_init_params

pointer to vss_init_params

3.2.2.2.9.1.32 cpri_ethernet_init_params_t* cpri_init_params_t::ethernet_init_params

pointer to ethernet_init_params

3.2.2.2.9.1.33 cpri_hdlc_init_params_t* cpri_init_params_t::hdlc_init_params

pointer to hdlc_init_params

3.2.2.2.10 struct cpri_auxiliary_params_t

CPRI auxiliary mode parameters.

This structure should be used as the CPRI LLD parameters for using auxiliary mode with CPRI .

Data Fields

- uint32_t * mask
- uint32_t mask_size

3.2.2.2.10.1 Field Documentation

3.2.2.2.10.1.1 uint32_t* cpri_auxiliary_params_t::mask

pointer to the beginning of the mask registers array

3.2.2.2.10.1.2 uint32_t cpri_auxiliary_params_t::mask_size

mask registers array size

3.2.2.3 Macro Definition Documentation

3.2.2.3.1 #define CPRI_IQ_DEV_NAME0 "c_i0"

CPRI global control flags.

CPRI naming definesCPRI1 IQ device name for SIO

3.2.2.3.2 #define CPRI_IQ_DEV_NAME1 "c_i1"

CPRI2 IQ device name for SIO.

3.2.2.3.3 #define CPRI_IQ_DEV_NAME2 "c_i2"

CPRI3 IQ device name for SIO.

3.2.2.3.4 #define CPRI_IQ_DEV_NAME3 "c_i3"

CPRI4 IQ device name for SIO.

3.2.2.3.5 #define CPRI_IQ_DEV_NAME4 "c_i4"

CPRI5 IQ device name for SIO.

3.2.2.3.6 #define CPRI_IQ_DEV_NAME5 "c_i5"

CPRI6 IQ device name for SIO.

3.2.2.3.7 #define CPRI_VSS_DEV_NAME0 "c_v0"

CPRI1 VSS device name for SIO.

3.2.2.3.8 #define CPRI_VSS_DEV_NAME1 "c_v1"

CPRI2 VSS device name for SIO.

3.2.2.3.9 #define CPRI_VSS_DEV_NAME2 "c_v2"

CPRI3 VSS device name for SIO.

3.2.2.3.10 #define CPRI_VSS_DEV_NAME3 "c_v3"

CPRI4 VSS device name for SIO.

3.2.2.3.11 #define CPRI_VSS_DEV_NAME4 "c_v4"

CPRI5 VSS device name for SIO.

3.2.2.3.12 #define CPRI_VSS_DEV_NAME5 "c_v5"

CPRI6 VSS device name for SIO.

3.2.2.3.13 #define CPRI_ETHERNET_DEV_NAME0 "c_e0"

CPRI1 Ethernet device name for BIO.

3.2.2.3.14 #define CPRI_ETHERNET_DEV_NAME1 "c_e1"

CPRI2 Ethernet device name for BIO.

3.2.2.3.15 #define CPRI_ETHERNET_DEV_NAME2 "c_e2"

CPRI3 Ethernet device name for BIO.

3.2.2.3.16 #define CPRI_ETHERNET_DEV_NAME3 "c_e3"

CPRI4 Ethernet device name for BIO.

3.2.2.3.17 #define CPRI_ETHERNET_DEV_NAME4 "c_e4"

CPRI5 Ethernet device name for BIO.

3.2.2.3.18 #define CPRI_ETHERNET_DEV_NAME5 "c_e5"

CPRI6 Ethernet device name for BIO.

3.2.2.3.19 #define CPRI_HDLC_DEV_NAME0 "c_h0"

CPRI1 HDLC device name for BIO.

3.2.2.3.20 #define CPRI_HDLC_DEV_NAME1 "c_h1"

CPRI2 HDLC device name for BIO.

3.2.2.3.21 #define CPRI_HDLC_DEV_NAME2 "c_h2"

CPRI3 HDLC device name for BIO.

3.2.2.3.22 #define CPRI_HDLC_DEV_NAME3 "c_h3"

CPRI4 HDLC device name for BIO.

3.2.2.3.23 #define CPRI_HDLC_DEV_NAME4 "c_h4"

CPRI5 HDLC device name for BIO.

3.2.2.3.24 #define CPRI_HDLC_DEV_NAME5 "c_h5"

CPRI6 HDLC device name for BIO.

3.2.2.3.25 #define CPRI_IQ_ENABLED &cpriIqInitialize

CPRI IQ device enabled.

3.2.2.3.26 #define CPRI_VSS_ENABLED &cpriVssInitialize

CPRI VSS device enabled.

3.2.2.3.27 #define CPRI_ETHERNET_ENABLED &cpriEthernetInitialize

CPRI Ethernet device enabled.

3.2.2.3.28 #define CPRI_HDLC_ENABLED &cpriHdlcInitialize

CPRI HDLC device enabled.

3.2.2.3.29 #define CPRI_IQ_DISABLED NULL

CPRI IQ device disabled.

3.2.2.3.30 #define CPRI_VSS_DISABLED NULL

CPRI VSS device disabled.

3.2.2.3.31 #define CPRI_ETHERNET_DISABLED NULL

CPRI Ethernet device disabled.

3.2.2.3.32 #define CPRI_HDLC_DISABLED NULL

CPRI HDLC device disabled.

3.2.2.3.33 #define CPRI_NOT_ACTIVE 0

CPRI feature is not active.

3.2.2.3.34 #define CPRI_ACTIVE 1

CPRI feature is active.

3.2.2.3.35 #define CPRI_NOT_SYNCHONIZED 0

CPRI is not synchronized.

3.2.2.3.36 #define CPRI_SYNCHONIZED 1

CPRI is synchronized.

3.2.2.3.37 #define CPRI_IQ_ACTIVE 0x00000001

CPRI IQ active.

3.2.2.3.38 #define CPRI_ETHERNET_ACTIVE 0x00000002

CPRI Ethernet active.

3.2.2.3.39 #define CPRI_HDLC_ACTIVE 0x00000004

CPRI HDLC active.

3.2.2.3.40 #define CPRI_VSS_ACTIVE 0x00000008

CPRI VSS active.

3.2.2.3.41 #define CPRI_ALL_DATA_TYPES_ACTIVE (CPRI_IQ_ACTIVE | CPRI_ETHERNET_ACTIVE | CPRI_HDLC_ACTIVE | CPRI_VSS_ACTIVE)

CPRI all data types active (all previous values OR'ed)

3.2.2.3.42 #define CPRI_DEVICE_DELAYS_CALCULATE 0x00010000

calculate delays of the cpri device

3.2.2.3.43 #define CPRI_DEVICE_STATISTICS_SET 0x00020000

enable cpri events to be used for statistics in sys timers (external to cpri unit)

3.2.2.3.44 #define CPRI_DEVICE_TX_CONTROL_TABLE_WRITE 0x00040000

write a control table TX entrance

3.2.2.3.45 #define CPRI_DEVICE_TX_CONTROL_TABLE_READ 0x00080000

read an entrance from TX control table

3.2.2.3.46 #define CPRI_DEVICE_RX_CONTROL_TABLE_READ 0x00100000

read an entrance from RX control table

3.2.2.3.47 #define CPRI_DEVICE_RX_BFN_COUNTER_READ 0x00200000

read BFN (radio frame) counter from cpri internal counters

3.2.2.3.48 #define CPRI_DEVICE_RX_HFN_COUNTER_READ 0x00400000

read HFN (hyper frame) counter from cpri internal counters

3.2.2.3.49 #define CPRI_DEVICE_TX_BFN_COUNTER_READ 0x00800000

read BFN (radio frame) counter from cpri internal counters

3.2.2.3.50 #define CPRI_DEVICE_TX_HFN_COUNTER_READ 0x01000000

read HFN (hyper frame) counter from cpri internal counters

3.2.2.3.51 #define CPRI_DEVICE_RESET_REQUEST 0x02000000

Force reset request (master) or acknowledge (slave)

3.2.2.3.52 #define CPRI_DEVICE_RESET_REQUEST_DISABLE 0x04000000

Stop forcing reset request (master) or acknowledge (slave)

3.2.2.3.53 #define CPRI_DEVICE_RESET_ENABLE 0x08000000

enable master cpri unit to generate HW reset requests and slave to receive acknowledges

3.2.2.3.54 #define CPRI_DEVICE_RESET_DISABLE 0x10000000

disable master cpri unit to generate HW reset requests and slave to receive acknowledge

3.2.2.3.55 #define CPRI_DEVICE_CHECK_RESET_DETECTED 0x20000000

check if reset request or acknowledge was received

3.2.2.3.56 #define CPRI_DEVICE_TX_BFN_RESET 0x40000000

check if reset request or acknowledge was received

3.2.2.3.57 #define CPRI_DEVICE_DMA_RESTART 0x80000000

restart cpri after it was stopped by warm reset

3.2.2.3.58 #define CPRI_DEVICE_RECONFIGURATION_LEVEL0 0x80010000

Reconfigure the CPRI unit with a new CPRI parameters (see in CPRI block RM); cpri_reconfiguration↩ _level0_param_t is supplied as an input; Returns OS_ERR_BUSY if reconfiguration level 0 of another group is being executed, otherwise OS_SUCCESS.

3.2.2.3.59 #define CPRI_DEVICE_RECONFIGURATION_LEVEL1 0x80020000

Reconfigure the CPRI unit with a new CPRI parameters but same link rate and loopback setting(see in CPRI block RM); cpri_init_params_t is supplied as an input.

3.2.2.3.60 #define CPRI_DEVICE_RECONFIGURATION_LEVEL2 0x80040000

Reconfigure the CPRI unit with a new IQ data type setting (see in CPRI block RM); cpri_iq_init_params↩ _t is supplied as input.

3.2.2.3.61 #define CPRI_STATUS_QUERY 0x80080000

Get the CPRI status flag, such as #CPRI_RECONFIGURATION_LEVEL_0; a pointer to a uint32_t shall be supplied as an in-out argument to hold the status; The reconfiguration bits are cleared upon reading.

3.2.2.3.62 #define CPRI_AXC_0_ACTIVE 0x00000001

channel number 1

3.2.2.3.63 #define CPRI_AXC_1_ACTIVE 0x00000002

channel number 2

3.2.2.3.64 #define CPRI_AXC_2_ACTIVE 0x00000004

channel number 3

3.2.2.3.65 #define CPRI_AXC_3_ACTIVE 0x00000008

channel number 4

3.2.2.3.66 #define CPRI_AXC_4_ACTIVE 0x00000010

channel number 5

3.2.2.3.67 #define CPRI_AXC_5_ACTIVE 0x00000020

channel number 6

3.2.2.3.68 #define CPRI_AXC_6_ACTIVE 0x00000040

channel number 7

3.2.2.3.69 #define CPRI_AXC_7_ACTIVE 0x00000080

channel number 8

3.2.2.3.70 #define CPRI_AXC_8_ACTIVE 0x00000100

channel number 9

3.2.2.3.71 #define CPRI_AXC_9_ACTIVE 0x00000200

channel number 10

3.2.2.3.72 #define CPRI_AXC_10_ACTIVE 0x00000400

channel number 11

3.2.2.3.73 #define CPRI_AXC_11_ACTIVE 0x00000800

channel number 12

3.2.2.3.74 #define CPRI_AXC_12_ACTIVE 0x00001000

channel number 13

3.2.2.3.75 #define CPRI_AXC_13_ACTIVE 0x00002000

channel number 14

3.2.2.3.76 #define CPRI_AXC_14_ACTIVE 0x00004000

channel number 15

3.2.2.3.77 #define CPRI_AXC_15_ACTIVE 0x00008000

channel number 16

3.2.2.3.78 #define CPRI_AXC_16_ACTIVE 0x00010000

channel number 17

3.2.2.3.79 #define CPRI_AXC_17_ACTIVE 0x00020000

channel number 18

3.2.2.3.80 #define CPRI_AXC_18_ACTIVE 0x00040000

channel number 19

3.2.2.3.81 #define CPRI_AXC_19_ACTIVE 0x00080000

channel number 20

3.2.2.3.82 #define CPRI_AXC_20_ACTIVE 0x00100000

channel number 21

3.2.2.3.83 #define CPRI_AXC_21_ACTIVE 0x00200000

channel number 22

3.2.2.3.84 #define CPRI_AXC_22_ACTIVE 0x00400000

channel number 23

3.2.2.3.85 #define CPRI_AXC_23_ACTIVE 0x00800000

channel number 24

3.2.2.3.86 #define MAXIMUM_NUMBER_OF_AXCS 24

maximum number of CPRI channels (AxC's)

3.2.2.3.87 #define CPRI_IQ_THRESHOLD_EVENT 0x00000001

IQ threshold event.

3.2.2.3.88 #define CPRI_IQ_FIRST_THRESHOLD_EVENT 0x00000002

IQ 1st threshold event.

3.2.2.3.89 #define CPRI_IQ_SECOND_THRESHOLD_EVENT 0x00000004

IQ 2nd threshold event.

3.2.2.3.90 #define CPRI_VSS_EVENT 0x00000008

VSS threshold event.

3.2.2.3.91 #define CPRI_HDLC_EVENT 0x00000010

HDLC event.

3.2.2.3.92 #define CPRI_ETHERNET_EVENT 0x00000020

Ethernet event.

3.2.2.3.93 #define CPRI_HFN_TIMING_EVENT 0x00000040

HFN event.

3.2.2.3.94 #define CPRI_BFN_TIMING_EVENT 0x00000080

BFN event.

3.2.2.3.95 #define CPRI_MULTICAST_RX_IQ_THRESHOLD_EVENT 0x00000100

IQ 2nd destination threshold event.

3.2.2.3.96 #define CPRI_IQ_TX_BEGIN_EVENT 0x00000100

IQ transmit begin event.

3.2.2.3.97 #define CPRI_ALL_TER_RER_EVENTS

All events OR'ed.

3.2.2.3.98 #define CPRI_RECEIVE_IQ_OVERRUN_ERROR 0x00000001

IQ receive overrun error.

3.2.2.3.99 #define CPRI_TRANSMIT_IQ_UNDERRUN_ERROR 0x00000002

IQ transmit underrun error.

3.2.2.3.100 #define CPRI_RECEIVE_ETH_MEMORY_OVERRUN_ERROR 0x00000004

Ethernet receive overrun error.

3.2.2.3.101 #define CPRI_TRANSMIT_ETH_UNDERRUN_ERROR 0x00000008

Ethernet transmit underrun error.

3.2.2.3.102 #define CPRI_RECEIVE_ETH_BUFFER_DESCRIPTOR_UNDERRUN_ERRO↩ R 0x00000010

Ethernet receive BD underrun error.

3.2.2.3.103 #define CPRI_RECEIVE_HDLC_OVERRUN_ERROR 0x00000020

HDLC receive overrun error.

3.2.2.3.104 #define CPRI_TRANSMIT_HDLC_UNDERRUN_ERROR 0x00000040

HDLC transmit underrun error.

3.2.2.3.105 #define CPRI_RECEIVE_HDLC_BUFFER_DESCRIPTOR_UNDERRUN_ERRO↩ R 0x00000080

HDLC BD underrun error.

3.2.2.3.106 #define CPRI_RECEIVE_VSS_OVERRUN_ERROR 0x00000100

VSS receive overrun error.

3.2.2.3.107 #define CPRI_TRANSMIT_VSS_UNDERRUN_ERROR 0x00000200

VSS transmit underrun error.

3.2.2.3.108 #define CPRI_EXTERNAL_SYNC_SYNCHRONIZATION_LOST_ERROR 0x00008000

external synch lost error

3.2.2.3.109 #define CPRI_REMOTE_LOS_ERROR 0x00010000

remote LOS error

3.2.2.3.110 #define CPRI_REMOTE_LOF_ERROR 0x00020000

remote LOF error

3.2.2.3.111 #define CPRI_REMOTE_RAI_ERROR 0x00040000

remote RAI error

3.2.2.3.112 #define CPRI_REMOTE_SDI_ERROR 0x00080000

remote SDI error

3.2.2.3.113 #define CPRI_LOCAL_LOS_ERROR 0x00100000

LOS error.

3.2.2.3.114 #define CPRI_LOCAL_LOST_OF_FRAME_ERROR 0x00200000

LOF error.

3.2.2.3.115 #define CPRI_FREQUENCY_ALARM_ERROR 0x00800000

frequency alarm error

3.2.2.3.116 #define CPRI_REMOTE_RESET_ERROR 0x00400000

remote reset event

3.2.2.3.117 #define CPRI_ECC_CONFIGURATION_MEMORY_ERROR 0x00000400

ECC configuration memory error.

3.2.2.3.118 #define CPRI_ECC_DATA_MEMORY_ERROR 0x00000800

ECC data memory error.

3.2.2.3.119 #define CPRI_REMOTE_RESET_ACKNOWLEDGE 0x01000000

remote reset ACK

3.2.2.3.120 #define CPRI_ALL_ERRORS

bitwise-or of all CPRI errors

3.2.2.3.121 #define CPRI_ERROR_MULTIBIT_ECC_SECOND_DESTINATION_DMA_MEMORY 0↩ X00000001

2nd destination DMA memory error

3.2.2.3.122 #define CPRI_ERROR_MULTIBIT_ECC_COMBINER_MEMORY 0X00000002

combiner memory error

3.2.2.3.123 #define CPRI_ERROR_MULTIBIT_ECC_FRAME_ELASTIC_BUFFER_MEMORY 0↩ X00000004

frame elastic memory error

3.2.2.3.124 #define CPRI_ERROR_MULTIBIT_ECC_AUX_MEMORY 0X00000008

auxiliary memory error

3.2.2.3.125 #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_TRANSMIT_HDLC_MEMORY 0↩ X00000010

HDLC framer transmit memory error.

3.2.2.3.126 #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_RECEIVE_HDLC_MEMORY 0↩ X00000020

HDLC framer receive memory error.

3.2.2.3.127 #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_TRANSMIT_ETH_MEMORY 0↩ X00000040

Ethernet framer transmit memory error.

3.2.2.3.128 #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_RECEIVE_ETH_MEMORY 0↩ X00000080

Ethernet framer receive memory error.

3.2.2.3.129 #define CPRI_ERROR_MULTIBIT_ECC_TRANSMIT_CONTROL_WORD_TABLE_MEM↩ ORY 0X00000100

transmit control word table memory error

3.2.2.3.130 #define CPRI_ERROR_MULTIBIT_ECC_RECEIVE_CONTROL_WORD_TABLE_MEMO↩ RY 0X00000200

receive control word memory error

3.2.2.3.131 #define CPRI_ERROR_MULTIBIT_ECC_RECEIVE_DMA_MEMORY 0X00000800

receive DMA memory error

3.2.2.3.132 #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_TRANSMIT_IQ_MEMORY 0↩ X00001000

IQ framer transmit memory error.

3.2.2.3.133 #define CPRI_ERROR_MULTIBIT_ECC_FRAMER_RECEIVE_IQ_MEMORY 0X00002000

IQ framer receive memory error.

3.2.2.3.134 #define CPRI_ERROR_MULTIBIT_ECC_TRANSMIT_CONFIGURATION_MEMORY 0↩ X00004000

transmit configuration memory error

3.2.2.3.135 #define CPRI_ERROR_MULTIBIT_ECC_RECEIVE_CONFIGURATION_MEMORY 0↩ X00008000

receive configuration memory error

3.2.2.4 Enumeration Type Documentation

3.2.2.4.1 enum cpri_loop_mode_t

Enumeration of CPRI loop modes.

3.2.2.4.2 enum cpri_sync_mode_t

Enumeration of CPRI working modes.

3.2.2.4.3 enum ext_sync_act_t

Enumeration of CPRI external sync active modes.

3.2.2.4.4 enum shared_sync_mode_source_t

Enumeration shared sync modes (defines which CPRI is the sync "master").

3.2.2.4.5 enum reset_ack_assert_length_t

Enumeration of reset acknowledge length, counted in number of hyperframes.

3.2.2.4.6 enum reset_detection_length_t

Enumeration of reset acknowledge(master) or request(slave) detection length, counted in number of hyperframes.

3.2.2.4.7 enum cpri_protocol_t

Enumeration of CPRI available protocol types.

3.2.2.4.8 enum cpri_multicast_mode_t

Enumeration of CPRI multicast modes.

3.2.2.4.9 enum mapping_mode_t

Enumeration of CPRI mapping modes.

3.2.2.4.10 enum sampling_width_select_t

Enumeration of CPRI available sample rates.

3.2.2.4.11 enum iq_bus_transaction_size_t

Enumeration of CPRI IQ transaction sizes.

3.2.2.4.12 enum vss_bus_transaction_size_t

Enumeration of CPRI VSS transaction sizes.

3.2.2.4.13 enum oversampling_ratio_t

Enumeration of CPRI BD oversampling available ratios.

3.2.2.4.14 enum cpri_interrupt_type_t

CPRI interrupt types definition.

This enum should be to define the different interrupt types supported by CPRI

RER and TER interrupts can support the following events: CPRI_IQ_THRESHOLD_EVENT, CPRI_IQ_FIRST_THRESHOLD_EVENT, CPRI_IQ_SECOND_THRESHOLD_EVENT, CPRI_VSS_EVENT, CPRI_HDLC_EVENT, CPRI_HDLC_EVENT, CPRI_HFN_TIMING_EVENT, CPRI_BFN_TIMING_EVENT, CPRI_MULTICAST_RX_IQ_THRESHOLD_EVENT (RER only), CPRI_IQ_TX_BEGIN_EVENT (TER only) CPRI driver will assign RER/TER interrupts in consecutive order up to the number of available lines.

Transmit control and receive control interrupts can support the following events: CPRI_VSS_EVENT, CPRI_HDLC_EVENT, CPRI_ETHERENT_EVENT

transmit timing and receive timing interrupts can support the following events: CPRI_HFN_TIMING_EVENT, CPRI_BFN_TIMING_EVENT

3.2.2.5 Function Documentation

3.2.2.5.1 os_status cpriInitialize (cpri_global_init_params_t * *global_parms,* cpri_init_params_t(*) *init_params[]*)

Initializes the CPRI driver's structures

The CPRI main initialization function

Parameters

in	*global_parms*	- cpri global initialization parameters structure
in	*init_params*	- cpri Initialization parameters array.

Returns

OS status

3.2.2.5.2 os_status cpriIqInitialize (cpri_num_t *cpri_num,* cpri_iq_init_params_t * *init_params,* cpri_iq_additional_params_t * *additional_init_params*)

Initializes the CPRI driver's structures

The CPRI IQ initialization function

Parameters

in	*cpri_num*	- cpri CPRI block number.
in	*init_params*	- cpri IQ Initialization parameters.
in	*additional_↩ init_params*	- additional cpri IQ Initialization parameters.

Returns

OS status

3.2.2.5.3 os_status cpriVssInitialize (cpri_num_t *cpri_num*, cpri_vss_init_params_t * *init_params*, void ** *vss_handle*)

Initializes the CPRI driver's structures

The CPRI VSS initialization function

Parameters

in	*cpri_num*	- cpri CPRI block number.
in	*init_params*	- cpri VSS Initialization parameters.
out	*vss_handle*	(B4 family only) - the returned VSS handler.

Returns

OS status

3.2.2.5.4 os_status cpriEthernetInitialize (cpri_num_t *cpri_num*, cpri_ethernet_init_params_t * *init_params*, void ** *ethernet_handle*)

Initializes the CPRI driver's structures

The CPRI VSS Ethernet function

Parameters

in	*cpri_num*	- cpri CPRI block number.

in	*init_params*	- cpri Ethernet Initialization parameters.
out	*ethernet_handle*	(B4 family only) - the returned Ethernet handler.

Returns

OS status

3.2.2.5.5 os_status cpriHdlcInitialize (cpri_num_t *cpri_num*, cpri_hdlc_init_params_t * *init_params*, void ** *hdlc_handle*)

Initializes the CPRI driver's structures

The CPRI HLCD initialization function

Parameters

in	*cpri_num*	- cpri CPRI block number.
in	*init_params*	- cpri HDLC Initialization parameters.
out	*hdlc_handle*	(B4 family only) - the returned HDLC handler.

Returns

OS status

3.2.3 CPRI runtime

3.2.3.1 Overview

CPRI runtime API.

Data Structures

- struct cpri_delays_t
- struct cpri_control_table_entry_t
- struct cpri_iq_open_params_t
- struct cpri_vss_open_params_t
- struct cpri_ethernet_open_params_t
- struct cpri_ethernet_channel_params_t
- struct cpri_hdlc_open_params_t
- struct cpri_hdlc_channel_params_t
- struct cpri_iq_int_cb_param_t
- struct cpri_channel_params_t
- struct cpri_error_param_t
- struct cpri_reconfiguration_level0_param_t

Macros

- #define CPRI_ETHERNET_CMD_RX_POLL (0x00000400 | BIO_LLD_COMMAND | BIO_C↩ OMMAND_READ)
- #define CPRI_ETHERNET_CMD_TX_CONFIRM (0x00000800 | BIO_LLD_COMMAND)
- #define CPRI_HDLC_CMD_RX_POLL (0x00000400 | BIO_LLD_COMMAND | BIO_COMM↩ AND_READ)
- #define CPRI_HDLC_CMD_TX_CONFIRM (0x00000800 | BIO_LLD_COMMAND)

3.2.3.2 Data Structure Documentation

3.2.3.2.1 struct cpri_delays_t

CPRI delay calculation structure.

This structure should be used for the cpriDelaysCalculate to put the calculated data in

Data Fields

- double t_ext
- double t_14
- double t_offset

3.2.3.2.1.1 Field Documentation

3.2.3.2.1.1.1 double cpri_delays_t::t_ext

input - the sum of RX,TX physical and optical delays on the board in nano seconds.

if the value is not known, 0 can be used and accuracy will degrade accordingly. (refer to RM for more details on delays - SFP_RXREC, SFP_TXREC, SFP_RXRE, SFP_TXRE)

3.2.3.2.1.1.2 double cpri_delays_t::t_14

calculated delay

3.2.3.2.1.1.3 double cpri_delays_t::t_offset

calculated delay - relevant for multi-hop configurations only

3.2.3.2.2 struct cpri_control_table_entry_t

CPRI control table entry data.

This structure should be used for writing/reading data for a single control word entry

Data Fields

- uint8_t address

- uint32_t data_register_0
- uint32_t data_register_1
- uint16_t data_register_2

3.2.3.2.2.1 Field Documentation

3.2.3.2.2.1.1 uint8_t cpri_control_table_entry_t::address

address of the control table entry

3.2.3.2.2.1.2 uint32_t cpri_control_table_entry_t::data_register_0

data register

3.2.3.2.2.1.3 uint32_t cpri_control_table_entry_t::data_register_1

data register

3.2.3.2.2.1.4 uint16_t cpri_control_table_entry_t::data_register_2

data register

3.2.3.2.3 struct cpri_iq_open_params_t

CPRI IQ opening parameters.

This structure should be used when calling osSioDeviceOpen() with the CPRI IQ device handle.

Data Fields

- uint8_t * multicast_buffer_data_base
- void(* hfn_tx)(cpri_num_t cpri_num, uint32_t event, void *data)
- void(* hfn_rx)(cpri_num_t cpri_num, uint32_t event, void *data)
- void(* bfn_tx)(cpri_num_t cpri_num, uint32_t event, void *data)
- void(* bfn_rx)(cpri_num_t cpri_num, uint32_t event, void *data)
- void(* iq_tx_begin)(cpri_num_t cpri_num, uint32_t event, void *data)
- void(* iq_multicast_rx)(cpri_num_t cpri_num, uint32_t event, void *data)
- void(* error_callback)(void *param)

3.2.3.2.3.1 Field Documentation

3.2.3.2.3.1.1 uint8_t* cpri_iq_open_params_t::multicast_buffer_data_base

to be used only in multicast or multiunicast modes (otherwise can be 0).

3.2.3.2.3.1.2 void(* cpri_iq_open_params_t::hfn_tx)(cpri_num_t cpri_num, uint32_t event, void *data)

Callback for handling CPRI_HFN_TIMING_EVENT event (initializing core only).

3.2.3.2.3.1.3 void(∗ cpri_iq_open_params_t::hfn_rx)(cpri_num_t cpri_num, uint32_t event, void ∗data)

Callback for handling CPRI_HFN_TIMING_EVENT event (initializing core only).

3.2.3.2.3.1.4 void(∗ cpri_iq_open_params_t::bfn_tx)(cpri_num_t cpri_num, uint32_t event, void ∗data)

Callback for handling CPRI_BFN_TIMING_EVENT event (initializing core only).

3.2.3.2.3.1.5 void(∗ cpri_iq_open_params_t::bfn_rx)(cpri_num_t cpri_num, uint32_t event, void ∗data)

Callback for handling CPRI_BFN_TIMING_EVENT event (initializing core only).

3.2.3.2.3.1.6 void(∗ cpri_iq_open_params_t::iq_tx_begin)(cpri_num_t cpri_num, uint32_t event, void ∗data)

Callback for handling CPRI_IQ_TX_BEGIN_EVENT event (initializing core only).

3.2.3.2.3.1.7 void(∗ cpri_iq_open_params_t::iq_multicast_rx)(cpri_num_t cpri_num, uint32_t event, void ∗data)

Callback for handling CPRI_MULTICAST_RX_IQ_THRESHOLD_EVENT event (initializing core only).

3.2.3.2.3.1.8 void(∗ cpri_iq_open_params_t::error_callback)(void ∗param)

allback for handling general error interrupts (initializing core only)

3.2.3.2.4 struct cpri_vss_open_params_t

CPRI VSS opening parameters.

This structure should be used when calling osSioDeviceOpen() with the CPRI VSS device handle.

Data Fields

- void(∗ error_callback)(void ∗param)

3.2.3.2.4.1 Field Documentation

3.2.3.2.4.1.1 void(∗ cpri_vss_open_params_t::error_callback)(void ∗param)

Callback for handling CPRI_MULTICAST_RX_IQ_THRESHOLD_EVENT event (master core only).

3.2.3.2.5 struct cpri_ethernet_open_params_t

CPRI Ethernet opening parameters.

This structure should be used when calling osBioDeviceOpen() with the CPRI Ethernet device handle.

Data Fields

- void(∗ error_callback)(void ∗param)

3.2.3.2.5.1 Field Documentation

3.2.3.2.5.1.1 void(∗ cpri_ethernet_open_params_t::error_callback)(void ∗param)

Callback for handling general error interrupts (master core only).

3.2.3.2.6 struct cpri_ethernet_channel_params_t

CPRI Ethernet channel opening parameters.

This structure should be used when opening a CPRI Ethernet channel

Data Fields

- uint32_t bd_ring_steering_bits:3
- uint32_t buffer_steering_bits:3
- cpri_buffer_attributes_t bd_ring_attributes
- cpri_buffer_attributes_t buffer_attributes

3.2.3.2.6.1 Field Documentation

3.2.3.2.6.1.1 uint32_t cpri_ethernet_channel_params_t::bd_ring_steering_bits

steering bits for the BD ring)

3.2.3.2.6.1.2 uint32_t cpri_ethernet_channel_params_t::buffer_steering_bits

steering bits for the data buffers

3.2.3.2.6.1.3 cpri_buffer_attributes_t cpri_ethernet_channel_params_t::bd_ring_attributes

BD ring attributes.

3.2.3.2.6.1.4 cpri_buffer_attributes_t cpri_ethernet_channel_params_t::buffer_attributes

buffer attributes

3.2.3.2.7 struct cpri_hdlc_open_params_t

CPRI HDLC opening parameters.

This structure should be used when calling osBioDeviceOpen() with the CPRI HDLC device handle.

Data Fields

- void(∗ error_callback)(void ∗param)

3.2.3.2.7.1 Field Documentation

3.2.3.2.7.1.1 void(∗ cpri_hdlc_open_params_t::error_callback)(void ∗param)

Callback for handling general error interrupts (master core only).

3.2.3.2.8 struct cpri_hdlc_channel_params_t

CPRI HDLC channel opening parameters.

This structure should be used when opening a CPRI Ethernet channel

Data Fields

- uint32_t bd_ring_steering_bits:3
- uint32_t buffer_steering_bits:3
- cpri_buffer_attributes_t bd_ring_attributes
- cpri_buffer_attributes_t buffer_attributes

3.2.3.2.8.1 Field Documentation

3.2.3.2.8.1.1 uint32_t cpri_hdlc_channel_params_t::bd_ring_steering_bits

steering bits for the BD ring)

3.2.3.2.8.1.2 uint32_t cpri_hdlc_channel_params_t::buffer_steering_bits

steering bits for the data buffers

3.2.3.2.8.1.3 cpri_buffer_attributes_t cpri_hdlc_channel_params_t::bd_ring_attributes

BD ring attributes.

3.2.3.2.8.1.4 cpri_buffer_attributes_t cpri_hdlc_channel_params_t::buffer_attributes

buffer attributes

3.2.3.2.9 struct cpri_iq_int_cb_param_t

CPRI IQ interrupts callback type.

This structure should be used when supplying a callback for IQ device

Data Fields

- void * callback_param
- cpri_num_t cpri_num
- uint32_t event_type

3.2.3.2.9.1 Field Documentation

3.2.3.2.9.1.1 void* cpri_iq_int_cb_param_t::callback_param

Callback for handling IQ interrupts.

3.2.3.2.9.1.2 cpri_num_t cpri_iq_int_cb_param_t::cpri_num

CPRI block.

3.2.3.2.9.1.3 uint32_t cpri_iq_int_cb_param_t::event_type

Event description.

3.2.3.2.10 struct cpri_channel_params_t

CPRI IQ channel opening parameters.

This structure should be used when opening a CPRI IQ channel

3.2.3.2.11 struct cpri_error_param_t

CPRI error parameters.

This sturcture is passed to the error callback when error event occurs. It contains the needed information so that the callback can know which CPRI unit and what is the error.

Data Fields

- cpri_num_t cpri_num
- uint32_t error_events
- uint32_t error_ecc_multibit_events

3.2.3.2.11.1 Field Documentation

3.2.3.2.11.1.1 cpri_num_t cpri_error_param_t::cpri_num

CPRI block.

3.2.3.2.11.1.2 uint32_t cpri_error_param_t::error_events

Error events description.

3.2.3.2.11.1.3 uint32_t cpri_error_param_t::error_ecc_multibit_events

ECC multibit error events description.

3.2.3.2.12 struct cpri_reconfiguration_level0_param_t

CPRI reconfiguration level0 parameters.

This structure is passed to the reconfiguration level0 flow, which is invoked via CPRI_DEVICE_RECO↩ NFIGURATION_LEVEL0 SIO device control command.

Data Fields

- cpri_link_rate_t maximal_desired_link_rate
- cpri_link_rate_t minimal_accepted_link_rate
- cpri_init_params_t(∗ cpri_init_params)[ ]

3.2.3.2.12.1 Field Documentation

3.2.3.2.12.1.1 cpri_link_rate_t cpri_reconfiguration_level0_param_t::maximal_desired_link_rate

maximal link rate.

must not be greater then RCWLR defined rate

3.2.3.2.12.1.2 cpri_link_rate_t cpri_reconfiguration_level0_param_t::minimal_accepted_link_rate

minimal link rate accepted without failure

3.2.3.2.12.1.3 cpri_init_params_t(∗ cpri_reconfiguration_level0_param_t::cpri_init_params)[ ]

reconfiguration parameters of all the CPRI units

3.2.3.3 Macro Definition Documentation

3.2.3.3.1 #define CPRI_ETHERNET_CMD_RX_POLL (0x00000400 | BIO_LLD_COMMAND | BIO_COMMAND_READ)

Process received frames, when Rx interrupt is disabled (param: NULL).

3.2.3.3.2 #define CPRI_ETHERNET_CMD_TX_CONFIRM (0x00000800 | BIO_LLD_COMMAND)

Confirm transmitted frames (param: NULL).

3.2.3.3.3 #define CPRI_HDLC_CMD_RX_POLL (0x00000400 | BIO_LLD_COMMAND | BIO_COMMAND_READ)

Process received frames, when Rx interrupt is disabled (param: NULL).

3.2.3.3.4 #define CPRI_HDLC_CMD_TX_CONFIRM (0x00000800 | BIO_LLD_COMMAND)

Confirm transmitted frames (param: NULL).

3.2.4 CPRI B4860 architecture-specific definitions

3.2.4.1 Overview

```
Definition of CPRI B4860 architecture specific defines, data
           structures and functions.
```

Data Structures

- struct cpri_global_init_params_t
- union cpri_buffer_attributes_t

Macros

- #define REGISTERS_TYPE soc_ccsr_map_t
- #define DEVICE_REGISTERS g_soc_ccsr_map
- #define CPRI_GENERAL_REGISTERS (DEVICE_REGISTERS->cpri.cpri_unit[0].cpri_↩ general_registers)
- #define CPRI_MAX_NUM_OF_CORES OS_SOC_MAX_NUM_OF_CORES
- #define CPRI_ICCR_CLOCKS 0x000000FF
- #define CPRI_NUM_OF_TER_RER_INTERRUPTS_PER_UNIT 16
- #define CPRI_NUM_OF_TX_CONTROL_INTERRUPTS_PER_UNIT 1
- #define CPRI_NUM_OF_RX_CONTROL_INTERRUPTS_PER_UNIT 1
- #define CPRI_NUM_OF_TX_TIMING_INTERRUPTS_PER_UNIT 1
- #define CPRI_NUM_OF_RX_TIMING_INTERRUPTS_PER_UNIT 1
- #define CPRI_NUM_OF_ERROR_INTERRUPTS_PER_UNIT 1
- #define CPRI_TOTAL_NUM_OF_TER_RER_INTERRUPTS 16
- #define CPRI_TOTAL_NUM_OF_TX_CONTROL_INTERRUPTS NUM_OF_CPRI_MODUL↩ ES
- #define CPRI_TOTAL_NUM_OF_RX_CONTROL_INTERRUPTS NUM_OF_CPRI_MODUL↩ ES
- #define CPRI_TOTAL_NUM_OF_TX_TIMING_INTERRUPTS NUM_OF_CPRI_MODULES
- #define CPRI_TOTAL_NUM_OF_RX_TIMING_INTERRUPTS NUM_OF_CPRI_MODULES

- #define CPRI_TOTAL_NUM_OF_ERROR_INTERRUPTS NUM_OF_CPRI_MODULES
- #define CPRI_NUMBER_OF_CLOCK_DOMAINS (NUM_OF_CPRI_GROUPS)
- #define CPRI_GET_CLOCK_DOMAIN_INDEX(CPRI_UNIT_INIT_PARAMS) ((uint8_t)CPRI_UNIT_INIT_PARAMS->group)

Enumerations

- enum cpri_group_t
- enum cpri_link_rate_t

Functions

- cpri_num_t cpriPairNumGet (cpri_num_t cpri_num)
- double cpriSerdesDelaysCalculate (cpri_num_t cpri_num)
- os_status cpriArchStatisticsSet ()
- os_status cpriPowerGating (bool powerDown)

3.2.4.2 Data Structure Documentation

3.2.4.2.1 struct cpri_global_init_params_t

Global initialization structure of CPRI.

This structure is passed once for a device init and include parameters for initialization of all the CPRI units used on the device.

Data Fields

- cpri_cores_t initializing_core
- uint8_t cpri_num_of_used_units
- cpri_link_rate_t maximal_desired_link_rate
- cpri_link_rate_t minimal_accepted_link_rate

3.2.4.2.1.1 Field Documentation

3.2.4.2.1.1.1 cpri_cores_t cpri_global_init_params_t::initializing_core

initializing_core

3.2.4.2.1.1.2 uint8_t cpri_global_init_params_t::cpri_num_of_used_units

number of active cpri units

3.2.4.2.1.1.3 cpri_link_rate_t cpri_global_init_params_t::maximal_desired_link_rate

maximal link rate.

must not be greater then RCWLR defined rate

3.2.4.2.1.1.4 cpri_link_rate_t cpri_global_init_params_t::minimal_accepted_link_rate

minimal link rate accepted without failure

3.2.4.2.2 union cpri_buffer_attributes_t

CPRI buffer attributes 16 bits width register layout.

3.2.4.2.2.1 Field Documentation

3.2.4.2.2.1.1 uint16_t cpri_buffer_attributes_t::c

Cacheable.

3.2.4.2.2.1.2 uint16_t cpri_buffer_attributes_t::lru

0 - Normal L2 cache behavior, 1 - Access will be marked in L2 cache as least recently used.

3.2.4.2.2.1.3 uint16_t cpri_buffer_attributes_t::w

0 - Write back mode, 1 - Write through mode

3.2.4.2.2.1.4 uint16_t cpri_buffer_attributes_t::m

Coherency.

3.2.4.2.2.1.5 uint16_t cpri_buffer_attributes_t::g

Guarded.

3.2.4.2.2.1.6 uint16_t cpri_buffer_attributes_t::l2pid

L2 partition ID.

3.2.4.2.2.1.7 uint16_t cpri_buffer_attributes_t::enhanced

Valid if Enhanced bit is set; Specifies the Enhanced type value; Used by PAMU for command classification.

Enhanced access

3.2.4.2.2.1.8 uint16_t cpri_buffer_attributes_t::liodn

Logical IO device number; Used by PAMU.

3.2.4.3 Macro Definition Documentation

3.2.4.3.1 #define REGISTERS_TYPE soc_ccsr_map_t

Chips' CCSR structure.

3.2.4.3.2 #define DEVICE_REGISTERS g_soc_ccsr_map

pointer to chips' CCSR base

3.2.4.3.3 #define CPRI_GENERAL_REGISTERS (DEVICE_REGISTERS->cpri.cpri_unit[0].cpri↩ _general_registers)

pointer to CPRI general registers base

3.2.4.3.4 #define CPRI_MAX_NUM_OF_CORES OS_SOC_MAX_NUM_OF_CORES

maximum number of cores in this platform

3.2.4.3.5 #define CPRI_ICCR_CLOCKS 0x000000FF

bitwise constant for all CPRI units' clocks for ICCR register

3.2.4.3.6 #define CPRI_NUM_OF_TER_RER_INTERRUPTS_PER_UNIT 16

maximum number of transmit and receive interrupts per CPRI unit

3.2.4.3.7 #define CPRI_NUM_OF_TX_CONTROL_INTERRUPTS_PER_UNIT 1

maximum number of transmit control interrupts interrupts per CPRI unit

3.2.4.3.8 #define CPRI_NUM_OF_RX_CONTROL_INTERRUPTS_PER_UNIT 1

maximum number of receive control interrupts per CPRI unit

3.2.4.3.9 #define CPRI_NUM_OF_TX_TIMING_INTERRUPTS_PER_UNIT 1

maximum number of transmit timing interrupts per CPRI unit

3.2.4.3.10 #define CPRI_NUM_OF_RX_TIMING_INTERRUPTS_PER_UNIT 1

maximum number of receive timing interrupts per CPRI unit

3.2.4.3.11 #define CPRI_NUM_OF_ERROR_INTERRUPTS_PER_UNIT 1

maximum number of error interrupts per CPRI unit

3.2.4.3.12 #define CPRI_TOTAL_NUM_OF_TER_RER_INTERRUPTS 16

maximum number of transmit and receive interrupts for all CPRI units

3.2.4.3.13 #define CPRI_TOTAL_NUM_OF_TX_CONTROL_INTERRUPTS NUM_OF_CPRI_MODU↩ LES

maximum number of transmit control interrupts for all CPRI units

3.2.4.3.14 #define CPRI_TOTAL_NUM_OF_RX_CONTROL_INTERRUPTS NUM_OF_CPRI_MODU↩ LES

maximum number of receive control interrupts for all CPRI units

3.2.4.3.15 #define CPRI_TOTAL_NUM_OF_TX_TIMING_INTERRUPTS NUM_OF_CPRI_MODULES

maximum number of transmit timing interrupts for all CPRI units

3.2.4.3.16 #define CPRI_TOTAL_NUM_OF_RX_TIMING_INTERRUPTS NUM_OF_CPRI_MODULES

maximum number of receive timing interrupts for all CPRI units

3.2.4.3.17 #define CPRI_TOTAL_NUM_OF_ERROR_INTERRUPTS NUM_OF_CPRI_MODULES

maximum number of error interrupts for all CPRI units

3.2.4.3.18 #define CPRI_NUMBER_OF_CLOCK_DOMAINS (NUM_OF_CPRI_GROUPS)

maximum number of clocks domain

3.2.4.3.19 #define CPRI_GET_CLOCK_DOMAIN_INDEX(*CPRI_UNIT_INIT_PARAMS*) ((uint8_t)CPRI_UNIT_INIT_PARAMS->group)

returns clock domain for a given CPRI unit

3.2.4.4 Enumeration Type Documentation

3.2.4.4.1 enum cpri_group_t

Enumeration of CPRI groups.

3.2.4.4.2 enum cpri_link_rate_t

Enumeration of CPRI available link rates.

3.2.4.5 Function Documentation

3.2.4.5.1 cpri_num_t cpriPairNumGet (cpri_num_t *cpri_num*)

get CPRI pair for a specific CPRI unit.

Parameters

in	*cpri_num*	- cpri CPRI block number.

Returns

cpri_num_t - number of pair cpri module

3.2.4.5.2 double cpriSerdesDelaysCalculate (cpri_num_t *cpri_num*)

calculate CPRI serdes delays

Parameters

in	*cpri_num*	- number of cpri module

Returns

double - value of serdes delay (in seconds)

3.2.4.5.3 os_status cpriArchStatisticsSet ( )

activates statistics measurement in device timers

Returns

os_status - OS status

3.2.4.5.4 os_status cpriPowerGating (bool *powerDown*)

Provides a mechanism for gating clocks to all CPRI

Parameters

in	*powerDown*	- If set to TRUE, will disable CPRI If set to False, will power up CPRI

Returns

os_status - OS status

3.3 sRIO Module API

3.3.1 Overview

```
sRIO runtime API

sRIO initialize.
```

Modules

- sRIO initialization
- sRIO runtime

3.3.2 sRIO initialization

3.3.2.1 Overview

```
sRIO initialization API.
```

Data Structures

- struct srio_init_params_t
- struct srio_sys_init_param_t

Macros

- #define SRIO_DEFAULT_HOST_ID 0
- #define SRIO_ACCEPT_ALL 0x00000001

Functions

- os_status initSrioSystem (void *device_handle)
- os_status srioInitialize (struct srio_init_params_s *init_params, unsigned int num_devices, uint32_t max_num_srio_devices, ocn_dma_id_t ocn_dma_id_to_use, bool initiateSrio)
- os_status srioDevicesInitialize (srio_port_id_t srio_port, uint16_t num_devices, os_mem_type heap)
- os_status srioDeviceAdd (srio_device_t *device)
- os_status srioRecover (srio_port_id_t port_num)
- uint16_t srioAlternateIdSet (srio_port_id_t port_num, uint16_t alternate_id)
- void srioAlternateIdDisable (srio_port_id_t port_num)
- uint32_t srioAcceptAllConfigure (srio_port_id_t port_num, uint32_t accept)

3.3.2.2 Data Structure Documentation

3.3.2.2.1 struct srio_init_params_t

Initialization structure of SRIO.

Data Fields

- unsigned int num_ports
- ocn_dma_id_t ocn_dma_to_use

3.3.2.2.1.1 Field Documentation

3.3.2.2.1.1.1 unsigned int srio_init_params_t::num_ports

Number of ports to initialize, equals number of element in init_ports[].

3.3.2.2.1.1.2 ocn_dma_id_t srio_init_params_t::ocn_dma_to_use

Which OCeaN DMA to use for enumeration process at system initiation.

3.3.2.2.2 struct srio_sys_init_param_t

sRIO system initiation structure.

Data Fields

- srio_port_id_t port_num
- srio_connectivity_t srio_connectivity
- bool enumerate

3.3.2.2.2.1 Field Documentation

3.3.2.2.2.1.1 srio_port_id_t srio_sys_init_param_t::port_num

Port num to enumerate.

3.3.2.2.2.1.2 srio_connectivity_t srio_sys_init_param_t::srio_connectivity

Connectivity mode in SerDes.

3.3.2.2.2.1.3 bool srio_sys_init_param_t::enumerate

Will be set to 1 if enumeration is to take place, 0 if not.

3.3.2.3 Macro Definition Documentation

3.3.2.3.1 #define SRIO_DEFAULT_HOST_ID 0

sRIO Default Host ID

3.3.2.3.2 #define SRIO_ACCEPT_ALL 0x00000001

Accept all.

3.3.2.4 Function Documentation

3.3.2.4.1 os_status initSrioSystem (void * *device_handle*)

Scan or discover the SRIO system.

Parameters

device_handle	- pointer to srio_sys_init_param_t structure

Returns

OS status

3.3.2.4.2 os_status srioInitialize (struct srio_init_params_s * *init_params*, unsigned int *num_devices*, uint32_t *max_num_srio_devices*, ocn_dma_id_t *ocn_dma_id_to_use*, bool *initiateSrio*)

Basic sRIO initializations.

Parameters

init_params	- a pointer to srio_init_params_s structure
num_devices	- Number of available sRIO ports
max_num_srio_devices	- Maximum number of connected sRIO devices in the system
ocn_dma_id_to_use	- OCeaN DMA id to use for sRIO system initiation
initiateSrio	- true if SmartDSP is owner of sRIO initiation

3.3.2.4.3 os_status srioDevicesInitialize (srio_port_id_t *srio_port*, uint16_t *num_devices*, os_mem_type *heap*)

Allocate memory for sRIO device management.

Parameters

in	*srio_port*	- Lynx PHY port number
in	*num_devices*	- number of devices to allocate.
in	*heap*	- must be shared and non catchable.

Returns

OS status.

3.3.2.4.4 os_status srioDeviceAdd (srio_device_t * *device*)

Add a device to sRIO device management.

Parameters

in	*device*	- pointer to a device structure.

Returns

OS status.

3.3.2.4.5 os_status srioRecover (srio_port_id_t *port_num*)

Recover from an error on the port num

Parameters

in	*port_num*	- Lynx PHY port number to recover

Returns

OS Status

3.3.2.4.6 uint16_t srioAlternateIdSet (srio_port_id_t *port_num,* uint16_t *alternate_id*)

Configures an alternate ID for this port.

Parameters

in	*port_num*	- Lynx PHY port number
in	*alternate_id*	- Alternate ID

Returns

Previous alternate ID.

3.3.2.4.7 void srioAlternateIdDisable (srio_port_id_t *port_num*)

Disable an alternate ID for this port.

Parameters

in	*port_num*	- Lynx PHY port number

3.3.2.4.8 uint32_t srioAcceptAllConfigure (srio_port_id_t *port_num*, uint32_t *accept*)

Allows the port to accept inputs to any device ID.

Parameters

in	*port_num*	- Lynx PHY port number
in	*accept*	- SRIO_ACCEPT_ALL or 0

Returns

Current status of accept all

3.3.3 sRIO runtime

3.3.3.1 Overview

sRIO runtime API.

Data Structures

- struct srio_device_t
- struct srio_in_win_config_t
- struct srio_out_win_config_t
- struct srio_maint_win_config_t

Macros

- #define SRIO_ATMU_DEFAULT_WINDOW 0
- #define SRIO_LCFBA_ADDRESS 0x3FF000000ULL
- #define SRIO_LCF_ATMU_OUT_WIN_SIZE SRIO_ATMU_OUT_WIN_1M
- #define srioGetID() (g_srio_id + 0) /∗ The + prevents usage as lvalue ∗/
- #define srioGetAnyID() (g_srio_any_id + 0) /∗ The + prevents usage as lvalue ∗/
- #define srioIsSmallSystem() (g_srio_small_system == TRUE)

Typedefs

- typedef uint16_t srio_dev_id_t

Enumerations

- enum srio_flow_level_t { LOW_FLOW_LEVEL = 0x00000000, MEDIUM_FLOW_LEVEL = 0x04000000, HIGH_FLOW_LEVEL = 0x08000000 }
- enum srio_port_id_t
- enum srio_connectivity_t { DISABLE_LB, DIGITAL_LB, EXTERNAL_LB }
- enum srio_write_type_t
- enum srio_in_target_interface_t
- enum srio_atmu_in_win_sz_t {
  SRIO_ATMU_IN_WIN_LAST = 0x0, SRIO_ATMU_IN_WIN_4K = 0xB, SRIO_ATMU_IN_WIN_8K = 0xC,
  SRIO_ATMU_IN_WIN_16K = 0xD, SRIO_ATMU_IN_WIN_32K = 0xE, SRIO_ATMU_IN_WIN_64K = 0xF,
  SRIO_ATMU_IN_WIN_128K = 0x10, SRIO_ATMU_IN_WIN_256K = 0x11, SRIO_ATMU_IN_WIN_512K = 0x12,
  SRIO_ATMU_IN_WIN_1M = 0x13, SRIO_ATMU_IN_WIN_2M = 0x14, SRIO_ATMU_IN_WIN_4M = 0x15,
  SRIO_ATMU_IN_WIN_8M = 0x16, SRIO_ATMU_IN_WIN_16M = 0x17, SRIO_ATMU_IN_WIN_32M = 0x18,
  SRIO_ATMU_IN_WIN_64M = 0x19, SRIO_ATMU_IN_WIN_128M = 0x1A, SRIO_ATMU_IN_WIN_256M = 0x1B,
  SRIO_ATMU_IN_WIN_512M = 0x1C, SRIO_ATMU_IN_WIN_1G = 0x1D, SRIO_ATMU_IN_WIN_2G = 0x1E,
  SRIO_ATMU_IN_WIN_4G = 0x1F, SRIO_ATMU_IN_WIN_8G = 0x20, SRIO_ATMU_IN_WIN_16G = 0x21 }
- enum srio_atmu_out_win_sz_t {
  SRIO_ATMU_OUT_WIN_LAST = 0x0, SRIO_ATMU_OUT_WIN_4K = 0xB, SRIO_ATMU_OUT_WIN_8K = 0xC,
  SRIO_ATMU_OUT_WIN_16K = 0xD, SRIO_ATMU_OUT_WIN_32K = 0xE, SRIO_ATMU_OUT_WIN_64K = 0xF,
  SRIO_ATMU_OUT_WIN_128K = 0x10, SRIO_ATMU_OUT_WIN_256K = 0x11, SRIO_ATMU_OUT_WIN_512K = 0x12,
  SRIO_ATMU_OUT_WIN_1M = 0x13, SRIO_ATMU_OUT_WIN_2M = 0x14, SRIO_ATMU_OUT_WIN_4M = 0x15,
  SRIO_ATMU_OUT_WIN_8M = 0x16, SRIO_ATMU_OUT_WIN_16M = 0x17, SRIO_ATMU_OUT_WIN_32M = 0x18,
  SRIO_ATMU_OUT_WIN_64M = 0x19, SRIO_ATMU_OUT_WIN_128M = 0x1A, SRIO_ATMU_OUT_WIN_256M = 0x1B,
  SRIO_ATMU_OUT_WIN_512M = 0x1C, SRIO_ATMU_OUT_WIN_1G = 0x1D, SRIO_ATMU_OUT_WIN_2G = 0x1E,
  SRIO_ATMU_OUT_WIN_4G = 0x1F, SRIO_ATMU_OUT_WIN_8G = 0x20, SRIO_ATMU_OUT_WIN_16G = 0x21,
  SRIO_ATMU_OUT_WIN_32G = 0x22, SRIO_ATMU_OUT_WIN_64G = 0x23 }
- enum srio_traffic_management_t { TM_MODE_DISABLE, TM_MODE_BASIC }

Functions

- void srioOutboundWindowFree (srio_port_id_t port_num, uint32_t win_num)
- os_status srioOutboundWindowFind (srio_port_id_t port_num, uint32_t *win_num)
- void srioOutboundWindowEnable (srio_port_id_t port_num, uint32_t win_num)
- void srioOutboundWindowDisable (srio_port_id_t port_num, uint32_t win_num)
- os_status srioOutboundWindowOpen (srio_port_id_t port_num, srio_out_win_config_t *srio_out_win_config)
- os_status srioInboundWindowFind (srio_port_id_t port_num, uint32_t *win_num)
- void srioInboundWindowFree (srio_port_id_t port_num, uint32_t win_num)
- void srioInboundWindowEnable (srio_port_id_t port_num, uint32_t win_num)
- void srioInboundWindowDisable (srio_port_id_t port_num, uint32_t win_num)
- os_status srioInboundWindowOpen (srio_port_id_t port_num, srio_in_win_config_t *srio_in_win_config)
- os_status srioMaintenanceAccess (os_dma_handle dma, uint32_t atmu_win_id, srio_port_id_t port_num, uint64_t data, uint32_t offset, uint32_t access, uint32_t ch, int mode)
- os_status srioMaintenanceTargetSet (srio_port_id_t port_num, uint32_t atmu_win_id, srio_dev_id_t target_id, uint8_t hop_count, uint32_t offset)
- void srioMaintenanceAtmuFree (srio_port_id_t port_num, uint32_t atmu_win_id)
- os_status srioMaintenanceAtmuOpen (srio_port_id_t port_num, srio_maint_win_config_t *srio_main_win_config)
- os_status srioTrafficManagementSupport (srio_traffic_management_t tm_mode)
- os_status srioDeviceIdGetByAttr (srio_device_t *device)
- void srioDeviceGetByIndex (srio_device_t **device, srio_port_id_t srio_port, uint16_t index)
- void srioDeviceGetById (srio_device_t **device, srio_port_id_t srio_port, srio_dev_id_t dev_id)
- uint16_t srioNumDiscoverdDevicesGet ()
- void srioClearPortErrors (srio_port_id_t port)

3.3.3.2 Data Structure Documentation

3.3.3.2.1 struct srio_device_t

sRIO device information structure.

Data Fields

- uint16_t dev_id
- uint16_t hop_count
- uint16_t switch_port_num
- srio_port_id_t srio_port_num

3.3.3.2.1.1 Field Documentation

3.3.3.2.1.1.1 uint16_t srio_device_t::dev_id

Device ID on sRIO.

3.3.3.2.1.1.2 uint16_t srio_device_t::hop_count

Hop count to device.

3.3.3.2.1.1.3 uint16_t srio_device_t::switch_port_num

Port number on which it is connected to switch at hop_count-1.

3.3.3.2.1.1.4 srio_port_id_t srio_device_t::srio_port_num

sRIO Port number on which it is connected to DSP

3.3.3.2.2 struct srio_in_win_config_t

Inbound window configuration.

Default window - only protect is considered. Non default windows are enabled if configured.

Data Fields

- unsigned int win_num
- uint64_t base
- uint64_t translation
- bool snoop_enable
- srio_atmu_in_win_sz_t win_size
- srio_in_target_interface_t target_interface

3.3.3.2.2.1 Field Documentation

3.3.3.2.2.1.1 unsigned int srio_in_win_config_t::win_num

Inbound ATMU window number.

3.3.3.2.2.1.2 uint64_t srio_in_win_config_t::base

34 bit RapidIO address

3.3.3.2.2.1.3 uint64_t srio_in_win_config_t::translation

36 bit OCeaN address.

3.3.3.2.2.1.4 bool srio_in_win_config_t::snoop_enable

snoop local processor

3.3.3.2.2.1.5 srio_atmu_in_win_sz_t srio_in_win_config_t::win_size

Inbound ATMU window size.

3.3.3.2.2.1.6 srio_in_target_interface_t srio_in_win_config_t::target_interface

Inbound ATMU target interface.

3.3.3.2.3 struct srio_out_win_config_t

Outbound window configuration.

Windows are opened disabled, and need to be enabled before use and disabled before free.

Warning

For window 0 - Base is not considered.

Data Fields

- unsigned int win_num
- uint16_t srio_id
- uint64_t base
- uint64_t translation
- srio_atmu_out_win_sz_t win_size
- srio_flow_level_t srio_flow_level
- srio_write_type_t srio_write_access_type

3.3.3.2.3.1 Field Documentation

3.3.3.2.3.1.1 unsigned int srio_out_win_config_t::win_num

Window number.

3.3.3.2.3.1.2 uint16_t srio_out_win_config_t::srio_id

Device ID towards which to direct the flow.

3.3.3.2.3.1.3 uint64_t srio_out_win_config_t::base

36 bit Ocean address.

3.3.3.2.3.1.4 uint64_t srio_out_win_config_t::translation

34 bit RapidIO address

3.3.3.2.3.1.5 srio_atmu_out_win_sz_t srio_out_win_config_t::win_size

Window size.

3.3.3.2.3.1.6 srio_flow_level_t srio_out_win_config_t::srio_flow_level

Priority level.

3.3.3.2.3.1.7 srio_write_type_t srio_out_win_config_t::srio_write_access_type

Access type on RapidIO.

3.3.3.2.4 struct srio_maint_win_config_t

Maintenance window configuration.

Data Fields

- uint32_t win_num
- os_phys_ptr base
- srio_atmu_out_win_sz_t win_size
- srio_dev_id_t srio_id
- uint8_t hop_count
- uint32_t offset

3.3.3.2.4.1 Field Documentation

3.3.3.2.4.1.1 uint32_t srio_maint_win_config_t::win_num

Window number.

3.3.3.2.4.1.2 os_phys_ptr srio_maint_win_config_t::base

36 bit OCeaN address

3.3.3.2.4.1.3 srio_atmu_out_win_sz_t srio_maint_win_config_t::win_size

Window size.

3.3.3.2.4.1.4 srio_dev_id_t srio_maint_win_config_t::srio_id

Device ID towards which to direct the flow.

3.3.3.2.4.1.5 uint8_t srio_maint_win_config_t::hop_count

Hop count to target device.

3.3.3.2.4.1.6 uint32_t srio_maint_win_config_t::offset

Target Register offset from RapidIO configuration space.

3.3.3.3 Macro Definition Documentation

3.3.3.3.1 #define SRIO_ATMU_DEFAULT_WINDOW 0

ATMU window at index 0 is the default in/out ATMU window.

3.3.3.3.2 #define SRIO_LCFBA_ADDRESS 0x3FF000000ULL

Local Configuration Space (LCS) 34 bit RapidIO base address.

3.3.3.3.3 #define SRIO_LCF_ATMU_OUT_WIN_SIZE SRIO_ATMU_OUT_WIN_1M

Local Configuration Space (LCS) outband window size.

3.3.3.3.4 #define srioGetID( ) (g_srio_id + 0) /* The + prevents usage as lvalue */

Returns the current sRIO device ID.

Returns

Device ID

3.3.3.3.5 #define srioGetAnyID( ) (g_srio_any_id + 0) /* The + prevents usage as lvalue */

Returns the sRIO device ID for an unidentified device.

Returns

Unidentified Device ID

3.3.3.3.6 #define srioIsSmallSystem( ) (g_srio_small_system == TRUE)

Returns TRUE if system is small (8 bit ID).

3.3.3.4 Typedef Documentation

3.3.3.4.1 typedef uint16_t srio_dev_id_t

sRIO device ID type

3.3.3.5 Enumeration Type Documentation

3.3.3.5.1 enum srio_flow_level_t

ATMU transaction flow levels (priority)

Warning

Errata A-006729 (for Rev1 only): Constrain inbound, non-posted requests targeting main memory to use priority 0 only. Inbound, posted requests (NWrite, SWrite) may use priority 0, 1, or 2. Constrain outbound RMan descriptor FLOWLVL to >= 2, so RMan transmit packet priorities are >= 1. Inbound requests targeting RMan may use priority 0, 1, or 2.

Enumerator

LOW_FLOW_LEVEL Low priority flow level.
MEDIUM_FLOW_LEVEL Medium priority flow level.
HIGH_FLOW_LEVEL High priority flow level.

3.3.3.5.2 enum srio_port_id_t

sRIO port numbers

3.3.3.5.3 enum srio_connectivity_t

sRIO loopback modes

Enumerator

DISABLE_LB No loopback - standard system.
DIGITAL_LB Digital internal loopback.
EXTERNAL_LB External loopback.

3.3.3.5.4 enum srio_write_type_t

ATMU outbound write types.

3.3.3.5.5 enum srio_in_target_interface_t

ATMU inbound target ports.

3.3.3.5.6 enum srio_atmu_in_win_sz_t

ATMU inbound window sizes.

Enumerator

SRIO_ATMU_IN_WIN_LAST Used for indicating last configuration.
SRIO_ATMU_IN_WIN_4K sRIO inbound ATMU window size - 4KB

SRIO_ATMU_IN_WIN_8K sRIO inbound ATMU window size - 8KB
SRIO_ATMU_IN_WIN_16K sRIO inbound ATMU window size - 16KB
SRIO_ATMU_IN_WIN_32K sRIO inbound ATMU window size - 32KB
SRIO_ATMU_IN_WIN_64K sRIO inbound ATMU window size - 64KB
SRIO_ATMU_IN_WIN_128K sRIO inbound ATMU window size - 128KB
SRIO_ATMU_IN_WIN_256K sRIO inbound ATMU window size - 256KB
SRIO_ATMU_IN_WIN_512K sRIO inbound ATMU window size - 512KB
SRIO_ATMU_IN_WIN_1M sRIO inbound ATMU window size - 1MB
SRIO_ATMU_IN_WIN_2M sRIO inbound ATMU window size - 2MB
SRIO_ATMU_IN_WIN_4M sRIO inbound ATMU window size - 4MB
SRIO_ATMU_IN_WIN_8M sRIO inbound ATMU window size - 8MB
SRIO_ATMU_IN_WIN_16M sRIO inbound ATMU window size - 16MB
SRIO_ATMU_IN_WIN_32M sRIO inbound ATMU window size - 32MB
SRIO_ATMU_IN_WIN_64M sRIO inbound ATMU window size - 64MB
SRIO_ATMU_IN_WIN_128M sRIO inbound ATMU window size - 128MB
SRIO_ATMU_IN_WIN_256M sRIO inbound ATMU window size - 256MB
SRIO_ATMU_IN_WIN_512M sRIO inbound ATMU window size - 512MB
SRIO_ATMU_IN_WIN_1G sRIO inbound ATMU window size - 1GB
SRIO_ATMU_IN_WIN_2G sRIO inbound ATMU window size - 2GB
SRIO_ATMU_IN_WIN_4G sRIO inbound ATMU window size - 4GB
SRIO_ATMU_IN_WIN_8G sRIO inbound ATMU window size - 8GB
SRIO_ATMU_IN_WIN_16G sRIO inbound ATMU window size - 16GB

3.3.3.5.7 enum srio_atmu_out_win_sz_t

ATMU outbound window sizes.

Enumerator

SRIO_ATMU_OUT_WIN_LAST Used for indicating last configuration.
SRIO_ATMU_OUT_WIN_4K sRIO outbound ATMU window size - 4KB
SRIO_ATMU_OUT_WIN_8K sRIO outbound ATMU window size - 8KB
SRIO_ATMU_OUT_WIN_16K sRIO outbound ATMU window size - 16KB
SRIO_ATMU_OUT_WIN_32K sRIO outbound ATMU window size - 32KB
SRIO_ATMU_OUT_WIN_64K sRIO outbound ATMU window size - 64KB
SRIO_ATMU_OUT_WIN_128K sRIO outbound ATMU window size - 128KB
SRIO_ATMU_OUT_WIN_256K sRIO outbound ATMU window size - 256KB
SRIO_ATMU_OUT_WIN_512K sRIO outbound ATMU window size - 512KB
SRIO_ATMU_OUT_WIN_1M sRIO outbound ATMU window size - 1MB
SRIO_ATMU_OUT_WIN_2M sRIO outbound ATMU window size - 2MB
SRIO_ATMU_OUT_WIN_4M sRIO outbound ATMU window size - 4MB
SRIO_ATMU_OUT_WIN_8M sRIO outbound ATMU window size - 8MB
SRIO_ATMU_OUT_WIN_16M sRIO outbound ATMU window size - 16MB
SRIO_ATMU_OUT_WIN_32M sRIO outbound ATMU window size - 32MB

SRIO_ATMU_OUT_WIN_64M sRIO outbound ATMU window size - 64MB
SRIO_ATMU_OUT_WIN_128M sRIO outbound ATMU window size - 128MB
SRIO_ATMU_OUT_WIN_256M sRIO outbound ATMU window size - 256MB
SRIO_ATMU_OUT_WIN_512M sRIO outbound ATMU window size - 512MB
SRIO_ATMU_OUT_WIN_1G sRIO outbound ATMU window size - 1GB
SRIO_ATMU_OUT_WIN_2G sRIO outbound ATMU window size - 2GB
SRIO_ATMU_OUT_WIN_4G sRIO outbound ATMU window size - 4GB
SRIO_ATMU_OUT_WIN_8G sRIO outbound ATMU window size - 8GB
SRIO_ATMU_OUT_WIN_16G sRIO outbound ATMU window size - 16GB
SRIO_ATMU_OUT_WIN_32G sRIO outbound ATMU window size - 32GB (requires flat address space)
SRIO_ATMU_OUT_WIN_64G sRIO outbound ATMU window size - 64GB (requires flat address space)

3.3.3.5.8 enum srio_traffic_management_t

RapidIO Traffic Management Mode.

Enumerator

TM_MODE_DISABLE Traffic Management Disabled.
TM_MODE_BASIC Traffic Management Enabled (basic)

3.3.3.6 Function Documentation

3.3.3.6.1 void srioOutboundWindowFree (srio_port_id_t *port_num*, uint32_t *win_num*)

Free outbound ATMU window.

Parameters

in	*port_num*	- sRIO port number to find window on.
in	*win_num*	- outband ATMU Window number

Warning

Don't call on SRIO_MAINTENANCE_ATMU_OUT_WIN until after maintenance accesses are finished being generated.

3.3.3.6.2 os_status srioOutboundWindowFind (srio_port_id_t *port_num*, uint32_t * *win_num*)

Get a free outbound ATMU window.

Parameters

in	*port_num*	- sRIO port number to find window on.
out	*win_num*	- Window number.

Returns

OS Status

3.3.3.6.3 void srioOutboundWindowEnable (srio_port_id_t *port_num,* uint32_t *win_num*)

Enable outbound ATMU window.

Parameters

in	*port_num*	- sRIO port number to find window on.
in	*win_num*	- Window number.

3.3.3.6.4 void srioOutboundWindowDisable (srio_port_id_t *port_num,* uint32_t *win_num*)

Disable outbound ATMU window.

Parameters

in	*port_num*	- sRIO port number to find window on.
in	*win_num*	- Window number.

3.3.3.6.5 os_status srioOutboundWindowOpen (srio_port_id_t *port_num,* srio_out_win_config_t * *srio_out_win_config*)

Configure an outbound ATMU window.

Parameters

in	*port_num*	- sRIO port number to find window on.
in	*srio_out_win_config*	- Window configuration.

Returns

OS Status

3.3.3.6.6 os_status srioInboundWindowFind (srio_port_id_t *port_num*, uint32_t * *win_num*)

Get a free inbound ATMU window.

Parameters

in	*port_num*	- sRIO port number to find window on.
out	*win_num*	- Window number

Returns

OS Status

3.3.3.6.7 void srioInboundWindowFree (srio_port_id_t *port_num*, uint32_t *win_num*)

Free an inbound ATMU window.

Parameters

in	*port_num*	- sRIO port number to find window on.
in	*win_num*	- Window number

3.3.3.6.8 void srioInboundWindowEnable (srio_port_id_t *port_num*, uint32_t *win_num*)

Enable inbound ATMU window.

Parameters

in	*port_num*	- sRIO port number to find window on.
in	*win_num*	- Window number

3.3.3.6.9 void srioInboundWindowDisable (srio_port_id_t *port_num*, uint32_t *win_num*)

Disable inbound ATMU window.

Parameters

in	*port_num*	- sRIO port number to find window on.
in	*win_num*	- Window number

3.3.3.6.10 os_status srioInboundWindowOpen (srio_port_id_t *port_num*, srio_in_win_config_t * *srio_in_win_config*)

Configure an inbound ATMU window.

Parameters

in	*port_num*	- sRIO port number to find window on.
in	*srio_in_win_config*	- Window configuration

Returns

OS Status

3.3.3.6.11 os_status srioMaintenanceAccess (os_dma_handle *dma*, uint32_t *atmu_win_id*, srio_port_id_t *port_num*, uint64_t *data*, uint32_t *offset*, uint32_t *access*, uint32_t *ch*, int *mode*)

Generate sRIO maintenance access.

Parameters

in	*dma*	- Pointer to OCeaN DMA (returned by ocnDmaDeviceOpen()
in	*atmu_win_id*	- Outband window number
in	*port_num*	- sRIO port number
in	*data*	- Address for the maintenance access to write/read to/from
in	*offset*	- Offset of register in CSR to access
in	*access*	- size of maintenance transaction
in	*ch*	- OCeaN DMA channel to use
in	*mode*	- Set to OCN_DMA_WRITE or OCN_DMA_READ

Returns

OS Status.

3.3.3.6.12 os_status srioMaintenanceTargetSet (srio_port_id_t *port_num*, uint32_t *atmu_win_id*, srio_dev_id_t *target_id*, uint8_t *hop_count*, uint32_t *offset*)

Modify the setting of a given maintenance ATMU windows.

Parameters

in	*port_num*	- sRIO port number
in	*atmu_win_id*	- sRIO outband ATMU window number
in	*target_id*	- sRIO device ID to access with maintenance
in	*hop_count*	- Hop count to sRIO device
in	*offset*	- Configuration address

Returns

OS Status.

3.3.3.6.13 void srioMaintenanceAtmuFree (srio_port_id_t *port_num*, uint32_t *atmu_win_id*)

Free the ATMU windows allocated by srioMaintenanceAtmuOpen()

Parameters

in	*port_num*	- sRIO port number to close maintnenance ATMU windows
in	*atmu_win_id*	- sRIO outband ATMU window number

Returns

OS Status.

3.3.3.6.14 os_status srioMaintenanceAtmuOpen (srio_port_id_t *port_num*, srio_maint_win_config_t * *srio_main_win_config*)

Open sRIO maintenance ATMU window. The size of the ATMU window will be 64KB.

Parameters

in	*port_num*	- sRIO port number.
in	*srio_main_win_config*	- sRIO maintenance configuration structure

Returns

OS Status.

3.3.3.6.15 os_status srioTrafficManagementSupport (srio_traffic_management_t *tm_mode*)

Modify the setting of the traffic management support for all ports

Parameters

in	*tm_mode*	- traffic management mode

Returns

OS Status.

3.3.3.6.16 os_status srioDeviceIdGetByAttr (srio_device_t * *device*)

Return a sRIO device by its attributes.

Parameters

device	- (in/out) device structure with attributes other than dev_id.

Returns

OS status.

3.3.3.6.17 void srioDeviceGetByIndex (srio_device_t ** *device*, srio_port_id_t *srio_port*, uint16_t *index*)

Return a sRIO device by its index in the manager.

Parameters

in	*index*	- Index of the requested sRIO device.
in	*srio_port*	- sRIO port number.
out	*device*	- Pointer to a device structure.

3.3.3.6.18 void srioDeviceGetById (srio_device_t ** *device*, srio_port_id_t *srio_port*, srio_dev_id_t *dev_id*)

Return a sRIO device by its device id.

Parameters

in	*dev_id*	- Device ID of requested sRIO device.
in	*srio_port*	- sRIO port number.
out	*device*	- Pointer to a device structure.

3.3.3.6.19 uint16_t srioNumDiscoverdDevicesGet ( )

Returns the number of discovered sRIO.

Returns

Number of discovered devices.

3.3.3.6.20 void srioClearPortErrors (srio_port_id_t *port*)

Clears errors on the port

Parameters

in	*port*	- Lynx PHY port number to recover

Returns

OS Status

Warning

Not called by the OS directly. User should call in sRIO error ISR.

3.4 I2C Module API

3.4.1 Overview

Modules

- I2C Initialization
- runtime API
- I2C B4860-specific definitions

Macros

- #define RETRY_DEF 0x1000
- #define I2C_TIMEOUT 0xA00
- #define I2C_CMD_SCAN_BUS (0x00000100 | CIO_LLD_COMMAND)
- #define I2C_CMD_STOP_DEVICE (0x00000200 | CIO_LLD_COMMAND)
- #define I2C_CMD_DEVICE_BUSY (0x00000400 | CIO_LLD_COMMAND)
- #define I2C_CMD_FREE_STUCK_BUS (0x00000800 | CIO_LLD_COMMAND)
- #define I2C_CMD_SET_FREQUENCY_DEVIDER (0x00001000 | CIO_LLD_COMMAND)
- #define I2C_CMD_SET_DIGITAL_FILTER_SAMPLING_RATE (0x00002000 | CIO_LLD_C↩ OMMAND)
- #define I2C_CMD_SET_TIMEOUT (0x00004000 | CIO_LLD_COMMAND)
- #define I2C_FREQ_DEF 400
- #define I2C_DFSR_DEF 0x10

3.4.2 Macro Definition Documentation

3.4.2.1 #define RETRY_DEF 0x1000

Default number for pending on an interrupt or a status bit.

3.4.2.2 #define I2C_TIMEOUT 0xA00

I2C Timeout.

3.4.2.3 #define I2C_CMD_SCAN_BUS (0x00000100 | CIO_LLD_COMMAND)

Scan the bus for i2c devices (param: A pointer to i2c_scan_cmd_params_t type)

3.4.2.4 #define I2C_CMD_STOP_DEVICE (0x00000200 | CIO_LLD_COMMAND)

Stop I2C device (param: NULL)

3.4.2.5 #define I2C_CMD_DEVICE_BUSY (0x00000400 | CIO_LLD_COMMAND)

Check if the I2C device is busy (param: NULL)

3.4.2.6 #define I2C_CMD_FREE_STUCK_BUS (0x00000800 | CIO_LLD_COMMAND)

Free the bus (param:NULL)

3.4.2.7 #define I2C_CMD_SET_FREQUENCY_DEVIDER (0x00001000 | CIO_LLD_COMMAND)

Set frequency devider register (param:frequency devider value)

3.4.2.8 #define I2C_CMD_SET_DIGITAL_FILTER_SAMPLING_RATE (0x00002000 | CIO_LLD_COMMAND)

Set digital filter sampling register (param:digital filter sampling rate value)

3.4.2.9 #define I2C_CMD_SET_TIMEOUT (0x00004000 | CIO_LLD_COMMAND)

Set the time interval between polling the interrupt bit (param:new timeout to wait)

3.4.2.10 #define I2C_FREQ_DEF 400

Default frequency in KHZ.

3.4.2.11 #define I2C_DFSR_DEF 0x10

Default value to digital filter sampling rate.

3.4.3 I2C Initialization

3.4.3.1 Overview

```
I2C initialize routines.
```

Data Structures

- struct i2c_init_params_t
- struct i2c_open_params_t

Functions

- os_status i2cInitialize (i2c_init_params_t *init_params)

3.4.3.2 Data Structure Documentation

3.4.3.2.1 struct i2c_init_params_t

I2C parameter structure.

Hold all the parameter that are needed to initialize the I2C; Include parameters that the driver must get from the user, and other parameters that gets default value.

Data Fields

- i2c_id_t i2c_id
- uint32_t retry
- uint16_t frequency
- uint8_t dfsr
- uint8_t owner_core

3.4.3.2.1.1 Field Documentation

3.4.3.2.1.1.1 i2c_id_t i2c_init_params_t::i2c_id

Base address of I2C memory map.

3.4.3.2.1.1.2 uint32_t i2c_init_params_t::retry

How many times should we retry polling before aborting.

3.4.3.2.1.1.3 uint16_t i2c_init_params_t::frequency

Frequency of SCL of the device (in KHZ)

3.4.3.2.1.1.4 uint8_t i2c_init_params_t::dfsr

Digital filter sampling rate.

3.4.3.2.1.1.5 uint8_t i2c_init_params_t::owner_core

Core ID which owns this I2C device.

3.4.3.2.2 struct i2c_open_params_t

I2C Controller opening parameters.

This structure should be used when calling osCioDeviceOpen() with the I2C device handle.

Data Fields

- void(∗ error_cb)(uint32_t events)

3.4.3.2.2.1 Field Documentation

3.4.3.2.2.1.1 void(∗ i2c_open_params_t::error_cb)(uint32_t events)

Callback for handling general errors.

3.4.3.3 Function Documentation

3.4.3.3.1 os_status i2cInitialize (i2c_init_params_t ∗ *init_params*)

I2C initialization function.

Parameters

in	*init_params*	- I2C initialization parameters structure

Return values

OS_SUCCESS	: I2C initialized successfully.
Error	status, encoded in os_error.h, for other errors.

Warning

None.

3.4.4 runtime API

3.4.4.1 Overview

runtime API definitions.

Data Structures

- struct i2c_scan_cmd_params_t

Enumerations

- enum i2c_trans_mode { I2C_TX, I2C_RX }
- enum i2c_next_trans_action { I2C_NORMAL, I2C_REPEATED_START, I2C_SKIP_ADDRESS }

3.4.4.2 Data Structure Documentation

3.4.4.2.1 struct i2c_scan_cmd_params_t

I2C scan command parameter structure.

Data Fields

- uint8_t * addresses
- int(* print_func)(const char *,...)
- uint8_t start_address
- uint8_t end_address

3.4.4.2.1.1 Field Documentation

3.4.4.2.1.1.1 uint8_t* i2c_scan_cmd_params_t::addresses

A buffer to be filled with i2c device addresses in ascending order.

The byte after the last device address will be NULL. Must be at least of size (end_address - start_address + 1) or NULL if no filling is desired.

3.4.4.2.1.1.2 int(* i2c_scan_cmd_params_t::print_func)(const char *,...)

A printing function.

If no printing is desired, must be NULL

3.4.4.2.1.1.3 uint8_t i2c_scan_cmd_params_t::start_address

1st I2C address to scan

3.4.4.2.1.1.4 uint8_t i2c_scan_cmd_params_t::end_address

last I2C address to scan

3.4.4.3 Enumeration Type Documentation

3.4.4.3.1 enum i2c_trans_mode

transaction mode, transmit/receive transaction

Enumerator

I2C_TX Transmit.
I2C_RX Receive.

3.4.4.3.2 enum i2c_next_trans_action

Action required at the next transmit/receive transaction.

Enumerator

I2C_NORMAL Regular transaction; STOP at the end, START at the next trans with address phase.
I2C_REPEATED_START No STOP at the end, START at the next trans with address phase.
I2C_SKIP_ADDRESS No STOP at the end, no START at the next trans nor address phase.

3.4.5 I2C B4860-specific definitions

3.4.5.1 Overview

Macros

- #define I2C_NUM_OF_DEVICES 4
- #define I2C_1_DEVICE_NAME "i2c1"
- #define I2C_2_DEVICE_NAME "i2c2"
- #define I2C_3_DEVICE_NAME "i2c3"
- #define I2C_4_DEVICE_NAME "i2c4"

Enumerations

- enum i2c_id_t

3.4.5.2 Macro Definition Documentation

3.4.5.2.1 #define I2C_NUM_OF_DEVICES 4

Number of I2C devices.

3.4.5.2.2 #define I2C_1_DEVICE_NAME "i2c1"

I2C1 Device Name.

3.4.5.2.3 #define I2C_2_DEVICE_NAME "i2c2"

I2C2 Device Name.

3.4.5.2.4 #define I2C_3_DEVICE_NAME "i2c3"

I2C3 Device Name.

3.4.5.2.5 #define I2C_4_DEVICE_NAME "i2c4"

I2C4 Device Name.

3.4.5.3 Enumeration Type Documentation

3.4.5.3.1 enum i2c_id_t

Enumeration of I2C ID's.

3.5 MAPLE-B3 Module API

3.5.1 Overview

Modules

- Maple API
- Maple PE Generic API
- Maple CRCPE API
- Maple TVPE API
- Maple FTPE API
- Maple EQPE API
- Maple DEPE API
- Maple PUSCH API
- Maple PUFFT API
- Maple PDSCH API
- Maple TCPE API
- Maple Trace Events API

3.5.2 Maple API

3.5.2.1 Overview

```
MAPLE device API
```

Modules

- Maple Initialization
- Maple Workarounds API

MAPLE MMU attributes

- #define MAPLE_MMU_REGA_SHIFT (0)
- #define MAPLE_MMU_REGC_SHIFT (16)
- #define MAPLE_MMU_REGM_SHIFT (32)

- #define MAPLE_MMU_ATTR_FLEX_SEGMENT MMU_ATTR_BIT_SET(MAPLE_MMU_REGA_SHIFT, 10)
- #define MAPLE_MMU_ATTR_WPERM MMU_ATTR_BIT_SET(MAPLE_MMU_REGA_SHIFT, 1)
- #define MAPLE_MMU_ATTR_RPERM MMU_ATTR_BIT_SET(MAPLE_MMU_REGA_SHIFT, 2)
- #define MAPLE_MMU_ATTR_ENABLE MMU_ATTR_BIT_SET(MAPLE_MMU_REGA_SHIFT, 0)
- #define MAPLE_MMU_ATTR_DID(id) MMU_ATTR_FIELD_SET(MAPLE_MMU_REGC_SHIFT, 8, id)
- #define MAPLE_MMU_CHB_ATTR_ETYPE MMU_ATTR_BIT_SET(MAPLE_MMU_REGM_SHIFT, 5)
- #define MAPLE_MMU_CHB_ATTR_ENHANCED MMU_ATTR_BIT_SET(MAPLE_MMU_REGM_SHIFT, 6)
- #define MAPLE_MMU_CHB_ATTR_LIODN(attr) MMU_ATTR_FIELD_SET(MAPLE_MMU_REGM_SHIFT, 7, attr)
- #define MAPLE_MMU_ATTR_CACHEABLE MMU_ATTR_BIT_SET(MAPLE_MMU_REGM_SHIFT, 21)
- #define MAPLE_MMU_ATTR_GAURDED MMU_ATTR_BIT_SET(MAPLE_MMU_REGM_SHIFT, 22)
- #define MAPLE_MMU_ATTR_COHERENT MMU_ATTR_BIT_SET(MAPLE_MMU_REGM_SHIFT, 23)
- #define MAPLE_MMU_ATTR_WRITETHROUGH MMU_ATTR_BIT_SET(MAPLE_MMU_REGM_SHIFT, 24)
- #define MAPLE_MMU_ATTR_LEAST_RU MMU_ATTR_BIT_SET(MAPLE_MMU_REGM_SHIFT, 25)
- #define MAPLE_MMU_ATTR_L2_PARTITION_ID(id) MMU_ATTR_FIELD_SET(MAPLE_MMU_REGM_SHIFT, 26, id)

3.5.2.2 Macro Definition Documentation

3.5.2.2.1 #define MAPLE_MMU_REGA_SHIFT (0)

Bit shift of MAPLE MMU REGA in the os_mmu_attr.

3.5.2.2.2 #define MAPLE_MMU_REGC_SHIFT (16)

Bit shift of MAPLE MMU REGC in the os_mmu_attr.

3.5.2.2.3 #define MAPLE_MMU_REGM_SHIFT (32)

Bit shift of MAPLE MMU REGM in the os_mmu_attr.

3.5.2.2.4 #define MAPLE_MMU_ATTR_FLEX_SEGMENT MMU_ATTR_BIT_SET(MAPLE_MMU_REGA_SHIFT, 10)

Flexible segment; More flexible size restrictions for small segments.

3.5.2.2.5 #define MAPLE_MMU_ATTR_WPERM MMU_ATTR_BIT_SET(MAPLE_MMU_REGA_SHIFT, 1)

Write permitted; If there is no read or write permission and protection is enabled a violation occurs.

3.5.2.2.6 #define MAPLE_MMU_ATTR_RPERM MMU_ATTR_BIT_SET(MAPLE_MMU_REGA_SHIFT, 2)

Read permitted; If there is no read or write permission and protection is enabled a violation occurs.

3.5.2.2.7 #define MAPLE_MMU_ATTR_ENABLE MMU_ATTR_BIT_SET(MAPLE_MMU_REGA_SHIFT, 0)

Enables to enable mmu segment during segment update.

3.5.2.2.8 #define MAPLE_MMU_ATTR_DID(*id*) MMU_ATTR_FIELD_SET(MAPLE_MMU_REGC_SHIFT, 8, id)

Data task ID.

3.5.2.2.9 #define MAPLE_MMU_CHB_ATTR_ETYPE MMU_ATTR_BIT_SET(MAPLE_MMU_REGM_SHIFT, 5)

CHB etype attribute.

3.5.2.2.10 #define MAPLE_MMU_CHB_ATTR_ENHANCED MMU_ATTR_BIT_SET(MAPLE_MMU_REGM_SHIFT, 6)

CHB enhanced attribute.

3.5.2.2.11 #define MAPLE_MMU_CHB_ATTR_LIODN(*attr*) MMU_ATTR_FIELD_SET(MAPLE_MMU_REGM_SHIFT, 7, attr)

CHB Logical IO Device Number attribute.

3.5.2.2.12 #define MAPLE_MMU_ATTR_CACHEABLE MMU_ATTR_BIT_SET(MAPLE_MMU_↩ REGM_SHIFT, 21)

This memory segment is cacheable.

3.5.2.2.13 #define MAPLE_MMU_ATTR_GAURDED MMU_ATTR_BIT_SET(MAPLE_MMU_R↩ EGM_SHIFT, 22)

Data Guarded Segment.

3.5.2.2.14 #define MAPLE_MMU_ATTR_COHERENT MMU_ATTR_BIT_SET(MAPLE_MMU_↩ REGM_SHIFT, 23)

Data Coherent Memory Segment.

3.5.2.2.15 #define MAPLE_MMU_ATTR_WRITETHROUGH MMU_ATTR_BIT_SET(MAPLE_M↩ MU_REGM_SHIFT, 24)

This memory segment uses the write through policy.

3.5.2.2.16 #define MAPLE_MMU_ATTR_LEAST_RU MMU_ATTR_BIT_SET(MAPLE_MMU_R↩ EGM_SHIFT, 25)

This memory segment should be considered as least recently used.

3.5.2.2.17 #define MAPLE_MMU_ATTR_L2_PARTITION_ID(*id*) MMU_ATTR_FIELD_SET(M↩ APLE_MMU_REGM_SHIFT, 26, id)

KIBO attribute which indicates the L2 memory partitioning ID.

3.5.2.3 Maple Initialization

3.5.2.3.1 Overview

```
MAPLE device initialization API
```

Data Structures

- struct maple_depe_wa_params_t
- struct maple_malloc_t
- struct maple_mmu_seg_t
- struct maple_mmu_seg_update_t
- struct maple_mmu_err_t

- struct maple_mmu_init_param_t
- union maple_version_t
- struct maple_open_params_t
- struct maple_init_params_t

Macros

- #define MAPLE_B3LW_UCODE ((void *)&maple3lw_code[0])
- #define MAPLE_B3W_UCODE ((void *)&maple3w_code[0])
- #define MAPLE_B3W_INIT maple3w_init
- #define MAPLE_B3LW_INIT maple3lw_init
- #define MAPLE_FLG_DEFAULT 0

Enumerations

- enum maple_pcr_opcodes_t { ,
  MAPLE_PARSE_TVPE_BD = 0x2 , MAPLE_PARSE_FTPE_2_BD = 0x5, MAPLE_PARSE_↩
  DEPE_BD = 0x6 ,
  MAPLE_PARSE_EQPE_BD = 0x8, MAPLE_PARSE_CONVPE_BD = 0x9, MAPLE_PARSE_↩
  PUSCH_EDF_BD = 0xa,
  MAPLE_PARSE_PDSCH_EDF_BD = 0xb, MAPLE_PDSCH_EXT_SYM_START = 0xc , MA↩
  PLE_PUSCH_USER_NEIGHBOR_READY = 0x14,
  MAPLE_PDSCH_EXT_SYM_START_1 = 0x20, MAPLE_PDSCH_EXT_SYM_START_2 = 0x21, MAPLE_PDSCH_EXT_SYM_START_3 = 0x22,
  MAPLE_PDSCH_LATE_PARAMS_READY_0 = 0x23, MAPLE_PDSCH_LATE_PARAMS_↩
  READY_1 = 0x24, MAPLE_PDSCH_LATE_PARAMS_READY_2 = 0x25,
  MAPLE_PDSCH_LATE_PARAMS_READY_3 = 0x26 }
- enum maple_dev_id_t {
  MAPLE_DEV_ID_NONE = 0xFF, MAPLE_LW_ID_0 = 1, MAPLE_LW_ID_1 = 2,
  MAPLE_W_ID_2 = 3 }

Functions

- INLINE long mapleDirectRead (void *addr, void *data, uint32_t size, uint32_t dev_id, uint32_t param)
- INLINE long mapleDirectWrite (void *addr, void *data, uint32_t size, uint32_t dev_id, uint32_t param)
- os_status mapleInitialize (struct maple_init_params_s *init_params, unsigned int num_devices)

Variables

- unsigned long int maple3lw_code [ ]
- unsigned long int maple3w_code [ ]

MAPLE Error Flags indications

- enum {
  MAPLE_MMU_MULTI_HIT_ERR = 0x00000000, MAPLE_MMU_MISS_ERR = 0x00000010,
  MAPLE_MMU_READ_ERR = 0x00000020,
  MAPLE_MMU_WRITE_ERR = 0x00000030 }
- enum maple_pspicer2_flag_t {
  MAPLE3LW_ECC_TV_DRE = 0x80000000, MAPLE3LW_ECC_TV_NILL = 0x40000000, M↩
  APLE3LW_ECC_TV_HARQ = 0x20000000,
  MAPLE3LW_ECC_FTPE0 = 0x10000000, MAPLE3LW_ECC_FTPE1 = 0x08000000, MAPL↩
  E3LW_ECC_FTPE2 = 0x04000000,
  MAPLE3LW_ECC_PUPE = 0x02000000, MAPLE3LW_ECC_PDPE = 0x01000000, MAPLE3↩
  LW_ECC_EQPE = 0x00800000,
  MAPLE3W_ECC_FTPE0 = 0x80000000, MAPLE3W_ECC_FTPE1 = 0x40000000, MAPLE3↩
  W_ECC_TCPE = 0x20000000,
  MAPLE3W_ECC_CRPEULB0 = 0x10000000, MAPLE3W_ECC_CRPEULB1 = 0x08000000,
  MAPLE3W_ECC_CRPE_DL = 0x02000000,
  MAPLE_ECC_RISC3_TRACE_BUFFER = 0x00008000, MAPLE_ECC_RISC2_TRACE_BUF↩
  FER = 0x00004000, MAPLE_ECC_RISC1_TRACE_BUFFER = 0x00002000,
  MAPLE_ECC_RISC0_TRACE_BUFFER = 0x00001000, MAPLE_ECC_HW_SCHEDULER_↩
  MEM1 = 0x00000800, MAPLE_ECC_HW_SCHEDULER_MEM0 = 0x00000400,
  MAPLE_ECC_DMA3_CMD_MEM1 = 0x00000200, MAPLE_ECC_DMA2_CMD_MEM1 =
  0x00000100, MAPLE_ECC_DMA1_CMD_MEM1 = 0x00000080,
  MAPLE_ECC_DMA0_CMD_MEM1 = 0x00000040, MAPLE_ECC_DMA3_CMD_MEM0 =
  0x00000020, MAPLE_ECC_DMA2_CMD_MEM0 = 0x00000010,
  MAPLE_ECC_DMA1_CMD_MEM0 = 0x00000008, MAPLE_ECC_DMA0_CMD_MEM0 =
  0x00000004, MAPLE_ECC_DRAM = 0x00000002,
  MAPLE_ECC_IRAM = 0x00000001 }
- #define MAPLE_SYS_ERR 0x80000000
- #define MAPLE_MMU_ERR 0x10000000
- #define MAPLE_ECC_ERR 0x00010000

MAPLE Device Control Commands.

- #define MAPLE_CMD_MALLOC (0x00000300 | COP_LLD_COMMAND)
- #define MAPLE_CMD_GET_UCODE_VERSION (0x00000400 | COP_LLD_COMMAND)
- #define MAPLE_CMD_PCR_ACTIVATE_WITH_POLL (0x00000500 | COP_LLD_COMMA↩
  ND)
- #define MAPLE_CMD_PCR_ACTIVATE_NO_POLL (0x00000600 | COP_LLD_COMMAND)
- #define MAPLE_CMD_MMU_SEGMENT_FIND (0x00000700 | COP_LLD_COMMAND)
- #define MAPLE_CMD_MMU_SEGMENT_UPDATE (0x00000800 | COP_LLD_COMMAND)
- #define MAPLE_CMD_MMU_SEGMENT_ENABLE (0x00000900 | COP_LLD_COMMAND)
- #define MAPLE_CMD_MMU_SEGMENT_DISABLE (0x00000A00 | COP_LLD_COMMAND)
- #define MAPLE_CMD_MMU_ENABLE (0x00000B00 | COP_LLD_COMMAND)
- #define MAPLE_CMD_MMU_DISABLE (0x00000C00 | COP_LLD_COMMAND)
- #define MAPLE_CMD_MMU_PROTECT_DISABLE (0x00000D00 | COP_LLD_COMMAND)
- #define MAPLE_CMD_MMU_PROTECT_ENABLE (0x00000E00 | COP_LLD_COMMAND)
- #define MAPLE_CMD_MMU_ERROR_DETECT (0x00000F00 | COP_LLD_COMMAND)

- #define MAPLE_CMD_SOFT_RESET (0x00001000 | COP_LLD_COMMAND)
- #define MAPLE_CMD_SET_TRACE_EVENTS (0x00001100 | COP_LLD_COMMAND)
- #define MAPLE_CMD_CLEAR_TRACE_BUFFER (0x00001200 | COP_LLD_COMMAND)
- #define MAPLE_CMD_GET_TRACE_BUFFER (0x00001300 | COP_LLD_COMMAND)
- #define MAPLE_CMD_GET_TRACE_BUFFER_WRITE_PTR (0x00001400 | COP_LLD_CO↩ MMAND)
- #define MAPLE_CMD_ACTIVATE_PDSCH_ACK_INT (0x00001500 | COP_LLD_COMMA↩ ND)
- #define MAPLE_CMD_DISABLE_ECC_PROTECTION (0x00001600 | COP_LLD_COMMAN↩ D)
- #define MAPLE_CMD_ENABLE_ECC_PROTECTION (0x00001700 | COP_LLD_COMMAND)
- #define MAPLE_CMD_DEPE_WA (0x00001800 | COP_LLD_COMMAND)
- #define MAPLE_CMD_PRE_POST_TASK_ID (0x00001900 | COP_LLD_COMMAND)

MAPLE Device Names used for COP registration

- #define MAPLE_0_NAME "MPL00"
- #define MAPLE_1_NAME "MPL01"
- #define MAPLE_2_NAME "MPL02"

Flags for maple3lw_dev_init_params_t.mode1

- enum { MAPLE_B3LW_INIT_INTERNAL_AXI_PE_ALC = 0x00000000, MAPLE_B3LW_IN↩ IT_EXTERNAL_AXI_PE_ALC = 0x10000000 }
- enum { MAPLE_B3LW_INIT_AXI_PR_SCHEME0 = 0x00000000, MAPLE_B3LW_INIT_AX↩ I_PR_SCHEME1 = 0x01000000, MAPLE_B3LW_INIT_AXI_PR_SCHEME2 = 0x02000000 }
- enum maple_b3lw_init_power_sch_pe_t { MAPLE_B3LW_INIT_POWER_SCHEME_PE_FTPE0 = 2, MAPLE_B3LW_INIT_POWER_S↩ CHEME_PE_FTPE1 = 4, MAPLE_B3LW_INIT_POWER_SCHEME_PE_FTPE2 = 6, MAPLE_B3LW_INIT_POWER_SCHEME_PE_TVPE = 8, MAPLE_B3LW_INIT_POWER_S↩ CHEME_PE_EQPE = 12 }
- enum maple_b3lw_init_power_sch_t { MAPLE_B3LW_INIT_POWER_SCHEME_AUTO = 0, MAPLE_B3LW_INIT_POWER_SCHEME_ON = 1, MAPLE_B3LW_INIT_POWER_SCHEM↩ E_OFF = 2 }
- INLINE uint32_t MAPLE_B3LW_INIT_PEx_POWER_SCHEME (uint8_t axi_idx, uint8_t prior- ity)
- INLINE uint32_t MAPLE_B3LW_INIT_AXIx_DEFAULT_PRIORITY (maple_b3lw_init_↩ power_sch_pe_t pe, maple_b3lw_init_power_sch_t pwr_sch)

Flags for maple3lw_dev_init_params_t.mode2

- enum maple_b3lw_init_axi_aloc_bus_type_t { MAPLE_B3LW_INIT_AXI_ALOC_READ_BU↩ S = 0, MAPLE_B3LW_INIT_AXI_ALOC_WRITE_BUS = 2 }
- enum maple_b3lw_init_axi_aloc_pe_t { MAPLE_B3LW_INIT_AXI_ALOC_PE_TVPE = 28, MAPLE_B3LW_INIT_AXI_ALOC_PE_↩ FTPE0 = 24, MAPLE_B3LW_INIT_AXI_ALOC_PE_FTPE1 = 20, MAPLE_B3LW_INIT_AXI_ALOC_PE_FTPE2 = 16, MAPLE_B3LW_INIT_AXI_ALOC_PE_↩

CRC_DEPE = 12, MAPLE_B3LW_INIT_AXI_ALOC_PE_EQPE = 8, MAPLE_B3LW_INIT_AXI_ALOC_PE_PDSCH = 4, MAPLE_B3LW_INIT_AXI_ALOC_PE_PUSCH = 0 }

- enum maple_b3lw_init_axi_aloc_t { MAPLE_B3LW_INIT_ALOC_AXI0 = 0, MAPLE_B3LW_INIT_ALOC_AXI1 = 1, MAPLE_B3LW_INIT_ALOC_AXI2 = 2, MAPLE_B3LW_INIT_ALOC_AXI3 = 3 }
- INLINE uint32_t MAPLE_B3LW_INIT_AXI_PE_ALC_PARAM (maple_b3lw_init_axi_aloc_pe_t pe, maple_b3lw_init_axi_aloc_t axi_aloc, maple_b3lw_init_axi_aloc_bus_type_t type)

3.5.2.3.2 Data Structure Documentation

3.5.2.3.2.1 struct maple_depe_wa_params_t

MAPLEB3 Struct for WA for DEPE ERR007563.

Data Fields

- uint32_t bd [16]

3.5.2.3.2.1.1 Field Documentation

3.5.2.3.2.1.2 uint32_t maple_depe_wa_params_t::bd[16]

BD to write.

3.5.2.3.2.2 struct maple_malloc_t

MAPLE Malloc structure - use when calling on mapleCtrl with MAPLE_CMD_MALLOC.

Data Fields

- uint32_t size
- void ** addr
- uint32_t alignment

3.5.2.3.2.2.1 Field Documentation

3.5.2.3.2.2.2 uint32_t maple_malloc_t::size

Size of memory to allocate.

3.5.2.3.2.2.3 void** maple_malloc_t::addr

Address allocated.

3.5.2.3.2.2.4 uint32_t maple_malloc_t::alignment

Alignment requirements.

3.5.2.3.2.3 struct maple_mmu_seg_t

MAPLE MMU segment structure - use for maple MMU initialization.

Data Fields

- os_mmu_attr attr
- os_virt_ptr virt_addr
- os_phys_ptr phys_addr
- uint32_t size

3.5.2.3.2.3.1 Field Documentation

3.5.2.3.2.3.2 os_mmu_attr maple_mmu_seg_t::attr

Maple MMU attributes.

3.5.2.3.2.3.3 os_virt_ptr maple_mmu_seg_t::virt_addr

Virtual start address of MMU segment.

3.5.2.3.2.3.4 os_phys_ptr maple_mmu_seg_t::phys_addr

Physical start address of MMU segment.

3.5.2.3.2.3.5 uint32_t maple_mmu_seg_t::size

Size of the segment; size = 0 is saved for 4G segments.

3.5.2.3.2.4 struct maple_mmu_seg_update_t

Maple MMU segment update parameters.

Data Fields

- os_mmu_segment_handle descriptor
- maple_mmu_seg_t attributes

3.5.2.3.2.4.1 Field Documentation

3.5.2.3.2.4.2 os_mmu_segment_handle maple_mmu_seg_update_t::descriptor

Maple MMU descriptor number.

3.5.2.3.2.4.3 maple_mmu_seg_t maple_mmu_seg_update_t::attributes

Maple MMU segment parameters.

3.5.2.3.2.5 struct maple_mmu_err_t

Maple MMU error detect output parameters.

Data Fields

- os_virt_ptr err_addr
- uint32_t err_type
- uint32_t descritptors
- bool err_event

3.5.2.3.2.5.1 Field Documentation

3.5.2.3.2.5.2 os_virt_ptr maple_mmu_err_t::err_addr

Violation virtual address.

3.5.2.3.2.5.3 uint32_t maple_mmu_err_t::err_type

Error type.

Use MAPLE_MMU_MULTI_HIT_ERR, MAPLE_MMU_MISS_ERR, MAPLE_MMU_READ_ERR, MAPLE_MMU_WRITE_ERR

3.5.2.3.2.5.4 uint32_t maple_mmu_err_t::descritptors

Mask of enabled descriptors.

3.5.2.3.2.5.5 bool maple_mmu_err_t::err_event

Indicates on MMU error event.

Set by HW and cleared by mapleSysErrIsr

3.5.2.3.2.6 struct maple_mmu_init_param_t

Maple MMU enable parameters; Use it for MAPLE_CMD_MMU_ENABLE.

Data Fields

- os_virt_ptr trash_vaddr
- uint8_t steering_bits

3.5.2.3.2.6.1 Field Documentation

3.5.2.3.2.6.2 os_virt_ptr maple_mmu_init_param_t::trash_vaddr

Virtual address used by the MAPLE-B3 for AXI initiators accesses which result in internal MMU error.

3.5.2.3.2.6.3 uint8_t maple_mmu_init_param_t::steering_bits

Trash Address Steering.

Use the following flags to set steering_bits field MAPLE_STEERING_DSP_CLUSTER_0, MAPLE_S↩ TEERING_DSP_CLUSTER_1, MAPLE_STEERING_DSP_CLUSTER_2, MAPLE_STEERING_CHB, MAPLE_STEERING_MAPLE_TRG

3.5.2.3.2.7 union maple_version_t

MAPLE Version structure - use when calling on mapleCtrl with MAPLE_CMD_GET_UCODE_VERS↩ ION.

3.5.2.3.2.7.1 Field Documentation

3.5.2.3.2.7.2 uint32_t maple_version_t::maple_w

If set the uCode is targeted for MAPLE-B3W instance.

3.5.2.3.2.7.3 uint32_t maple_version_t::dev_fw

Target Device firmware.

3.5.2.3.2.7.4 uint32_t maple_version_t::ucode_vesion

This field describes the ucode and API version used to initiate the MAPLE.

3.5.2.3.2.8 struct maple_open_params_t

MAPLE Device Open parameters.

Data Fields

- void(∗ error_callback)(uint32_t device_id, uint32_t error_type)
- bool config_mbus_mmu
- void ∗ mbus_base
- bool config_sbus_mmu
- void ∗ sbus_base
- void ∗ dev_init_params
- void(∗ maple_init)(void ∗input)

3.5.2.3.2.8.1 Field Documentation

3.5.2.3.2.8.2 void(∗ maple_open_params_t::error_callback)(uint32_t device_id, uint32_t error_type)

Callback for errors.

Will be called with for ALL errors

3.5.2.3.2.8.3 bool maple_open_params_t::config_mbus_mmu

Boolean flag - should a data MMU segment (PRAM) be configured.

3.5.2.3.2.8.4 void∗ maple_open_params_t::mbus_base

MBUS base address as seen on host.

Only used if config_mbus_mmu is set to TRUE

3.5.2.3.2.8.5 bool maple_open_params_t::config_sbus_mmu

Boolean flag - should a data MMU segment (PSIF) be configured.

Set to FALSE if MAPLE on device and in general CCSR address space

3.5.2.3.2.8.6 void* maple_open_params_t::sbus_base

SBUS base address as seen on host.

Only used if config_sbus_mmu is set to TRUE

3.5.2.3.2.8.7 void* maple_open_params_t::dev_init_params

Initialization parameters for MAPLE device initialization, use maple3lw_dev_init_params_t or maple3w↩ _dev_init_params_t.

3.5.2.3.2.8.8 void(* maple_open_params_t::maple_init)(void *input)

Pointer to maple_init function supplied by Maple Firmware.

3.5.2.3.2.9 struct maple_init_params_t

Maple Initialization parameters.

Data Fields

- unsigned int num_maples

3.5.2.3.2.9.1 Field Documentation

3.5.2.3.2.9.2 unsigned int maple_init_params_t::num_maples

Number of elements in maple_init_params_s array.

3.5.2.3.3 Macro Definition Documentation

3.5.2.3.3.1 #define MAPLE_B3LW_UCODE ((void *)&maple3lw_code[0])

MapleB3LW ucode definition, one for all standards.

3.5.2.3.3.2 #define MAPLE_B3W_UCODE ((void *)&maple3w_code[0])

MapleB3W ucode definition, one for all standards.

3.5.2.3.3.3 #define MAPLE_B3W_INIT maple3w_init

MaplB3W initialization function.

3.5.2.3.3.4 #define MAPLE_B3LW_INIT maple3lw_init

MaplB3LW initialization function.

3.5.2.3.3.5 #define MAPLE_SYS_ERR 0x80000000

OS_INT_MAPLE_SYS_ERR was asserted.

Might be because of Maple MMU error in case it was already handles by another core

3.5.2.3.3.6 #define MAPLE_MMU_ERR 0x10000000

OS_INT_MAPLE_SYS_ERR was asserted because of Maple MMU error.

3.5.2.3.3.7 #define MAPLE_ECC_ERR 0x00010000

OS_INT_MAPLE_ECC_ERR was asserted.

3.5.2.3.3.8 #define MAPLE_CMD_MALLOC (0x00000300 | COP_LLD_COMMAND)

When osCopDeviceControl called with MAPLE_CMD_MALLOC, param should be a typecast of maple_malloc_t.

3.5.2.3.3.9 #define MAPLE_CMD_GET_UCODE_VERSION (0x00000400 | COP_LLD_COMMAND)

Maple ucode version.

use maple_version_t to extract the version

3.5.2.3.3.10 #define MAPLE_CMD_PCR_ACTIVATE_WITH_POLL (0x00000500 | COP_LLD_COMMAND)

Activate Maple PCR routine and wait until it finishes.

3.5.2.3.3.11 #define MAPLE_CMD_PCR_ACTIVATE_NO_POLL (0x00000600 | COP_LLD_COMMAND)

Activate Maple PCR routine and don't wait until it finishes.

3.5.2.3.3.12 #define MAPLE_CMD_MMU_SEGMENT_FIND (0x00000700 | COP_LLD_COMMAND)

Find available maple mmu segment; pass pointer to os_mmu_segment_handle as parameters.

3.5.2.3.3.13 #define MAPLE_CMD_MMU_SEGMENT_UPDATE (0x00000800 | COP_LLD_COMMAND)

Update maple mmu segment; pass pointer to maple_mmu_seg_update_t as parameter; can be used only on descriptors found by MAPLE_CMD_MMU_SEGMENT_FIND.

3.5.2.3.3.14 #define MAPLE_CMD_MMU_SEGMENT_ENABLE (0x00000900 | COP_LLD_COMMAND)

Enable maple mmu segment; pass pointer to os_mmu_segment_handle as parameter; Another option to enable segment is usage of MAPLE_MMU_ATTR_ENABLE with MAPLE_CMD_MMU_SEGMENT↩ _UPDATE.

3.5.2.3.3.15 #define MAPLE_CMD_MMU_SEGMENT_DISABLE (0x00000A00 | COP_LLD_COMMAND)

Disable maple mmu segment; pass pointer to os_mmu_segment_handle as parameter.

3.5.2.3.3.16 #define MAPLE_CMD_MMU_ENABLE (0x00000B00 | COP_LLD_COMMAND)

Enable maple mmu address translation; pass pointer to maple_mmu_init_param_t as parameter By default enables MMU protection, use MAPLE_CMD_MMU_PROTECT_DISABLE for disabling it.

3.5.2.3.3.17 #define MAPLE_CMD_MMU_DISABLE (0x00000C00 | COP_LLD_COMMAND)

Disable maple mmu address translation.

3.5.2.3.3.18 #define MAPLE_CMD_MMU_PROTECT_DISABLE (0x00000D00 | COP_LLD_COMMAND)

Disable maple mmu memory protection disable.

3.5.2.3.3.19 #define MAPLE_CMD_MMU_PROTECT_ENABLE (0x00000E00 | COP_LLD_COMMAND)

Disable maple mmu memory protection enable.

3.5.2.3.3.20 #define MAPLE_CMD_MMU_ERROR_DETECT (0x00000F00 | COP_LLD_COMMAND)

Detect the cause of the error.

Pass pointer to maple_mmu_err_t as output parameter; Use it inside error_callback and only for MAPL↩ E_MMU_ERR

3.5.2.3.3.21 #define MAPLE_CMD_SOFT_RESET (0x00001000 | COP_LLD_COMMAND)

Activate Maple Soft Reset, param should be a typecast of maple_dev_init_params_t.

3.5.2.3.3.22 #define MAPLE_CMD_SET_TRACE_EVENTS (0x00001100 | COP_LLD_COMMAND)

Sets Maple Trace events to be traced, param should be a bitmask of events to trace according to maple_↩ trace_event_enable_t.

3.5.2.3.3.23 #define MAPLE_CMD_CLEAR_TRACE_BUFFER (0x00001200 | COP_LLD_COMMAND)

Clear MAPLE trace buffer.

3.5.2.3.3.24 #define MAPLE_CMD_GET_TRACE_BUFFER (0x00001300 | COP_LLD_COMMAND)

Copies Trace events from Trace buffer, param should be a pointer to an array of maple_trace_event_t.

3.5.2.3.3.25 #define MAPLE_CMD_GET_TRACE_BUFFER_WRITE_PTR (0x00001400 | COP_LLD_COMMAND)

Returns the Trace Buffer Write Pointer, param should be a pointer to a uint32_t type.

3.5.2.3.3.26 #define MAPLE_CMD_ACTIVATE_PDSCH_ACK_INT (0x00001500 | COP_LLD_COMMAND)

Returns the , param should be a the PDSCH sector number.

3.5.2.3.3.27 #define MAPLE_CMD_DISABLE_ECC_PROTECTION (0x00001600 | COP_LLD_COMMAND)

Clear and mask Maples general ECC error event.

3.5.2.3.3.28 #define MAPLE_CMD_ENABLE_ECC_PROTECTION (0x00001700 | COP_LLD_COMMAND)

Clear and unmask Maples general ECC error event.

3.5.2.3.3.29 #define MAPLE_CMD_DEPE_WA (0x00001800 | COP_LLD_COMMAND)

Activate DEPE WA for errta ERR007563.

3.5.2.3.3.30 #define MAPLE_CMD_PRE_POST_TASK_ID (0x00001900 | COP_LLD_COMMAND)

Configure the Task ID of the MMU descriptors to be used for all MAPLE-B3 external accesses related to the One-Shot initialization flow.

3.5.2.3.3.31 #define MAPLE_FLG_DEFAULT 0

Default maple flags to be used in maple_init_params_t.

3.5.2.3.3.32 #define MAPLE_0_NAME "MPL00"

Maple 0 device Name.

3.5.2.3.3.33 #define MAPLE_1_NAME "MPL01"

Maple 1 device Name.

3.5.2.3.3.34 #define MAPLE_2_NAME "MPL02"

Maple 2 device Name.

3.5.2.3.4 Enumeration Type Documentation

3.5.2.3.4.1 anonymous enum

Enumerator

MAPLE_MMU_MULTI_HIT_ERR Maple mmu multiple hit error.
MAPLE_MMU_MISS_ERR Maple mmu miss error.
MAPLE_MMU_READ_ERR Maple mmu read privilege error.
MAPLE_MMU_WRITE_ERR Maple mmu write privilege error.

3.5.2.3.4.2 enum maple_pspicer2_flag_t

Enumerator

MAPLE3LW_ECC_TV_DRE MAPLE3_LW eTVPE2 DRE memoryECC event indication.
MAPLE3LW_ECC_TV_NILL MAPLE3_LW eTVPE2 NIIL memory ECC event indication.
MAPLE3LW_ECC_TV_HARQ MAPLE3_LW eTVPE2 HARQ memory ECC event indication.
MAPLE3LW_ECC_FTPE0 MAPLE3_LW eFTPE2_0 memory ECC event indication.
MAPLE3LW_ECC_FTPE1 MAPLE3_LW eFTPE2_1 memory ECC event indication.
MAPLE3LW_ECC_FTPE2 MAPLE3_LW eFTPE2_2 memory ECC event indication.
MAPLE3LW_ECC_PUPE MAPLE3_LW PUPE2 memory ECC event indication.
MAPLE3LW_ECC_PDPE MAPLE3_LW PDPE2 memory ECC event indication.
MAPLE3LW_ECC_EQPE MAPLE3_LW EQPE memory ECC event indication.
MAPLE3W_ECC_FTPE0 MAPLE3_W eFTPE2_0 memory ECC event indication.
MAPLE3W_ECC_FTPE1 MAPLE3_W eFTPE2_1 memory ECC event indication.
MAPLE3W_ECC_TCPE MAPLE3_W TCPE memory ECC event indication.
MAPLE3W_ECC_CRPEULB0 MAPLE3_W CRPE-ULB2_0 memory ECC event indication.
MAPLE3W_ECC_CRPEULB1 MAPLE3_W CRPE-ULB2_1 memory ECC event indication.
MAPLE3W_ECC_CRPE_DL MAPLE3_W CRPE-DL2 memory ECC event indication.
MAPLE_ECC_RISC3_TRACE_BUFFER Trace Buffer memory of RISC3 ECC event indication.
MAPLE_ECC_RISC2_TRACE_BUFFER Trace Buffer memory of RISC2 ECC event indication.
MAPLE_ECC_RISC1_TRACE_BUFFER Trace Buffer memory of RISC1 ECC event indication.
MAPLE_ECC_RISC0_TRACE_BUFFER Trace Buffer memory of RISC0 ECC event indication.
MAPLE_ECC_HW_SCHEDULER_MEM1 Hardware Scheduler memory 1 ECC event indication.

MAPLE_ECC_HW_SCHEDULER_MEM0 Hardware Scheduler memory 0 ECC event indication.

MAPLE_ECC_DMA3_CMD_MEM1 System DMA engine 3 command memory 1 ECC event indication.
MAPLE_ECC_DMA2_CMD_MEM1 System DMA engine 2 command memory 1 ECC event indication.
MAPLE_ECC_DMA1_CMD_MEM1 System DMA engine 1 command memory 1 ECC event indication.
MAPLE_ECC_DMA0_CMD_MEM1 System DMA engine 0 command memory 1 ECC event indication.

MAPLE_ECC_DMA3_CMD_MEM0 System DMA engine 3 command memory 0 ECC event indication.
MAPLE_ECC_DMA2_CMD_MEM0 System DMA engine 2 command memory 0 ECC event indication.
MAPLE_ECC_DMA1_CMD_MEM0 System DMA engine 1 command memory 0 ECC event indication.
MAPLE_ECC_DMA0_CMD_MEM0 System DMA engine 0 command memory 0 ECC event indication.
MAPLE_ECC_DRAM DRAM memory ECC event indication.
MAPLE_ECC_IRAM IRAM memory ECC event indication.

3.5.2.3.4.3 enum maple_pcr_opcodes_t

MAPLEB3 PCR[OPCODE] routines mapping.

Enumerator

MAPLE_PARSE_TVPE_BD MAPLE-B3LW only.
MAPLE_PARSE_FTPE_2_BD MAPLE-B3LW only.
MAPLE_PARSE_DEPE_BD MAPLE-B3LW only.
MAPLE_PARSE_EQPE_BD MAPLE-B3LW only.
MAPLE_PARSE_CONVPE_BD MAPLE-B3LW only.
MAPLE_PARSE_PUSCH_EDF_BD MAPLE-B3LW only.
MAPLE_PARSE_PDSCH_EDF_BD MAPLE-B3LW only.
MAPLE_PDSCH_EXT_SYM_START MAPLE-B3LW only.
MAPLE_PUSCH_USER_NEIGHBOR_READY MAPLE-B3LW only.
MAPLE_PDSCH_EXT_SYM_START_1 MAPLE-B3LW only.
MAPLE_PDSCH_EXT_SYM_START_2 MAPLE-B3LW only.
MAPLE_PDSCH_EXT_SYM_START_3 MAPLE-B3LW only.
MAPLE_PDSCH_LATE_PARAMS_READY_0 MAPLE-B3LW only.
MAPLE_PDSCH_LATE_PARAMS_READY_1 MAPLE-B3LW only.
MAPLE_PDSCH_LATE_PARAMS_READY_2 MAPLE-B3LW only.
MAPLE_PDSCH_LATE_PARAMS_READY_3 MAPLE-B3LW only.

3.5.2.3.4.4 enum maple_dev_id_t

Maple device ID; Used in maple_init_params_t.

Enumerator

MAPLE_DEV_ID_NONE Used for signaling an unregistered MAPLE resource.
MAPLE_LW_ID_0 Maple device ID 0.
MAPLE_LW_ID_1 Maple device ID 1.
MAPLE_W_ID_2 Maple device ID 2.

3.5.2.3.4.5 anonymous enum

Values for maple3lw_dev_init_params_t.mode1[AXI_ALC] - Indicates whether the system AXI initiators buses allocation to each of the Processing Elements (PEs) is done internally by MAPLE-B3 or determined by the maple3lw_dev_init_params_t.mode2 parameter.

Enumerator

MAPLE_B3LW_INIT_INTERNAL_AXI_PE_ALC PEs allocation to each of the AXI interfaces is done internally by MAPLE-B3.

MAPLE_B3LW_INIT_EXTERNAL_AXI_PE_ALC PEs allocation to each of the AXI interfaces is done according to maple3lw_dev_init_params_t.mode2 configuration.

3.5.2.3.4.6 anonymous enum

Values for maple3lw_dev_init_params_t.mode1[AXI_PR_SCH] - Indicates the MAPLE-B3 which AX↩ Iinitiators priority scheme is applied.

Enumerator

MAPLE_B3LW_INIT_AXI_PR_SCHEME0 All MAPLE-B3 AXI master accesses are initiated with fixed priority level as described in the AXI0_DF_PR, AXI1_DF_PR, AXI2_DF_PRand AXI3_DF_PRfields for AXI0, AXI1, AXI2 and AXI3 respectively.

MAPLE_B3LW_INIT_AXI_PR_SCHEME1 All MAPLE-B3 AXI initiators accesses related to a certain BD are initiated with priority as described in the AXI_PR field of the BD. Previous DMA commands already launched into the relevant DMA queue are not upgraded with their priority accordingly.

MAPLE_B3LW_INIT_AXI_PR_SCHEME2 All MAPLE-B3 AXI initiators accesses related to a certain BD are initiated with priority as described in the AXI_PR field of the BD. All DM↩ A commands already launched into the relevant DMA queue are upgraded with their priority accordingly.

3.5.2.3.4.7 enum maple_b3lw_init_power_sch_pe_t

Parameter for MAPLE_B3LW_INIT_PEx_POWER_SCHEME - Indicates the PE to apply the power scheme on.

Enumerator

MAPLE_B3LW_INIT_POWER_SCHEME_PE_FTPE0 The power scheme will be set for eFTP↩ E0.

MAPLE_B3LW_INIT_POWER_SCHEME_PE_FTPE1 The power scheme will be set for eFTP↩ E1.

MAPLE_B3LW_INIT_POWER_SCHEME_PE_FTPE2 The power scheme will be set for eFTP↩ E2.

MAPLE_B3LW_INIT_POWER_SCHEME_PE_TVPE The power scheme will be set for eTVPE.

MAPLE_B3LW_INIT_POWER_SCHEME_PE_EQPE The power scheme will be set for EQPE.

3.5.2.3.4.8 enum maple_b3lw_init_power_sch_t

Parameter for MAPLE_B3LW_INIT_PEx_POWER_SCHEME - Indicates the power scheme to apply on the PE.

Enumerator

MAPLE_B3LW_INIT_POWER_SCHEME_AUTO Automatic power switching mode. The MAPLE-B3 power off the PE when there is no available BD for that PE.

MAPLE_B3LW_INIT_POWER_SCHEME_ON Always powered on. No power switching.

MAPLE_B3LW_INIT_POWER_SCHEME_OFF Powered off.

3.5.2.3.4.9 enum maple_b3lw_init_axi_aloc_bus_type_t

Parameter for MAPLE_B3LW_INIT_PEx_POWER_SCHEME - Indicates the PE's AXI Bus type.

Enumerator

MAPLE_B3LW_INIT_AXI_ALOC_READ_BUS The AXI Bus Allocation will be set for reading inputs by the PE.

MAPLE_B3LW_INIT_AXI_ALOC_WRITE_BUS The AXI Bus Allocation will be set for writing the PEs results.

3.5.2.3.4.10 enum maple_b3lw_init_axi_aloc_pe_t

Parameter for MAPLE_B3LW_INIT_PEx_POWER_SCHEME - Indicates the power scheme to apply on the PE.

Enumerator

MAPLE_B3LW_INIT_AXI_ALOC_PE_TVPE The AXI Bus Allocation will be set for eTVPE.

MAPLE_B3LW_INIT_AXI_ALOC_PE_FTPE0 The AXI Bus Allocation will be set for eFTPE0.

MAPLE_B3LW_INIT_AXI_ALOC_PE_FTPE1 The AXI Bus Allocation will be set for eFTPE1.

MAPLE_B3LW_INIT_AXI_ALOC_PE_FTPE2 The AXI Bus Allocation will be set for eFTPE2.

MAPLE_B3LW_INIT_AXI_ALOC_PE_CRC_DEPE The AXI Bus Allocation will be set for CRC and DEPE2.

MAPLE_B3LW_INIT_AXI_ALOC_PE_EQPE The AXI Bus Allocation will be set for EQPE.

MAPLE_B3LW_INIT_AXI_ALOC_PE_PDSCH The AXI Bus Allocation will be set for PDSCH2_EDF.

MAPLE_B3LW_INIT_AXI_ALOC_PE_PUSCH The AXI Bus Allocation will be set for PUSCH2_EDF.

3.5.2.3.4.11 enum maple_b3lw_init_axi_aloc_t

Parameter for MAPLE_B3LW_INIT_PEx_POWER_SCHEME - Indicates the power scheme to apply on the PE.

Enumerator

MAPLE_B3LW_INIT_ALOC_AXI0	Operation will be done using AXI0.
MAPLE_B3LW_INIT_ALOC_AXI1	Operation will be done using AXI1.
MAPLE_B3LW_INIT_ALOC_AXI2	Operation will be done using AXI2.
MAPLE_B3LW_INIT_ALOC_AXI3	Operation will be done using AXI3.

3.5.2.3.5 Function Documentation

3.5.2.3.5.1 INLINE long mapleDirectRead (void * *addr*, void * *data*, uint32_t *size*, uint32_t *dev_id*, uint32_t *param*)

Direct read from MAPLE.

Parameters

in	*addr*	- 4 byte aligned address in MAPLE memory map to access.
in	*data*	- 4 byte aligned pointer to data.
in	*size*	- Number of bytes to read.
in	*dev_id*	- Dummy value.
in	*param*	- Dummy value.

Returns

Number of bytes read

Warning

None

3.5.2.3.5.2 INLINE long mapleDirectWrite (void * *addr*, void * *data*, uint32_t *size*, uint32_t *dev_id*, uint32_t *param*)

Direct write to MAPLE.

Parameters

in	*addr*	- 4 byte aligned address in MAPLE memory map to access.

in	*data*	- 4 byte aligned pointer to data.
in	*size*	- Number of bytes to write.
in	*dev_id*	- Dummy value.
in	*param*	- Dummy value.

Returns

Number of bytes written

Warning

None

3.5.2.3.5.3 os_status mapleInitialize (struct maple_init_params_s * *init_params,* unsigned int *num_devices*)

Initializes the MAPLE driver's structures

The user should give the function an application dependent MAPLE parameters structure. Although this function is called by all cores, only the master core performs the initialization of the MAPLE registers.

Parameters

in	*init_params*	- MAPLE Initialization parameters. if NULL, default MAPLE parameters will be used.
in	*num_devices*	- Number of MAPLE devices to initialize.

Returns

OS_SUCCESS

Warning

This function is generally called by osArchInitialize() as part of the kernel and drivers

3.5.2.3.5.4 INLINE uint32_t MAPLE_B3LW_INIT_PEx_POWER_SCHEME (uint8_t *axi_idx,* uint8_t *priority*)

Calculates 32 bits word for setting an AXI default priority

Parameters

in	*axi_idx*	- Index of the AXI Bus (0 <= axi_idx < 4).
in	*priority*	- AXI Bus' default priority where 0 is the lowest priority and 3 is the highest priority (0 <= priority < 4).

Returns

32 bits word with MAPLE_B3LW_INIT_PE[axi_idx]_POWER_SCHEME calculated value

3.5.2.3.5.5 INLINE uint32_t MAPLE_B3LW_INIT_AXIx_DEFAULT_PRIORITY (maple_b3lw_init_power_sch_pe_t *pe*, maple_b3lw_init_power_sch_t *pwr_sch*)

Calculates 32 bits word for setting an AXI default priority

Parameters

in	*pe*	- Indicate the PE to apply the power scheme on.
in	*pwr_sch*	- Indicates the power scheme to apply on the PE.

Returns

32 bits word with MAPLE_B3LW_INIT_AXI[pe]_DEFAULT_PRIORITY calculated value

3.5.2.3.5.6 INLINE uint32_t MAPLE_B3LW_INIT_AXI_PE_ALC_PARAM (maple_b3lw_init_axi_aloc_pe_t *pe*, maple_b3lw_init_axi_aloc_t *axi_aloc*, maple_b3lw_init_axi_aloc_bus_type_t *type*)

Calculates 32 bits word for setting the PEs allocation to each of the AXI interfaces

Parameters

in	*pe*	- Indicate the PE to apply the AXI allocation
in	*axi_aloc*	- Indicate which AXI Bus will be allocated to the PE's operation.
in	*type*	- Indicate whether the allocation is for the PE's read or write operation.

Returns

32 bits word with MAPLE_B3LW_INIT_PE[pe][read/write]_AXI_ALLOCATION calculated value

Warning

Some of the PDSCH2_EDF and the PUSCH2_EDF related accesses will be assigned per the relevant PE involved in the processing. For example, PDSCH results (antenna data) will be output as per the relevant eFTPE2 bus allocation and not as the PDSCH2_EDF bus allocation.

3.5.2.3.6 Variable Documentation

3.5.2.3.6.1 unsigned long int maple3lw_code[]

MapleB3LW ucode array.

3.5.2.3.6.2 unsigned long int maple3w_code[]

MapleB3W ucode array.

3.5.2.4 Maple Workarounds API

3.5.2.4.1 Overview

```
MAPLE workarounds and additional API outside of MAPLE driver
```

Functions

- os_status maplePgCg (uint32_t maple_num)

3.5.2.4.2 Function Documentation

3.5.2.4.2.1 os_status maplePgCg (uint32_t *maple_num*)

Maple Power and Clock disable

Parameters

in	*maple_num*	- Number of the MAPLE to power OFF 0,1 stand for MAPLE3LW, 2 stands for MAPLE3W

Returns

OS_SUCCESS

Warning

Do not open MAPLE device number maple_num after maplePgCg(maple_num) activation, as it will result in MAPLE RISCs activation.

3.5.3 Maple PE Generic API

3.5.3.1 Overview

MAPLE all PE devices generic API

Modules

- Maple xxPE Runtime
- Maple xxPE Initialization

3.5.3.2 Maple xxPE Runtime

3.5.3.2.1 Overview

MAPLEB3 all PE devices generic runtime API

Data Structures

- struct maple_xxpe_ch_info_t

MAPLE XXPE Channel Control Commands

- #define MAPLE_PE_CH_CMD_RX_POLL (0x00000100 | COP_LLD_COMMAND)
- #define MAPLE_PE_CH_CMD_VIRT_TRANS_ENABLE (0x00000200 | COP_LLD_COMMA↩ ND)
- #define MAPLE_PE_CH_CMD_VIRT_TRANS_DISABLE (0x00000400 | COP_LLD_COMM↩ AND)
- #define MAPLE_PE_CH_CMD_GO (0x00000800 | COP_LLD_COMMAND)
- #define MAPLE_PE_CH_INFO_GET (0x00000900 | COP_LLD_COMMAND)
- #define MAPLE_PE_CH_ACTIVATE_JOB (0x00000A00 | COP_LLD_COMMAND)

3.5.3.2.2 Data Structure Documentation

3.5.3.2.2.1 struct maple_xxpe_ch_info_t

Structure to be used as output parameter for MAPLE_PE_CH_INFO_GET.

Data Fields

- void * bd_base
- void * bd_enq
- void * bd_deq
- uint32_t bd_ring_size
- uint32_t num_dispatched_bd
- uint32_t num_reaped_bd

3.5.3.2.2.1.1 Field Documentation

3.5.3.2.2.1.2 void* maple_xxpe_ch_info_t::bd_base

BD ring base address.

3.5.3.2.2.1.3 void* maple_xxpe_ch_info_t::bd_enq

Next BD to enqueue.

3.5.3.2.2.1.4 void* maple_xxpe_ch_info_t::bd_deq

Next BD to deque.

3.5.3.2.2.1.5 uint32_t maple_xxpe_ch_info_t::bd_ring_size

The size of the channel.

3.5.3.2.2.1.6 uint32_t maple_xxpe_ch_info_t::num_dispatched_bd

Counter for number of dispatched jobs; increased before BD is written to MAPLE memory.

3.5.3.2.2.1.7 uint32_t maple_xxpe_ch_info_t::num_reaped_bd

Counter for number of reaped jobs; increased after the reap callback.

3.5.3.2.3 Macro Definition Documentation

3.5.3.2.3.1 #define MAPLE_PE_CH_CMD_RX_POLL (0x00000100 | COP_LLD_COMMAND)

Polling the channel for finished jobs.

3.5.3.2.3.2 #define MAPLE_PE_CH_CMD_VIRT_TRANS_ENABLE (0x00000200 | COP_LLD_COMMAND)

Update channel translation mode: enable virtual to physical translation.

3.5.3.2.3.3 #define MAPLE_PE_CH_CMD_VIRT_TRANS_DISABLE (0x00000400 | COP_LLD_COMMAND)

Update channel translation mode: disable virtual to physical translation.

3.5.3.2.3.4 #define MAPLE_PE_CH_CMD_GO (0x00000800 | COP_LLD_COMMAND)

Go function for channel that have been opened with use_go_function flag; It sets OWNER bit on the first dispatched job thus triggers maple to start processing this batch.

3.5.3.2.3.5 #define MAPLE_PE_CH_INFO_GET (0x00000900 | COP_LLD_COMMAND)

Get channel handle information; helpful for debug.

3.5.3.2.3.6 #define MAPLE_PE_CH_ACTIVATE_JOB (0x00000A00 | COP_LLD_COMMAND)

Should be used only for jobs that were marked as a manually activated jobs and were not processed yet by MAPLE.

3.5.3.3 Maple xxPE Initialization

3.5.3.3.1 Overview

MAPLEB3 all PE devices generic initialization API

Data Structures

- struct maple_pe_init_params_t
- struct maple_pe_ch_open_params_t

Macros

- #define MAPLE_MAX_NUM_BD_PTRS 30

Typedefs

- typedef uint16_t maple_pe_ch
- typedef uint8_t maple_pe_hw_sem_id
- typedef os_status(∗ maple_xxpe_dispatch)(void ∗channel, void ∗jobs, int ∗num_jobs)
- typedef void(∗ maple_xxpe_reap)(void ∗channel, void ∗maple)

Enumerations

- enum maple_pe_bd_priority_t {
  MAPLE_PE_BD_RING_H, MAPLE_PE_BD_RING_H_L, MAPLE_PE_BD_RING_H_H_L,
  MAPLE_PE_BD_RING_H_H_H_L, MAPLE_PE_BD_RING_H_H_H_H_L, MAPLE_PE_BD_↩
  LAST_PRIORITY_SCHEME }
- enum maple_pe_num_bd_t {
  MAPLE_PE_NUM_BD_RINGS_1, MAPLE_PE_NUM_BD_RINGS_2, MAPLE_PE_NUM_B↩
  D_RINGS_3,
  MAPLE_PE_NUM_BD_RINGS_4, MAPLE_PE_NUM_BD_RINGS_5, MAPLE_PE_NUM_B↩
  D_RINGS_6,
  MAPLE_PE_NUM_BD_RINGS_7, MAPLE_PE_NUM_BD_RINGS_8 }
- enum { MAPLE_0_TYPE = 0x10000000, MAPLE_1_TYPE = 0x20000000, MAPLE_2_TYPE = 0x40000000 }
- enum maple_pe_type_t
- enum maple_pe_int_t { INT_LINE = 0x00000000, SRIO_DOORBELL_PORT0 = 0x00000100, SRIO_DOORBELL_PORT1 = 0x00000200 }
- enum maple_pe_int_trgt_t
- enum {
  MAPLE_STEERING_DSP_CLUSTER_0 = SOC_STEERING_BITS_DSP_CLUSTER_0, MAP↩
  LE_STEERING_DSP_CLUSTER_1 = SOC_STEERING_BITS_DSP_CLUSTER_1, MAPLE_S↩
  TEERING_DSP_CLUSTER_2 = SOC_STEERING_BITS_DSP_CLUSTER_2,
  MAPLE_STEERING_CHB = SOC_STEERING_BITS_CORE_NET_SLAVES, MAPLE_STEE↩
  RING_MAPLE_TRG = SOC_STEERING_BITS_MAPLE_SLAVES }

3.5.3.3.2 Data Structure Documentation

3.5.3.3.2.1 struct maple_pe_init_params_t

MAPLE xxPE generic device open parameters.

Data Fields

- unsigned int num_devices

3.5.3.3.2.1.1 Field Documentation

3.5.3.3.2.1.2 unsigned int maple_pe_init_params_t::num_devices

Number of elements in maple_pe_init_params_s array.

3.5.3.3.2.2 struct maple_pe_ch_open_params_t

MAPLE xxPE generic channel open parameters.

Data Fields

- os_mem_type channel_location
- os_hwi_dispatcher int_dispatcher
- os_hwi_handle int_num
- os_hwi_priority int_priority
- uint8_t mmu_task_id
- uint8_t steering_bits [MAPLE_MAX_NUM_BD_PTRS]
- uint32_t flags
- uint32_t high_priority:1
- uint32_t int_enable:1
- uint32_t no_translation:1
- uint32_t no_automatic_reap:1
- uint32_t single_channel_to_int:1
- uint32_t use_go_function:1

3.5.3.3.2.2.1 Field Documentation

3.5.3.3.2.2.2 uint32_t maple_pe_ch_open_params_t::flags

Users should set to 0 prior to setting any of the individual flags to ensure future compatibility.

Future releases may add more optimizations and flags to this 32 bit field. In order to ensure that these features, which will always be enabled by setting the relevant bit(s) to a value other than 0, won't be used unintentionally - the user should set flags to 0

3.5.3.3.2.2.3 uint32_t maple_pe_ch_open_params_t::high_priority

Set to 1 for a high priority channel or 0 low priority channel.

3.5.3.3.2.2.4 uint32_t maple_pe_ch_open_params_t::int_enable

Set to 1 to enable interrupts for this channel.

3.5.3.3.2.2.5 uint32_t maple_pe_ch_open_params_t::no_translation

Set to 1 to disable address translation on pointers passed from the applicaiton.

3.5.3.3.2.2.6 uint32_t maple_pe_ch_open_params_t::no_automatic_reap

Set to 1 to disable automatic reaping after dispatching when interrupts are disabled.

3.5.3.3.2.2.7 uint32_t maple_pe_ch_open_params_t::single_channel_to_int

Set to 1 for the driver to register the channel directly on the interrupt with no support for multiple channels on the interrupt line.

3.5.3.3.2.2.8 uint32_t maple_pe_ch_open_params_t::use_go_function

Set to 1 for using the GO function in order to trigger the batch of already dispatched jobs; In this mode ALL the dispatched jobs will be processed only after GO function activation.

3.5.3.3.2.2.9 os_mem_type maple_pe_ch_open_params_t::channel_location

Memory location for channel management.

Should be cacheable for performance

3.5.3.3.2.2.10 os_hwi_dispatcher maple_pe_ch_open_params_t::int_dispatcher

Interrupt dispatcher to be called when interrupt occurs.

3.5.3.3.2.2.11 os_hwi_handle maple_pe_ch_open_params_t::int_num

Which interrupt line on this device should the interrupt handler register to.

3.5.3.3.2.2.12 os_hwi_priority maple_pe_ch_open_params_t::int_priority

Interrupt priority.

3.5.3.3.2.2.13 uint8_t maple_pe_ch_open_params_t::mmu_task_id

MMU task id to be used for this channel.

3.5.3.3.2.2.14 uint8_t maple_pe_ch_open_params_t::steering_bits[MAPLE_MAX_NUM_BD_PTRS]

Steering bits for 16 possible pointers.

3.5.3.3.3 Macro Definition Documentation

3.5.3.3.3.1 #define MAPLE_MAX_NUM_BD_PTRS 30

Maximal number of pointers inside BD.

3.5.3.3.4 Typedef Documentation

3.5.3.3.4.1 typedef uint16_t maple_pe_ch

Maple channel number type.

3.5.3.3.4.2 typedef uint8_t maple_pe_hw_sem_id

Maple HW semaphore ID type.

3.5.3.3.4.3 typedef os_status(∗ maple_xxpe_dispatch)(void ∗channel, void ∗jobs, int ∗num_jobs)

MAPLE xxPE Dispatch function type.

3.5.3.3.4.4 typedef void(∗ maple_xxpe_reap)(void ∗channel, void ∗maple)

MAPLE xxPE Reap function type.

3.5.3.3.5 Enumeration Type Documentation

3.5.3.3.5.1 enum maple_pe_bd_priority_t

BD rings priority scheduling.

The scheduling mechanism used by the MAPLE u-code for scanning BD rings. The u-code will execute as many high priority BD as indicated by the number of "H" in the enumeration prior to executing a low priority BD.

Example: in case of MAPLE_PE_BD_RING_H_H_L after executing two jobs from any of the High priority rings of the XXXPE, it scans the next job in the Low priority rings of the XXXPE

Enumerator

MAPLE_PE_BD_RING_H MAPLE scans the High BD rings of the XXXPE only.
MAPLE_PE_BD_RING_H_L MAPLE executes one high followed by one low priority BD.
MAPLE_PE_BD_RING_H_H_L MAPLE executes two high followed by one low priority BD.
MAPLE_PE_BD_RING_H_H_H_L MAPLE executes three high followed by one low priority BD.

MAPLE_PE_BD_RING_H_H_H_H_L MAPLE executes four high followed by one low priority BD.
MAPLE_PE_BD_LAST_PRIORITY_SCHEME Software indicator - not to be used by the appli-caiton!

3.5.3.3.5.2 enum maple_pe_num_bd_t

Nunber of BD rings.

The number of BD rings for each priority. Low priority BD ring can be disabled by choosing paramater of maple_pe_bd_priority_t

Enumerator

MAPLE_PE_NUM_BD_RINGS_1	Only 1 BD ring per priority level is potentially valid.
MAPLE_PE_NUM_BD_RINGS_2	Only 2 BD rings per priority level are potentially valid.
MAPLE_PE_NUM_BD_RINGS_3	Only 3 BD rings per priority level are potentially valid.
MAPLE_PE_NUM_BD_RINGS_4	Only 4 BD rings per priority level are potentially valid.
MAPLE_PE_NUM_BD_RINGS_5	Only 5 BD rings per priority level are potentially valid.
MAPLE_PE_NUM_BD_RINGS_6	Only 6 BD rings per priority level are potentially valid.
MAPLE_PE_NUM_BD_RINGS_7	Only 7 BD rings per priority level are potentially valid.
MAPLE_PE_NUM_BD_RINGS_8	Only 8 BD rings per priority level are potentially valid.

3.5.3.3.5.3 anonymous enum

Enumerator

MAPLE_0_TYPE	MAPLE 0 PE type indication - LTE.
MAPLE_1_TYPE	MAPLE 1 PE type indication - LTE.
MAPLE_2_TYPE	MAPLE 2 PE type indication - WCDMA.

3.5.3.3.5.4 enum maple_pe_type_t

MAPLE xxPE types.

Warning

Not all types are supported by all MAPLE generations

3.5.3.3.5.5 enum maple_pe_int_t

MAPLE xxPE Interrupt options.

Enumerator

INT_LINE	Regular interrupt is generated according to MTVBRHPBxP[INT_TRGT].
SRIO_DOORBELL_PORT0	The door-bell should be initiated via Serial RapidIO port0.
SRIO_DOORBELL_PORT1	The door-bell should be initiated via Serial RapidIO port1.

3.5.3.3.5.6 enum maple_pe_int_trgt_t

MAPLE xxPE Interrupt line options.

Only relevant if maple_pe_int_t is INT_LINE; Defines which regular interrupt is to be asserted due to task completion in the BD ring

3.5.3.3.5.7 anonymous enum

MAPLE xxPE steering bits options to be used at maple_pe_ch_open_params_t.steering_bits[x].

Enumerator

MAPLE_STEERING_DSP_CLUSTER_0 DSP cluster 0.
MAPLE_STEERING_DSP_CLUSTER_1 DSP cluster 1.
MAPLE_STEERING_DSP_CLUSTER_2 DSP cluster 2.
MAPLE_STEERING_CHB Corenet/CHB.
MAPLE_STEERING_MAPLE_TRG Maple targets.

3.5.4 Maple CRCPE API

3.5.4.1 Overview

MAPLE CRCPE Initialization and Runtime API

Modules

- CRCPE Initialization
- CRCPE Runtime

3.5.4.2 CRCPE Initialization

3.5.4.2.1 Overview

CRCPE device initialization API

Data Structures

- struct crcpe_open_params_t

Typedefs

- typedef maple_pe_init_params_t maple_crcpe_init_params_t
- typedef struct maple_pe_init_params_s maple_crcpe_init_params_s
- typedef maple_pe_ch_open_params_t maple_crcpe_ch_open_params_t

Enumerations

- enum

Functions

- os_status mapleCrcpeInitialize (maple_crcpe_init_params_s *init_params, unsigned int num_devices, os_status(*channel_dispatch)(void *channel, void *jobs, int *num_jobs), void(*channel_reap)(void *channel, void *maple))

MAPLE CRCPE Device Names and IDs

- #define CRCPE_DEV_ID_0 0
- #define MAPLE_0_CRCPE_NAME "CRCPE0"
- #define MAPLE_1_CRCPE_NAME "CRCPE1"
- #define MAPLE_2_CRCPE_NAME "CRCPE2"

3.5.4.2.2 Data Structure Documentation

3.5.4.2.2.1 struct crcpe_open_params_t

MAPLE CRCPE Device Open parameters.

Data Fields

- void * maple_handle
- maple_pe_bd_priority_t maple_pe_bd_priority
- maple_pe_num_bd_t maple_pe_num_bd
- arch_specific crcpe_params

3.5.4.2.2.1.1 Field Documentation

3.5.4.2.2.1.2 void* crcpe_open_params_t::maple_handle

Handle returned from osCopDeviceOpen() for MAPLE controller.

3.5.4.2.2.1.3 maple_pe_bd_priority_t crcpe_open_params_t::maple_pe_bd_priority

BD rings priority scheduling - only configured by MAPLE master.

3.5.4.2.2.1.4 maple_pe_num_bd_t crcpe_open_params_t::maple_pe_num_bd

The number of BD rings for each priority - only configured by MAPLE master.

3.5.4.2.2.1.5 arch_specific crcpe_open_params_t::crcpe_params

Place holder for architecture specific initializations.

3.5.4.2.3 Macro Definition Documentation

3.5.4.2.3.1 #define CRCPE_DEV_ID_0 0

CRCPE device id for inside one Maple.

3.5.4.2.3.2 #define MAPLE_0_CRCPE_NAME "CRCPE0"

CRCPE device name for Maple 0.

3.5.4.2.3.3 #define MAPLE_1_CRCPE_NAME "CRCPE1"

CRCPE device name for Maple 1.

3.5.4.2.3.4 #define MAPLE_2_CRCPE_NAME "CRCPE2"

CRCPE device name for Maple 2.

3.5.4.2.4 Typedef Documentation

3.5.4.2.4.1 typedef maple_pe_init_params_t maple_crcpe_init_params_t

MAPLE CRCPE initialization parameters type for multiple PEs.

3.5.4.2.4.2 typedef struct maple_pe_init_params_s maple_crcpe_init_params_s

MAPLE CRCPE initialization parameters type for one PE.

3.5.4.2.4.3 typedef maple_pe_ch_open_params_t maple_crcpe_ch_open_params_t

MAPLE CRCPE channel open parameters type.

3.5.4.2.5 Enumeration Type Documentation

3.5.4.2.5.1 anonymous enum

MAPLE CRCPE steering bits pointers mapping.

Use it for accessing relevant maple_pe_ch_open_params_t.steering_bits[x]

3.5.4.2.6 Function Documentation

3.5.4.2.6.1 os_status mapleCrcpeInitialize (maple_crcpe_init_params_s * *init_params,* unsigned int *num_devices,* os_status(*)(void *channel, void *jobs, int *num_jobs) *channel_dispatch,* void(*)(void *channel, void *maple) *channel_reap*)

Initializes the CRCPE driver's structures

The driver can supply default MAPLE parameters for initialization. The user can override these parameters by specifying an alternative MAPLE parameters structure. Although this function is called by all cores, only the master core performs the initialization of the MAPLE registers.

Parameters

in	*init_params*	- MAPLE Initialization parameters. if NULL, default MAPLE parameters will be used.
in	*num_devices*	- Number of CRCPE devices
in	*channel_dispatch*	- Pointer to channel dispatch function.
in	*channel_reap*	- Pointer to channel reap function.

Returns

OS_SUCCESS

Warning

This function is generally called by osArchInitialize() as part of the kernel and drivers

3.5.4.3 CRCPE Runtime

3.5.4.3.1 Overview

CRCPE Runtime API

Data Structures

- struct maple_crcpe_job_t

Macros

- #define CRCPE_MAX_NUM_BD_FOR_DISPACTH 16

Functions

- INLINE uint16_t CRCPE_BS (unsigned short size)

MAPLE CRCPE Channel Control Commands

- #define MAPLE_CRCPE_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL

Flags for maple_crcpe_job_t.flags

- enum {
  CRCPE_MB_PRIORITY_0 = 0x0000, CRCPE_MB_PRIORITY_1 = 0x0100, CRCPE_MB_PRIORITY_2 = 0x0200,
  CRCPE_MB_PRIORITY_3 = 0x0300 }

- enum { CRCPE_CALCULATION_JOB = 0x0000, CRCPE_CHECK_JOB = 0x0080 }
- enum {
  CRCPE_CRC24_POLY0 = 0x0000, CRCPE_CRC24_POLY1 = 0x0001, CRCPE_CRC16_CCITT = 0x0002,
  CRCPE_CRC16 = 0x0003, CRCPE_CRC32 = 0x0004, CRCPE_CRC18 = 0x0005,
  CRCPE_CRC12 = 0x0006, CRCPE_CRC8 = 0x0007 }
- #define CRCPE_BD_MANUAL_ACTIVATION 0x8000
- #define CRCPE_INT_EN 0x1000
- #define CRCPE_UPDATE 0x0040
- #define CRCPE_RVRS_IN 0x0020
- #define CRCPE_RVRS_OUT 0x0010
- #define CRCPE_INV_OUT 0x0008

CRC auxiliray flags - used in maple_crcpe_job_t.aux

- #define CRCPE_CHECK_FAIL 0x80000000

3.5.4.3.2 Data Structure Documentation

3.5.4.3.2.1 struct maple_crcpe_job_t

CRCPE Job Descriptor.

This structure should be passed to the LLD on the cop_job_handle.device_specific field in order for the LLD to build the necessary BD

Data Fields

- uint16_t flags
- uint16_t buffer_size
- uint32_t * input
- uint32_t init_value
- uint32_t result
- uint32_t aux
- uint8_t bd_index

3.5.4.3.2.1.1 Field Documentation

3.5.4.3.2.1.2 uint16_t maple_crcpe_job_t::flags

User should provide an or mask of the following defines and enumerations; LLD will provide a mask for relevant bits.

Minimal sanity check for input parameters; CRCPE_INT_EN, CRCPE_UPDATE, CRCPE_RVRS_IN, CRCPE_RVRS_OUT, CRCPE_INV_OUT, CRCPE_MB_PRIORITY_X, CRCPE_XXX_JOB and CR↩ CPE_CRCXXX

3.5.4.3.2.1.3 uint16_t maple_crcpe_job_t::buffer_size

Buffer size on which to perform CRC calcultion.

Use CRCPE_BS() to initialize

3.5.4.3.2.1.4 uint32_t* maple_crcpe_job_t::input

Input Buffer Address.

This field points to the input buffer location in system memory, where MAPLE is to fetch the data into the CRCPE input buffer

3.5.4.3.2.1.5 uint32_t maple_crcpe_job_t::init_value

CRC Initialization value.

MAPLE uses this field as the CRC initialization value for the CRC processing. Only the relevant bits, based on the size of the polynomial are used

3.5.4.3.2.1.6 uint32_t maple_crcpe_job_t::result

CRC Result if CRCPE_CALCULATION_JOB is used.

The result of the CRC calculation performed on the input buffer

3.5.4.3.2.1.7 uint32_t maple_crcpe_job_t::aux

Auxiliary flags - User should provide an or mask of the following defines and enumerations; LLD will provide a mask for relevant bits; Minimal sanity check for input parameters; CRCPE_CHECK_FAIL.

3.5.4.3.2.1.8 uint8_t maple_crcpe_job_t::bd_index

Job's index in the BD ring.

Will be assigned by the driver

3.5.4.3.3 Macro Definition Documentation

3.5.4.3.3.1 #define CRCPE_MAX_NUM_BD_FOR_DISPACTH 16

Maximal number of CRCPE BD per dispatch.

Defines the maximal number of BD that can be dispatched with a single call to osCopChannelDispatch(). The larger the number the higher the stack consumption is in the driver.

Warning

Users may change this value, however it requires recompiling the drivers

3.5.4.3.3.2 #define MAPLE_CRCPE_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL

Poll the channel for finished jobs.

Use MAPLE_PE_CH_CMD_RX_POLL instead

3.5.4.3.3.3 #define CRCPE_BD_MANUAL_ACTIVATION 0x8000

Prevent the driver from dispatching the job, MAPLE will wait until the user calls a Channel Ctrl Command to activate the job.

3.5.4.3.3.4 #define CRCPE_INT_EN 0x1000

MAPLE issues an interrupt interrupt at the end of job.

3.5.4.3.3.5 #define CRCPE_UPDATE 0x0040

MAPLE copies the CRC result into the system memory at the and of the input buffer.

3.5.4.3.3.6 #define CRCPE_RVRS_IN 0x0020

MAPLE performs byte reverse CRC calculation/check.

3.5.4.3.3.7 #define CRCPE_RVRS_OUT 0x0010

MAPLE performs inverse operation on the CRC result.

3.5.4.3.3.8 #define CRCPE_INV_OUT 0x0008

MAPLE performs reverse on the CRC result.

3.5.4.3.3.9 #define CRCPE_CHECK_FAIL 0x80000000

If this is set in maple_crcpe_job_t.aux after running a job with CRCPE_CHECK_JOB set in maple_↩ crcpe_job_t.flags - the CRC check has failed.

3.5.4.3.4 Enumeration Type Documentation

3.5.4.3.4.1 anonymous enum

MBus Priority - used in maple_crcpe_job_t.flags.

Valid only if the [AXI_PR_SCH] of the MMC0P parameter is not 0.

Enumerator

CRCPE_MB_PRIORITY_0	The MBus accesses related to that BD are initiated with priority 0.
CRCPE_MB_PRIORITY_1	The MBus accesses related to that BD are initiated with priority 1.
CRCPE_MB_PRIORITY_2	The MBus accesses related to that BD are initiated with priority 2.
CRCPE_MB_PRIORITY_3	The MBus accesses related to that BD are initiated with priority 3.

3.5.4.3.4.2 anonymous enum

CRC job type - used in maple_crcpe_job_t.flags.

Describes whether the CRCPE is calculating the CRC on a buffer or validating the CRC on a buffer

Enumerator

CRCPE_CALCULATION_JOB MAPLE-B performs CRC calculation on the input buffer.
CRCPE_CHECK_JOB MAPLE-B performs CRC check on the input buffer.

3.5.4.3.4.3 anonymous enum

CRC polynomials - used in maple_crcpe_job_t.flags.

The polynomial used for CRC calucation/checking

Enumerator

CRCPE_CRC24_POLY0 Polynomial: D24 + D23 + D6 + D5 + D + 1.
CRCPE_CRC24_POLY1 Polynomial: D24 + D23 + D18 + D17 + D14 + D11 + D10 + D7 + D6 + D5 + D4 + D3 + D + 1.
CRCPE_CRC16_CCITT Polynomial: D16 +D12 + D5 + 1.
CRCPE_CRC16 Polynomial: D16 + D15 + D2 + 1.
CRCPE_CRC32 Polynomial: D32 + D26 + D23 + D22 + D16 + D12 + D11 + D10 + D8 + D7 + D5 + D4 + D2 + D + 1.
CRCPE_CRC18 Polynomial: D18 + D17 + D14 + D13 + D11 + D10 + D8 + D7 + D6 + D3 + D2 + 1.
CRCPE_CRC12 Polynomial: D12 + D11 + D10 + D8 + D5 + D4 + 1.
CRCPE_CRC8 Polynomial: D8 + D7 + D4 + D3 + D + 1.

3.5.4.3.5 Function Documentation

3.5.4.3.5.1 INLINE uint16_t CRCPE_BS (unsigned short *size*)

Calculates BD[CRC_BS] - used in maple_crcpe_job_t.buffer_size

Describes the input buffer size (in bytes); For CRC word generation the minimum block size is 2 bytes; For CRC check the minimum block size is 5 bytes

Parameters

in	*size*	- input buffer size (in bytes)

Returns

BD[CRC_BS] culculated value

3.5.5 Maple TVPE API

3.5.5.1 Overview

```
MAPLE TVPE Initialization and Runtime API
```

Modules

- TVPE Initialization
- TVPE Runtime

Macros

- #define TVPE_MAX_NUM_BD_FOR_DISPACTH 16
- #define NUM_VITERBI_POLYNOM_SETS 3
- #define NUM_VITERBI_PUNC_VECTORS 10

3.5.5.2 Macro Definition Documentation

3.5.5.2.1 #define TVPE_MAX_NUM_BD_FOR_DISPACTH 16

Maximum number of BDs that are allowed to dispatch all at once.

Users may change this value and recompile the drivers. Change in this value causes a linear change in runtime stack consumption

3.5.5.2.2 #define NUM_VITERBI_POLYNOM_SETS 3

Number of viterbi polynomial sets.

3.5.5.2.3 #define NUM_VITERBI_PUNC_VECTORS 10

Number of viterbi puncturing vectors.

3.5.5.3 TVPE Initialization

3.5.5.3.1 Overview

```
TVPE device initialization API
```

Data Structures

- struct tvpe_viterbi_polynomial_set
- struct tvpe_turbo_params_t
- struct tvpe_viterbi_params_t
- struct tvpe_open_params_t

Typedefs

- typedef maple_pe_init_params_t maple_tvpe_init_params_t

- typedef struct maple_pe_init_params_s maple_tvpe_init_params_s
- typedef maple_pe_ch_open_params_t maple_tvpe_ch_open_params_t

Enumerations

- enum

Functions

- os_status mapleTvpeInitialize (maple_tvpe_init_params_s *init_params, unsigned int num_devices, os_status(*channel_dispatch)(void *channel, void *jobs, int *num_jobs), void(*channel_reap)(void *channel, void *maple))

MAPLE TVPE Device Names and IDs

- #define TVPE_DEV_ID_0 0
- #define TVPE_DEV_ID_1 1
- #define MAPLE_0_TVPE_NAME "TVPE0"
- #define MAPLE_1_TVPE_NAME "TVPE1"

3.5.5.3.2 Data Structure Documentation

3.5.5.3.2.1 struct tvpe_viterbi_polynomial_set

MAPLE TVPE Device viterbi polynomial sets parameters.

Also used in control MAPLE_TVPE_CMD_SET_VITERBI_POLY.

Data Fields

- uint16_t m_polygen1
- uint16_t m_polygen0
- uint16_t m_polygen3
- uint16_t m_polygen2

3.5.5.3.2.1.1 Field Documentation

3.5.5.3.2.1.2 uint16_t tvpe_viterbi_polynomial_set::m_polygen1

Viterbi Polynomial 1.

Valid bits are [7:0]. Same as polygen1 at TVVPVG0CR

3.5.5.3.2.1.3 uint16_t tvpe_viterbi_polynomial_set::m_polygen0

Viterbi Polynomial 0.

Valid bits are [7:0]. Same as polygen0 at TVVPVG0CR

3.5.5.3.2.1.4 uint16_t tvpe_viterbi_polynomial_set::m_polygen3

Viterbi Polynomial 3.

Valid bits are [7:0]. Same as polygen3 at TVVPVG1CR

3.5.5.3.2.1.5 uint16_t tvpe_viterbi_polynomial_set::m_polygen2

Viterbi Polynomial 2.

Valid bits are [7:0]. Same as polygen2 at TVVPVG1CR

3.5.5.3.2.2 struct tvpe_turbo_params_t

MAPLE TVPE Device Turbo Open paramaters.

Data Fields

- uint32_t apq_threshold

3.5.5.3.2.2.1 Field Documentation

3.5.5.3.2.2.2 uint32_t tvpe_turbo_params_t::apq_threshold

The Aposteriori threshold.

Valid values are 0-0x7FFF. Required when app_stop_criteria is not 0

3.5.5.3.2.3 struct tvpe_viterbi_params_t

MAPLE TVPE Device Viterbi Open paramaters.

Data Fields

- uint64_t punturing_vector [NUM_VITERBI_PUNC_VECTORS]
- uint8_t puncturing_pattern [NUM_VITERBI_PUNC_VECTORS]
- tvpe_viterbi_polynomial_set polynomial_set

3.5.5.3.2.3.1 Field Documentation

3.5.5.3.2.3.2 uint64_t tvpe_viterbi_params_t::punturing_vector[NUM_VITERBI_PUNC_VECTORS]

Puncturing Vector.

For each bit in the vector: 0 - Punctured symbol, 1 - Valid symbol

3.5.5.3.2.3.3 uint8_t tvpe_viterbi_params_t::puncturing_pattern[NUM_VITERBI_PUNC_VECTORS]

Indicates the period of the punturing_vector[i].

Valid values are 0-63

3.5.5.3.2.3.4 tvpe_viterbi_polynomial_set tvpe_viterbi_params_t::polynomial_set

Viterbi Polynomial Sets.

MTVPVSxCyP

3.5.5.3.2.4 struct tvpe_open_params_t

MAPLE TVPE Device Open LLD paramaters.

Data Fields

- void * maple_handle
- maple_pe_bd_priority_t maple_pe_bd_priority
- maple_pe_num_bd_t maple_pe_num_bd
- tvpe_turbo_params_t * tvpe_turbo_params
- tvpe_viterbi_params_t * tvpe_viterbi_params

3.5.5.3.2.4.1 Field Documentation

3.5.5.3.2.4.2 void* tvpe_open_params_t::maple_handle

Handle returned from osCopDeviceOpen() for MAPLE controller.

3.5.5.3.2.4.3 maple_pe_bd_priority_t tvpe_open_params_t::maple_pe_bd_priority

BD rings priority scheduling - only configured by MAPLE master.

3.5.5.3.2.4.4 maple_pe_num_bd_t tvpe_open_params_t::maple_pe_num_bd

The number of BD rings for each priority - only configured by MAPLE master.

3.5.5.3.2.4.5 tvpe_turbo_params_t* tvpe_open_params_t::tvpe_turbo_params

Parameters for initializing the Turbo functionality of the TVPE.

May set to NULL for HW defaults or non MAPLE master devices

3.5.5.3.2.4.6 tvpe_viterbi_params_t* tvpe_open_params_t::tvpe_viterbi_params

Parameters for initializing the Viterbi functionality of the TVPE.

May set to NULL for HW defaults or non MAPLE master devices

3.5.5.3.3 Macro Definition Documentation

3.5.5.3.3.1 #define TVPE_DEV_ID_0 0

Maple eTVPE 0 device ID.

3.5.5.3.3.2 #define TVPE_DEV_ID_1 1

Maple eTVPE 1 device ID.

3.5.5.3.3.3 #define MAPLE_0_TVPE_NAME "TVPE0"

Maple eTVPE 0 device name.

3.5.5.3.3.4 #define MAPLE_1_TVPE_NAME "TVPE1"

Maple eTVPE 0 device name.

3.5.5.3.4 Typedef Documentation

3.5.5.3.4.1 typedef maple_pe_init_params_t maple_tvpe_init_params_t

MAPLE TVPE initialization parameters type for multiple PEs.

3.5.5.3.4.2 typedef struct maple_pe_init_params_s maple_tvpe_init_params_s

MAPLE TVPE initialization parameters type for one PE.

3.5.5.3.4.3 typedef maple_pe_ch_open_params_t maple_tvpe_ch_open_params_t

MAPLE TVPE channel open parameters type.

3.5.5.3.5 Enumeration Type Documentation

3.5.5.3.5.1 anonymous enum

MAPLE TVPE steering bits pointers mapping.

Use it for accessing relevant maple_pe_ch_open_params_t.steering_bits[x]

3.5.5.3.6 Function Documentation

3.5.5.3.6.1 os_status mapleTvpeInitialize (maple_tvpe_init_params_s * *init_params,* unsigned int *num_devices,* os_status(*)(void *channel, void *jobs, int *num_jobs) *channel_dispatch,* void(*)(void *channel, void *maple) *channel_reap*)

Initializes the TVPE driver's structures

Parameters

in	*init_params*	- TVPE Initialization parameters
in	*num_devices*	- Number of TVPE devices
in	*channel_dispatch*	- Pointer to channel dispatch function
in	*channel_reap*	- Pointer to channel reap function

Returns

OS_SUCCESS

Warning

This function is generally called by osArchInitialize() as part of the kernel and drivers

3.5.5.4 TVPE Runtime

3.5.5.4.1 Overview

TVPE Runtime API

Data Structures

- struct tvpe_job_e_param_t
- struct maple_tvpe_job_t

MAPLE TVPE Device Control Commands.

- #define MAPLE_TVPE_CMD_SET_VITERBI_POLY (0x00000200 | COP_LLD_COMMAND)
- #define MAPLE_TVPE_CMD_GET_PE_ACCUMULATOR (0x00000300 | COP_LLD_COMMAND)

MAPLE TVPE Channel Control Commands.

- #define MAPLE_TVPE_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL

MAPLE TVPE FLAGS for maple_tvpe_job_t.first_flags.

- enum maple_tvpe_k_t
- enum maple_tvpe_rate_t { TVPE_RATE_1_2 = 0x00000000, TVPE_RATE_1_3 = 0x00040000, TVPE_RATE_1_4 = 0x00080000 }
- enum maple_tvpe_type_t
- enum maple_tvpe_punc_t
- enum maple_tvpe_max_iter_t

- enum maple_tvpe_min_iter_t
- enum maple_tvpe_data_struct_t {
  TVPE_LTE_HARQ = 0x00000000, TVPE_WIMAX_HARQ = 0x00000001, TVPE_EDCH_HA↩
  RQ_MIXED = 0x00000002,
  TVPE_EDCH_HARQ_SEPARATE = 0x00000003, TVPE_SUB_BLK_INTRLV = 0x00000004,
  TVPE_UMTS_MIXED = 0x00000005,
  TVPE_SEPARATE_VECTORS = 0x00000006 }
- #define TVPE_BD_MANUAL_ACTIVATION 0x80000000
- #define TVPE_INT_EN 0x10000000
- #define TVPE_TURBO_LMAP_EN 0x08000000
- #define TVPE_3GLTE_EN 0x04000000
- #define TVPE_VITERBI_TBZE 0x00400000
- #define TVPE_ZTTB 0x00200000
- #define TVPE_TURBO_CRC_EN 0x00100000
- #define TVPE_TURBO_HAOE 0x00020000

MAPLE TVPE FLAGS for maple_tvpe_job_t.second_flags.

- enum maple_tvpe_crc_t {
  TVPE_CRC24_POLY0 = 0x00000000, TVPE_CRC24_POLY1 = 0x00000040, TVPE_CRC16_↩
  POLY0 = 0x00000080,
  TVPE_CRC16_POLY1 = 0x000000C0 }
- enum maple_tvpe_dre_t { TVPE_ONE_DRE = 0x00000000, TVPE_TWO_DRE = 0x00000010,
  TVPE_FOUR_DRE = 0x00000020 }
- enum maple_tvpe_mbus_prio_t {
  TVPE_MBUS_PRIORITY_0 = 0x00000000, TVPE_MBUS_PRIORITY_1 = 0x00000001, TVP↩
  E_MBUS_PRIORITY_2 = 0x00000002,
  TVPE_MBUS_PRIORITY_3 = 0x00000003 }
- #define TVPE_BLOCK_SIZE(K) (((uint32_t)((K) & 0x7FFF)) << 16)
- #define TVPE_TURBO_LLMAP_LCF(K) (((K) & 0x0FE) << 8)

MAPLE TVPE FLAGS for maple_tvpe_job_t.third_flags.

- enum maple_tvpe_soft_output_t
- enum maple_tvpe_vit_set_t { TVPE_VIT_SET_1 = 0x00040000, TVPE_VIT_SET_2 =
  0x00080000, TVPE_VIT_SET_3 = 0x000C0000 }
- #define TVPE_HOE 0x80000000
- #define TVPE_TURBO_DPNC_EN 0x10000000
- #define TVPE_BUF_SIZE(K) ((K) & 0x3FFFF)

MAPLE TVPE FLAGS for maple_tvpe_job_t.offsets.

- #define TVPE_TURBO_VEC_OFFSET(K) ((K) << 16)
- #define TVPE_HARD_OUT_OFFSET(K) ((K) << 4)
- #define TVPE_TURBO_LLR_OUT_SF(K) (K)

MAPLE TVPE FLAGS for harq_buffer.

- #define TVPE_HARQ_IHBSA(K) (((K) & 0x7FFF) << 16)
- #define TVPE_FIRST_TIME_HARQ 0x00008000
- #define TVPE_HARQ_IHBSZ(K) ((K) & 0x7FFF)

MAPLE TVPE FLAGS for maple_tvpe_job_t.harq_weight_flags.

- enum tvpe_harq_hus_t {
  TVPE_TURBO_HARQ_NO_UP_SCL = 0x00000000, TVPE_TURBO_HARQ_UP_SCL_2 = 0x10000000, TVPE_TURBO_HARQ_UP_SCL_4 = 0x20000000,
  TVPE_TURBO_HARQ_UP_SCL_8 = 0x30000000 }
- #define TVPE_TURBO_HARQ_W1_EN 0x04000000
- #define TVPE_TURBO_HARQ_W2_EN 0x02000000
- #define TVPE_TURBO_HARQ_W3_EN 0x01000000
- #define TVPE_TURBO_HARQ_W1(K) ((K & 0xFF) << 16)
- #define TVPE_TURBO_HARQ_W2(K) ((K & 0xFF) << 8)
- #define TVPE_TURBO_HARQ_W3(K) (K & 0xFF)

MAPLE TVPE FLAGS for maple_tvpe_job_t.fourth_flags.

- #define TVPE_CRC_REFLECT 0x80000000
- #define TVPE_AQC_AUTOSTOP 0x40000000
- #define TVPE_CRC_AUTOSTOP 0x20000000
- #define TVPE_CRC_SCRC 0x10000000
- #define TVPE_HARQ_EN 0x08000000
- #define TVPE_POLARITY 0x04000000
- #define TVPE_DOBSY 0x02000000
- #define TVPE_DOBSI 0x01000000

3.5.5.4.2 Data Structure Documentation

3.5.5.4.2.1 struct tvpe_job_e_param_t

MAPLE TVPE maple_tvpe_job_t.e_param_ptr structure.

Data Fields

- uint32_t e_cb_sd_ini
- uint32_t e_sd_minus
- uint32_t e_sd_plus
- uint32_t e_cb_pfa_ini
- uint32_t e_pfa_minus
- uint32_t e_pfa_plus
- uint32_t e_cb_psa_ini
- uint32_t e_psa_minus
- uint32_t e_psa_plus

3.5.5.4.2.1.1 Field Documentation

3.5.5.4.2.1.2 uint32_t tvpe_job_e_param_t::e_cb_sd_ini

The e_init parameter for SD, bits[31-30] include skip count value.

3.5.5.4.2.1.3 uint32_t tvpe_job_e_param_t::e_sd_minus

The e_minus parameter for SD.

3.5.5.4.2.1.4 uint32_t tvpe_job_e_param_t::e_sd_plus

The e_plus parameter for SD.

3.5.5.4.2.1.5 uint32_t tvpe_job_e_param_t::e_cb_pfa_ini

The e_init parameter for PF, bits[31-30] include skip count value.

3.5.5.4.2.1.6 uint32_t tvpe_job_e_param_t::e_pfa_minus

The e_minus parameter for PF.

3.5.5.4.2.1.7 uint32_t tvpe_job_e_param_t::e_pfa_plus

The e_plus parameter for PF.

3.5.5.4.2.1.8 uint32_t tvpe_job_e_param_t::e_cb_psa_ini

The e_init parameter for PS, bits[31-30] include skip count value.

3.5.5.4.2.1.9 uint32_t tvpe_job_e_param_t::e_psa_minus

The e_minus parameter for PS.

3.5.5.4.2.1.10 uint32_t tvpe_job_e_param_t::e_psa_plus

The e_plus parameter for PS.

3.5.5.4.2.2 struct maple_tvpe_job_t

MAPLE TVPE Job.

This structure should be passed to the LLD on the cop_job_handle.device_specific field in order for the LLD to build the necessary BD

Data Fields

- uint32_t first_flags
- uint32_t second_flags
- uint32_t third_flags
- void * hard_output_addr
- void * inputs

- uint32_t offsets
- uint32_t harq_buffer
- uint32_t harq_weight_flags
- void * soft_output_addr
- tvpe_job_e_param_t * e_param_ptr
- void * harq_acc_input_addr
- void * harq_acc_output_addr
- uint32_t fourth_flags
- uint8_t bd_index
- uint32_t pfs_vec_sizes
- uint16_t pf_vector_size
- uint16_t ps_vector_size

3.5.5.4.2.2.1 Field Documentation

3.5.5.4.2.2.2 uint32_t maple_tvpe_job_t::first_flags

User should provide an or mask of the following defines and enumerations; LLD will provide a mask for relevant bits; Minimal sanity check for input parameters;.

Viterbi and Turbo: TVPE_INT_EN, TVPE_ZTTB, TVPE_3GLTE_EN, maple_tvpe_type_t, maple_↩ tvpe_data_struct_t, maple_tvpe_rate_t, Viterbi only: maple_tvpe_k_t, maple_tvpe_punc_t TVPE_VIT↩ ERBI_TBZE, Turbo only: TVPE_TURBO_LMAP_EN, TVPE_TURBO_CRC_EN, TVPE_TURBO_↩ HAOE, maple_tvpe_max_iter_t, maple_tvpe_min_iter_t

3.5.5.4.2.2.3 uint32_t maple_tvpe_job_t::second_flags

User should provide an or mask of the following defines and enumerations; LLD will provide a mask for relevant bits.

Minimal sanity check for input parameters;

Viterbi and Turbo: TVPE_BLOCK_SIZE(K), maple_tvpe_mbus_prio_t Turbo only: maple_tvpe_dre_t, maple_tvpe_crc_t, TVPE_TURBO_LLMAP_LCF(K),

3.5.5.4.2.2.4 uint32_t maple_tvpe_job_t::third_flags

User should provide an or mask of the following defines and enumerations; LLD will provide a mask for relevant bits.

Minimal sanity check for input parameters;

Viterbi and Turbo: TVPE_HOE, TVPE_BUF_SIZE(K) Turbo only: maple_tvpe_soft_output_t, TVPE_↩ TURBO_DPNC_EN Viterbi only: maple_tvpe_vit_set_t

3.5.5.4.2.2.5 void* maple_tvpe_job_t::hard_output_addr

Address for hard outputs; Only valid if TVPE_HOE is set.

3.5.5.4.2.2.6 void* maple_tvpe_job_t::inputs

Address of input buffer.

- Points to the start address of the input buffer in case single input stream is required. Points to the start address in the system memory of the first out of several input vectors. The start address of the other vectors depends on maple_tvpe_job_t.offsets

3.5.5.4.2.2.7 uint32_t maple_tvpe_job_t::offsets

Vectors Offset, Hard Output Offset, LLR Output Shift.

Use the flags bellow to initialize this field.

Viterbi and Turbo: TVPE_HARD_OUT_OFFSET(K) Turbo only: maple_tvpe_soft_out_off_t, TVPE_↩ TURBO_LLR_OUT_SF(K), TVPE_TURBO_VEC_OFFSET(K)

3.5.5.4.2.2.8 uint32_t maple_tvpe_job_t::harq_buffer

HARQ input buffer descriptor fields: TVPE_HARQ_IHBSA(K), TVPE_FIRST_TIME_HARQ, TVPE↩ _HARQ_IHBSZ(K)

3.5.5.4.2.2.9 uint32_t maple_tvpe_job_t::harq_weight_flags

HARQ Up Scale and weights; Use the flags bellow to initialize this field.

Turbo only: tvpe_harq_hus_t, TVPE_TURBO_HARQ_W1_EN, TVPE_TURBO_HARQ_W2_EN, TV↩ PE_TURBO_HARQ_W3_EN, TVPE_TURBO_HARQ_W1(K), TVPE_TURBO_HARQ_W2(K), TV↩ PE_TURBO_HARQ_W3(K)

3.5.5.4.2.2.10 void* maple_tvpe_job_t::soft_output_addr

Address for soft outputs.

Only valid if user requested soft outputs. Turbo only

3.5.5.4.2.2.11 tvpe_job_e_param_t* maple_tvpe_job_t::e_param_ptr

A pointer to the E parameters data structure used by the TVPE in order to perform the Rate De-Matching process.

3.5.5.4.2.2.12 void* maple_tvpe_job_t::harq_acc_input_addr

HARQ Accumulator Input Base Address.

Points to base address of the HARQ Accumulator buffer to be fetched into ETVPE before the new input stream.

3.5.5.4.2.2.13 uint32_t maple_tvpe_job_t::pfs_vec_sizes

User should use the pf_vector_size, ps_vector_size parameters .

Parity First and Second Buffer Size, ready to be written to memory.

3.5.5.4.2.2.14 uint16_t maple_tvpe_job_t::pf_vector_size

Parity First Buffer Size.

The actual buffer size of the PF vector during E-DCH HARQ with Separate Vectors input data structure.

3.5.5.4.2.2.15 uint16_t maple_tvpe_job_t::ps_vector_size

Parity Second Buffer Size.

The actual buffer size of the PS vector during E-DCH HARQ with Separate Vectors input data structure.

3.5.5.4.2.2.16 void* maple_tvpe_job_t::harq_acc_output_addr

HARQ Accumulator Output Base Address.

3.5.5.4.2.2.17 uint32_t maple_tvpe_job_t::fourth_flags

User should use a mix of the following flags: TVPE_AQC_AUTOSTOP, TVPE_CRC_AUTOSTOP, T↩ VPE_CRC_SCRC, TVPE_POLARITY, TVPE_HARQ_EN, TVPE_DOBSY, TVPE_DOBSI, TVPE_C↩ RC_REFLECT.

3.5.5.4.2.2.18 uint8_t maple_tvpe_job_t::bd_index

Job's index in the BD ring.

Will be assigned by the driver

3.5.5.4.3 Macro Definition Documentation

3.5.5.4.3.1 #define MAPLE_TVPE_CMD_SET_VITERBI_POLY (0x00000200 | COP_LLD_COMMAND)

Sets Maple eTvpe to Viterbi Polynomial mode.

3.5.5.4.3.2 #define MAPLE_TVPE_CMD_GET_PE_ACCUMULATOR (0x00000300 | COP_LLD_COMMAND)

Sample a PE accumulator, The delta between two samples of the PEs accumulator indicates the number of maple cycles the specific PE was in processing state (including IO u-code and HW duration).

3.5.5.4.3.3 #define MAPLE_TVPE_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL

Polling channel control command.

Use MAPLE_PE_CH_CMD_RX_POLL for all PEs

3.5.5.4.3.4 #define TVPE_BD_MANUAL_ACTIVATION 0x80000000

Prevent the driver from dispatching the job, MAPLE will wait until the user calls a Channel Ctrl Command to activate the job.

3.5.5.4.3.5 #define TVPE_INT_EN 0x10000000

Interrupt enable.

3.5.5.4.3.6 #define TVPE_TURBO_LMAP_EN 0x08000000

LogMAP algorithm; Valid only for Turbo.

3.5.5.4.3.7 #define TVPE_3GLTE_EN 0x04000000

Using this bit defined 3GLTE; not defines UMTS or WiMAX.

3.5.5.4.3.8 #define TVPE_VITERBI_TBZE 0x00400000

Traceback from max path calculations.
Valid only for Viterbi

3.5.5.4.3.9 #define TVPE_ZTTB 0x00200000

Zero tail trellis.

3.5.5.4.3.10 #define TVPE_TURBO_CRC_EN 0x00100000

CRC check enable.
Valid only for Turbo

3.5.5.4.3.11 #define TVPE_TURBO_HAOE 0x00020000

HARQ Accumulator Output Enable.
Valid only for Turbo

3.5.5.4.3.12 #define TVPE_BLOCK_SIZE(*K*) (((uint32_t)((K) & 0x7FFF)) << 16)

Decoded block size [bits]; For Turbo - should't include tail bits.

3.5.5.4.3.13 #define TVPE_TURBO_LLMAP_LCF(*K*) (((K) & 0x0FE) << 8)

Linear LogMAP correction factor.
Valid only for Turbo decoding

3.5.5.4.3.14 #define TVPE_HOE 0x80000000

Hard output enable.

3.5.5.4.3.15 #define TVPE_TURBO_DPNC_EN 0x10000000

De-Puncturing.
Valid only during Turbo decoding; This flag stands for DPNC in BD.

3.5.5.4.3.16 #define TVPE_BUF_SIZE(K) ((K) & 0x3FFFF)

Input buffer size [bytes].

Valid values are 40-100K. Valid only for: Viterbi: STRUCT_PPCMS. Turbo: STRUCT_PPCMS, STRUCT_RATE_MATCHED

3.5.5.4.3.17 #define TVPE_TURBO_VEC_OFFSET(K) ((K) << 16)

Describes the relative offset between the vectors in the system memory.

3.5.5.4.3.18 #define TVPE_HARD_OUT_OFFSET(K) ((K) << 4)

Hard Output Offset.

Describes the bit offset of the first byte in which the first valid bit is to be written

3.5.5.4.3.19 #define TVPE_TURBO_LLR_OUT_SF(K) (K)

LLR Output Shift.

Determines which 8 bits of the internal 16 bit wide soft outputs are going to be output

3.5.5.4.3.20 #define TVPE_HARQ_IHBSA(K) (((K) & 0x7FFF) << 16)

Input HARQ Buffer Start Address; Describe the address location of the first bit of the input stream in the HARQ accumulator buffer.

3.5.5.4.3.21 #define TVPE_FIRST_TIME_HARQ 0x00008000

First Time HARQ; Indicates whether this HARQ buffer is the first one hence no HARQ accumulator exist nor should be supplied.

3.5.5.4.3.22 #define TVPE_HARQ_IHBSZ(K) ((K) & 0x7FFF)

Input HARQ Buffer Size; Describe the size of the current HARQ input stream into the accumulator.

3.5.5.4.3.23 #define TVPE_TURBO_HARQ_W1_EN 0x04000000

Weight 1 Enable.

3.5.5.4.3.24 #define TVPE_TURBO_HARQ_W2_EN 0x02000000

Weight 2 Enable.

3.5.5.4.3.25 #define TVPE_TURBO_HARQ_W3_EN 0x01000000

Weight 3 Enable.

3.5.5.4.3.26 #define TVPE_TURBO_HARQ_W1(K) ((K & 0xFF) << 16)

Weight 1; Valid only if TVPE_TURBO_HARQ_W1_EN is set.

3.5.5.4.3.27 #define TVPE_TURBO_HARQ_W2(*K*) ((K & 0xFF) << 8)

Weight 2; Valid only if TVPE_TURBO_HARQ_W2_EN is set.

3.5.5.4.3.28 #define TVPE_TURBO_HARQ_W3(*K*) (K & 0xFF)

Weight 3; Valid only if TVPE_TURBO_HARQ_W3_EN is set.

3.5.5.4.3.29 #define TVPE_CRC_REFLECT 0x80000000

CRC Reflect - set for CRC bits in descending order.

3.5.5.4.3.30 #define TVPE_AQC_AUTOSTOP 0x40000000

Aposteriori Quality stop criteria indication; the eTVPE stop the decoding automatically once the all the soft-decoded bits pass the [AQTH] threshold field.

3.5.5.4.3.31 #define TVPE_CRC_AUTOSTOP 0x20000000

CRC check stop criteria indication; The eTVPE stop the decoding automatically once the Hard Output bits pass CRC check.

Must not be set if TVPE_CRC_SCRC is set

3.5.5.4.3.32 #define TVPE_CRC_SCRC 0x10000000

Steady CRC stop criteria indication; The eTVPE stop the decoding automatically once CRC result is equal to CRC results from previous iteration.

Must not be set if TVPE_CRC_AUTOSTOPis set

3.5.5.4.3.33 #define TVPE_HARQ_EN 0x08000000

Indicated whether HARQ operation in eTVPE is enabled, hence which input data structures for the eTVPE are allowed.

3.5.5.4.3.34 #define TVPE_POLARITY 0x04000000

Determines the Polarity of input data to the eTVPE.

3.5.5.4.3.35 #define TVPE_DOBSY 0x02000000

Data out Byte order - set for ascending order.

3.5.5.4.3.36 #define TVPE_DOBSI 0x01000000

Data out Bit order - set for ascending order.

3.5.5.4.4 Enumeration Type Documentation

3.5.5.4.4.1 enum maple_tvpe_k_t

Viterbi Constraint Length (K).

Valid only for Viterbi decoding

3.5.5.4.4.2 enum maple_tvpe_rate_t

Rate 1/2, 1/3, 1/4.

Enumerator

TVPE_RATE_1_2 Viterbi Only.
TVPE_RATE_1_3 Viterbi and Turbo.
TVPE_RATE_1_4 Viterbi only.

3.5.5.4.4.3 enum maple_tvpe_type_t

Decoding Algorithm Type.

3.5.5.4.4.4 enum maple_tvpe_punc_t

Puncturing Scheme.

Valid only for Viterbi processing ([ALG]=1).

3.5.5.4.4.5 enum maple_tvpe_max_iter_t

Maximum number of iterations.

Valid only during Turbo decoding.

3.5.5.4.4.6 enum maple_tvpe_min_iter_t

Minimum number of iterations.

Valid only during Turbo decoding.

3.5.5.4.4.7 enum maple_tvpe_data_struct_t

Describes the possible input data structures in system memory.

Enumerator

TVPE_LTE_HARQ Supports Turbo: 3GLTE.
TVPE_WIMAX_HARQ Supports Turbo: WiMAX.
TVPE_EDCH_HARQ_MIXED Supports Turbo: UMTS.
TVPE_EDCH_HARQ_SEPARATE Supports Turbo: UMTS.
TVPE_SUB_BLK_INTRLV Supports Turbo: 3GLTE, WiMAX.

TVPE_UMTS_MIXED Supports Turbo: UMTS.
TVPE_SEPARATE_VECTORS Supports Turbo: 3GLTE, WiMAX.

3.5.5.4.4.8 enum maple_tvpe_crc_t

CRC Polynomials.

Enumerator

TVPE_CRC24_POLY0 Polynomial: D24 + D23 + D6 + D5 + D + 1.
TVPE_CRC24_POLY1 Polynomial: D24 + D23 + D18 + D17 + D14 + D11 + D10 + D7 + D6 + D5 + D4 + D3 + D + 1.
TVPE_CRC16_POLY0 Polynomial: D16 + D12 + D5 + 1.
TVPE_CRC16_POLY1 Polynomial: D16 + D15 + D2 + 1.

3.5.5.4.4.9 enum maple_tvpe_dre_t

Number of DRE engines.

Valid only during Turbo decoding.

Enumerator

TVPE_ONE_DRE Single DRE engine.
TVPE_TWO_DRE Two DRE engines.
TVPE_FOUR_DRE Four DRE engines.

3.5.5.4.4.10 enum maple_tvpe_mbus_prio_t

eTVPE MBUS Priority.

Enumerator

TVPE_MBUS_PRIORITY_0 The AXI Bus accesses related to that BD are initiated with priority 0.

TVPE_MBUS_PRIORITY_1 The AXI Bus accesses related to that BD are initiated with priority 1.

TVPE_MBUS_PRIORITY_2 The AXI Bus accesses related to that BD are initiated with priority 2.

TVPE_MBUS_PRIORITY_3 The AXI Bus accesses related to that BD are initiated with priority 3.

3.5.5.4.4.11 enum maple_tvpe_soft_output_t

Soft Output Enable.

Turbo decoder only.

3.5.5.4.4.12 enum maple_tvpe_vit_set_t

Viterbi Set.

Valid only for Viterbi decoding. Not suported on MSBA8100

Enumerator

TVPE_VIT_SET_1 Use MTVPVS1CxP to configure TVVPVG0CR and TVVPVG1CR.
TVPE_VIT_SET_2 Use MTVPVS2CxP to configure TVVPVG0CR and TVVPVG1CR.
TVPE_VIT_SET_3 Use MTVPVS3CxP to configure TVVPVG0CR and TVVPVG1CR.

3.5.5.4.4.13 enum tvpe_harq_hus_t

Enumerator

TVPE_TURBO_HARQ_NO_UP_SCL The Input data is not pre scaled before being added to the accumulator.
TVPE_TURBO_HARQ_UP_SCL_2 The Input data is up scaled by 2 before being added to the accumulator.
TVPE_TURBO_HARQ_UP_SCL_4 The Input data is up scaled by 4 before being added to the accumulator.
TVPE_TURBO_HARQ_UP_SCL_8 The Input data is up scaled by 8 before being added to the accumulator.

3.5.6 Maple FTPE API

3.5.6.1 Overview

```
MAPLE FTPE Initialization and Runtime API
```


```
MAPLE FTPE Initialization and Runtime API
```


Modules

- FTPE Initialization
- FTPE Runtime

3.5.6.2 FTPE Initialization

3.5.6.2.1 Overview

```
FTPE device initialization API
```

```
MapleB3 FTPE device initialization API
```

Data Structures

- struct ftpe_size_params_t
- struct ftpe_open_params_t

Macros

- #define FTPE_MAX_NUM_BD_FOR_DISPACTH 32
- #define FTPE_PARAM_DSS_NUM 6
- #define MAPLE_FTPE_BCIS_EN 0x00000008

Typedefs

- typedef maple_pe_init_params_t maple_ftpe_init_params_t
- typedef struct maple_pe_init_params_s maple_ftpe_init_params_s
- typedef maple_pe_ch_open_params_t maple_ftpe_ch_open_params_t

Enumerations

- enum maple_ftpe_buff_config_t { OUTPUT1_POST1_PRE1, OUTPUT2_POST0_PRE1, OUTPUT2_POST1_PRE0 , OUTPUT1_POST2_PRE0 }
- enum

Functions

- os_status mapleFtpeInitialize (maple_ftpe_init_params_s *init_params, unsigned int num_devices, os_status(*channel_dispatch)(void *channel, void *jobs, int *num_jobs), void(*channel_reap)(void *channel, void *maple))

FTPE Device Names and IDs

```
Device IDs for FTPE0 FTPE1 FTPE2;
Device Names for FTPE0 FTPE1 FTPE2;
```

- enum { FTPE_DEV_ID_0 = 0, FTPE_DEV_ID_1 = 1, FTPE_DEV_ID_2 = 2 }
- #define MAPLE_0_FTPE_0_NAME "FTPE00"
- #define MAPLE_0_FTPE_1_NAME "FTPE01"
- #define MAPLE_0_FTPE_2_NAME "FTPE02"
- #define MAPLE_1_FTPE_0_NAME "FTPE10"
- #define MAPLE_1_FTPE_1_NAME "FTPE11"
- #define MAPLE_1_FTPE_2_NAME "FTPE12"
- #define MAPLE_2_FTPE_0_NAME "FTPE20"
- #define MAPLE_2_FTPE_1_NAME "FTPE21"

3.5.6.2.2 Data Structure Documentation

3.5.6.2.2.1 struct ftpe_size_params_t

Parameters for initializing the iFFT (Data Size registers)

Used as a part of LLD ftpe_open_params_t

3.5.6.2.2.1.1 Field Documentation

3.5.6.2.2.1.2 uint32_t ftpe_size_params_t::ds0

DS0 Data Size 0.

3.5.6.2.2.1.3 uint32_t ftpe_size_params_t::ds1

DS1 Data Size 1.

3.5.6.2.2.1.4 uint32_t ftpe_size_params_t::ds2

DS2 Data Size 2.

3.5.6.2.2.1.5 uint32_t ftpe_size_params_t::ds3

DS3 Data Size 3.

3.5.6.2.2.1.6 uint32_t ftpe_size_params_t::ds4

DS4 Data Size 4.

3.5.6.2.2.1.7 uint32_t ftpe_size_params_t::ds5

DS5 Data Size 5.

3.5.6.2.2.2 struct ftpe_open_params_t

MAPLE FTPE LLD Device Open paramaters.

Data Fields

- void * maple_handle
- maple_pe_bd_priority_t maple_pe_bd_priority
- maple_pe_num_bd_t maple_pe_num_bd
- uint32_t ftpe_config
- ftpe_size_params_t(* ftpe_data_size_sets)[FTPE_PARAM_DSS_NUM]

3.5.6.2.2.2.1 Field Documentation

3.5.6.2.2.2.2 void* ftpe_open_params_t::maple_handle

Handle returned from osCopDeviceOpen() for MAPLE controller.

3.5.6.2.2.3 maple_pe_bd_priority_t ftpe_open_params_t::maple_pe_bd_priority

BD rings priority scheduling - only configured by MAPLE master.

3.5.6.2.2.4 maple_pe_num_bd_t ftpe_open_params_t::maple_pe_num_bd

The number of BD rings for each priority - only configured by MAPLE master.

3.5.6.2.2.5 uint32_t ftpe_open_params_t::ftpe_config

Parameter for global configurations register of the EFTPE.

See BC, BCIS in RM.

3.5.6.2.2.6 ftpe_size_params_t(* ftpe_open_params_t::ftpe_data_size_sets)[FTPE_PARAM_↩ DSS_NUM]

A pointer to array.

Parameters for initializing the iFFT (Data Size sets parameters). May set to NULL for HW defaults

3.5.6.2.3 Macro Definition Documentation

3.5.6.2.3.1 #define FTPE_MAX_NUM_BD_FOR_DISPACTH 32

MAPLE FTPE Channel constrain.

3.5.6.2.3.2 #define FTPE_PARAM_DSS_NUM 6

The number of Data Size Sets 0 - 5.

3.5.6.2.3.3 #define MAPLE_FTPE_BCIS_EN 0x00000008

Backward Compatible Input Scale; Used for open_params->ftpe_config.

3.5.6.2.3.4 #define MAPLE_0_FTPE_0_NAME "FTPE00"

FTPE 0 Name on MAPLE 0.

3.5.6.2.3.5 #define MAPLE_0_FTPE_1_NAME "FTPE01"

FTPE 1 Name on MAPLE 0.

3.5.6.2.3.6 #define MAPLE_0_FTPE_2_NAME "FTPE02"

FTPE 2 Name on MAPLE 0.

3.5.6.2.3.7 #define MAPLE_1_FTPE_0_NAME "FTPE10"

FTPE 0 Name on MAPLE 1.

3.5.6.2.3.8 #define MAPLE_1_FTPE_1_NAME "FTPE11"

FTPE 1 Name on MAPLE 1.

3.5.6.2.3.9 #define MAPLE_1_FTPE_2_NAME "FTPE12"

FTPE 2 Name on MAPLE 1.

3.5.6.2.3.10 #define MAPLE_2_FTPE_0_NAME "FTPE20"

FTPE 0 Name on MAPLE 2.

3.5.6.2.3.11 #define MAPLE_2_FTPE_1_NAME "FTPE21"

FTPE 1 Name on MAPLE 2.

3.5.6.2.4 Typedef Documentation

3.5.6.2.4.1 typedef maple_pe_init_params_t maple_ftpe_init_params_t

MAPLE FTPE initialization parameters type for multiple PEs.

3.5.6.2.4.2 typedef struct maple_pe_init_params_s maple_ftpe_init_params_s

MAPLE FTPE initialization parameters type for one PE.

3.5.6.2.4.3 typedef maple_pe_ch_open_params_t maple_ftpe_ch_open_params_t

MAPLE FTPE channel open parameters type.

3.5.6.2.5 Enumeration Type Documentation

3.5.6.2.5.1 enum maple_ftpe_buff_config_t

Buffers Configuration.

Used for open_params->ftpe_config

Enumerator

OUTPUT1_POST1_PRE1 One output buffer, one post multiplier buffer and one pre multiplier buffer.

OUTPUT2_POST0_PRE1 Double output buffer, one pre - multiplier buffer.

OUTPUT2_POST1_PRE0 Double output buffer, one post multiplier buffer.

OUTPUT1_POST2_PRE0 One output buffer, double post multiplier buffer.

3.5.6.2.5.2 anonymous enum

MAPLE FTPE steering bits pointers mapping.

Use it for accessing relevant maple_pe_ch_open_params_t.steering_bits[x]

3.5.6.2.5.3 anonymous enum

FTPE device ID.

Enumerator

FTPE_DEV_ID_0 FTPE 0 id.
FTPE_DEV_ID_1 FTPE 1 id.
FTPE_DEV_ID_2 FTPE 2 id.

3.5.6.2.6 Function Documentation

3.5.6.2.6.1 os_status mapleFtpeInitialize (maple_ftpe_init_params_s * *init_params,* unsigned int *num_devices,* os_status(*)(void *channel, void *jobs, int *num_jobs) *channel_dispatch,* void(*)(void *channel, void *maple) *channel_reap*)

Initializes the FTPE driver's structures

The driver can supply default MAPLE parameters for initialization. The user can override these parameters by specifying an alternative MAPLE parameters structure. Although this function is called by all cores, only the master core performs the initialization of the MAPLE registers.

Parameters

in	*init_params*	- MAPLE Initialization parameters. if NULL, default MAPLE parameters will be used.
in	*num_devices*	- Number of FTPE devices
in	*channel_↩ dispatch*	- Pointer to channel dispatch function.
in	*channel_reap*	- Pointer to channel reap function.

Returns

OS_SUCCESS

Warning

This function is generally called by osArchInitialize() as part of the kernel and drivers

3.5.6.3 FTPE Runtime

3.5.6.3.1 Overview

FTPE Runtime API

Data Structures

- struct maple_ftpe_mult_buff_params_t
- union maple_ftpe_job_extension_t
- struct maple_ftpe_job_t

MAPLE FTPE Channel Control Commands

- #define MAPLE_FTPE_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL

FTPE Device Control Commands

- #define MAPLE_FTPE_CMD_PRE_POST_UPDATE (0x00000300 | COP_LLD_COMMAND)
- #define MAPLE_FTPE_CMD_GET_PE_ACCUMULATOR (0x00000400 | COP_LLD_COMM↩ AND)

MAPLE eFTPE FLAGS for maple_ftpe_job_t.first_flags

- enum
- #define FTPE_BD_MANUAL_ACTIVATION 0x80000000
- #define FTPE_BD_CFG0_FC_CG 0x00080000
- #define FTPE_BD_CFG0_DSTZP(K) ((K) << 6)
- #define FTPE_BD_CFG0_OFFKB (0x1 << 0)
- #define FTPE_BD_CFG0_OFF16 (0x1 << 1)
- #define FTPE_BD_PACKED_EXP_STAT (0x1 << 2)
- #define FTPE_BD_CFG0_PRBIE (0x1 << 3)
- #define FTPE_BD_CFG0_SOUS (0x1 << 4)
- #define FTPE_BD_COHERENCY_MODE (0x1 << 5)
- #define FTPE_BD_CFG0_INT_EN 0x10000000

MAPLE eFTPE FLAGS for maple_ftpe_job_t.second_flags

- enum maple_ftpe_post_vec_t { FTPE_POST_MULTI_DISABLE = 0x00000000, FTPE_PO↩ ST_MULTI_EN_NO_UPDATE = 0x08000000, FTPE_POST_MULTI_EN_AND_UPDATE = 0x18000000 }
- enum maple_ftpe_pre_vec_t { FTPE_PRE_OPERATION_DISABLE = 0x00000000, FTPE_PRE↩ _OPERATION_EN_NO_UPDATE = 0x00000800, FTPE_PRE_OPERATION_EN_AND_UPD↩ ATE = 0x00001800 }
- enum maple_ftpe_post_scale_t
- enum maple_ftpe_extra_scale_t
- enum maple_ftpe_data_size_set_t { FTPE_DATA_SIZE_SET_GI = 0x00000000 }
- #define FTPE_BD_CFG1_AIUS 0x80000000
- #define FTPE_BD_CFG1_PROUR 0x40000000
- #define FTPE_BD_CFG1_PSMUR 0x20000000
- #define FTPE_BD_CFG1_ZERO_PAD 0x04000000
- #define FTPE_BD_CFG1_SINS 0x02000000
- #define FTPE_BD_CFG1_CPRE 0x01000000
- #define FTPE_BD_CFG1_GRDC 0x00080000
- #define FTPE_BD_CFG1_AFS 0x00040000
- #define FTPE_BD_CFG1_PSTME 0x00008000

- #define FTPE_BD_CFG1_CPIE 0x00004000
- #define FTPE_BD_CFG1_GR_EN 0x00002000
- #define FTPE_BD_CFG1_OVA_SCL 0x00000400
- #define FTPE_BD_CFG1_SCL_TYPE 0x00000200
- #define FTPE_BD_CFG1_ITE 0x00000100
- #define FTPE_BD_CFG1_GI_EN 0x00000080

MAPLE eFTPE FLAGS for maple_ftpe_job_t.third_flags

- enum maple_ftpe_axi_pr_t {
  FTPE_BD_AXI_PR_0 = 0x00000000, FTPE_BD_AXI_PR_1 = 0x00000001, FTPE_BD_AXI_↩
  PR_2 = 0x00000002,
  FTPE_BD_AXI_PR_3 = 0x00000003 }

MAPLE eFTPE FLAGS for maple_ftpe_job_t.fourth_flags

- #define FTPE_BD_CFG3_PM_SCL 0x80000000
- #define FTPE_BD_CFG3_USR_SCL_DWN_STG0(BITS) ((BITS) << 28)
- #define FTPE_BD_CFG3_USR_SCL_DWN_STG1(BITS) ((BITS) << 25)
- #define FTPE_BD_CFG3_USR_SCL_DWN_STG2(BITS) ((BITS) << 22)
- #define FTPE_BD_CFG3_USR_SCL_DWN_STG3(BITS) ((BITS) << 19)
- #define FTPE_BD_CFG3_USR_SCL_DWN_STG4(BITS) ((BITS) << 16)
- #define FTPE_BD_CFG3_USR_SCL_DWN_STG5(BITS) ((BITS) << 13)
- #define FTPE_BD_CFG3_IN_SCL(N) ((N) << 9)
- #define FTPE_BD_CFG3_ADP_OVA_SCL(N) (N)

MAPLE eFTPE FLAGS for maple_ftpe_job_t.fifth_flags

- #define FTPE_BD_CFG4_CPS(N) ((N) << 23)
- #define FTPE_BD_CFG4_BD_RPT(N) ((N) << 16)

3.5.6.3.2 Data Structure Documentation

3.5.6.3.2.1 struct maple_ftpe_mult_buff_params_t

MAPLE eFTPE Pre/Post Multiplication Buffer prameters.

See MAPLE_FTPE_CMD_PRE_POST_UPDATE

Data Fields

- void * pre_buff_ptr
- void * pst_buff_ptr
- uint16_t pre_buff_size
- uint16_t pst_buff_size
- uint8_t pst_steering_bits
- uint8_t pre_steering_bits

3.5.6.3.2.1.1 Field Documentation

3.5.6.3.2.1.2 void* maple_ftpe_mult_buff_params_t::pre_buff_ptr

Pre Operation buffer pointer; this pointer won't be virt to phys translated by driver.

3.5.6.3.2.1.3 void* maple_ftpe_mult_buff_params_t::pst_buff_ptr

Post Multiplication buffer pointer; this pointer won't be virt to phys translated by driver.

3.5.6.3.2.1.4 uint16_t maple_ftpe_mult_buff_params_t::pre_buff_size

Pre Multiplication buffer size.

3.5.6.3.2.1.5 uint16_t maple_ftpe_mult_buff_params_t::pst_buff_size

Post Multiplication buffer size.

3.5.6.3.2.1.6 uint8_t maple_ftpe_mult_buff_params_t::pst_steering_bits

Post multiplication buffer pointer steering bits Use the following flags to set steering_bits field MAPL↩ E_STEERING_DSP_CLUSTER_0, MAPLE_STEERING_DSP_CLUSTER_1, MAPLE_STEERING_↩ DSP_CLUSTER_2, MAPLE_STEERING_CHB, MAPLE_STEERING_MAPLE_TRG.

3.5.6.3.2.1.7 uint8_t maple_ftpe_mult_buff_params_t::pre_steering_bits

Pre multiplication buffer pointer steering bits Use the following flags to set steering_bits field MAPL↩ E_STEERING_DSP_CLUSTER_0, MAPLE_STEERING_DSP_CLUSTER_1, MAPLE_STEERING_↩ DSP_CLUSTER_2, MAPLE_STEERING_CHB, MAPLE_STEERING_MAPLE_TRG.

3.5.6.3.2.2 union maple_ftpe_job_extension_t

MAPLE eFTPE job's extension maple_ftpe_job_t->extension_ptr.

Data Fields

- uint32_t extension_header [0x9]
- uint32_t fcg:1
- uint32_t cg:1
- uint32_t fc_rpt:1
- uint32_t fc_pre:1
- uint32_t fc_in_sc:16
- void * fcb_ptr
- uint16_t fc_base_real
- uint16_t fc_base_img
- uint16_t fc_size
- uint16_t fc_step_size
- uint32_t fc_shift_real:24
- uint32_t next_num_pi:3
- uint32_t chip_offset:16
- uint32_t curr_num_pi:3
- uint32_t slot_gen_off:4

- uint32_t lss:1
- uint32_t ssn:24
- uint32_t code_length:16
- uint32_t slot_num:4
- void * preas_scl_ptr
- uint32_t fc_shift_img:24

3.5.6.3.2.2.1 Field Documentation

3.5.6.3.2.2.2 uint32_t maple_ftpe_job_extension_t::extension_header[0x9]

Used for faster headers setting/ writing.

3.5.6.3.2.2.3 uint32_t maple_ftpe_job_extension_t::fcg

Frequency Correction Generation.

3.5.6.3.2.2.4 uint32_t maple_ftpe_job_extension_t::cg

Code Generation.

3.5.6.3.2.2.5 uint32_t maple_ftpe_job_extension_t::fc_rpt

Frequency Correct Repeat.

3.5.6.3.2.2.6 uint32_t maple_ftpe_job_extension_t::fc_pre

Frequency Correction Pre Multiply.

3.5.6.3.2.2.7 uint32_t maple_ftpe_job_extension_t::fc_in_sc

Frequency Correction Input Step Counter.

3.5.6.3.2.2.8 void* maple_ftpe_job_extension_t::fcb_ptr

Frequency Correction Base value Pointer.

Valid only if FCG is enabled, FC_PTR is set and BD_RPT $>$ 0.

3.5.6.3.2.2.9 uint16_t maple_ftpe_job_extension_t::fc_base_real

Frequency Correction Base Value Real Part.

3.5.6.3.2.2.10 uint16_t maple_ftpe_job_extension_t::fc_base_img

Frequency Correction Base Value Imaginary Part.

3.5.6.3.2.2.11 uint16_t maple_ftpe_job_extension_t::fc_size

Frequency Correction Size.

3.5.6.3.2.2.12 uint16_t maple_ftpe_job_extension_t::fc_step_size

Frequency Correction Step Size.

3.5.6.3.2.2.13 uint32_t maple_ftpe_job_extension_t::fc_shift_real

Frequency Correction Shift Value Real Part.

3.5.6.3.2.2.14 uint32_t maple_ftpe_job_extension_t::next_num_pi

Next Number of Pilots.

3.5.6.3.2.2.15 uint32_t maple_ftpe_job_extension_t::chip_offset

Chip Offset.

3.5.6.3.2.2.16 uint32_t maple_ftpe_job_extension_t::curr_num_pi

Current Number of Pilots.

3.5.6.3.2.2.17 uint32_t maple_ftpe_job_extension_t::slot_gen_off

Slot Generation Off.

3.5.6.3.2.2.18 uint32_t maple_ftpe_job_extension_t::lss

Long Scrambling Sequence.

3.5.6.3.2.2.19 uint32_t maple_ftpe_job_extension_t::ssn

Scrambling Sequence Number.

3.5.6.3.2.2.20 uint32_t maple_ftpe_job_extension_t::code_length

Code Length.

3.5.6.3.2.2.21 uint32_t maple_ftpe_job_extension_t::slot_num

Slot Number.

3.5.6.3.2.2.22 void* maple_ftpe_job_extension_t::preas_scl_ptr

Pre Addition/Subtraction Scale Value if (BD_RPT=0) or Pre Addition/Subtraction Scale Value Pointer.

3.5.6.3.2.2.23 uint32_t maple_ftpe_job_extension_t::fc_shift_img

Frequency Correction Shift Value Imaginary Part.

3.5.6.3.2.3 struct maple_ftpe_job_t

MAPLE eFTPE job structure.

This structure should be passed to the LLD on the cop_job_handle.device_specific field in order for the LLD to build the necessary BD

Data Fields

- uint32_t first_flags
- uint32_t second_flags
- uint32_t third_flags
- uint32_t fourth_flags
- uint32_t fifth_flags
- uint32_t buffer_size
- void * input
- void * output
- void * bd_rpt_stat_ptr
- void * pre_operation_ptr
- void * post_multiplier_ptr
- void * per_resb_exp_ptr
- maple_ftpe_job_extension_t * extension_ptr
- uint8_t bd_index
- void * post_multiplication
- uint16_t post_mult_real
- uint16_t post_mult_img
- void * in_exp_ptr
- uint8_t input_exponent

3.5.6.3.2.3.1 Field Documentation

3.5.6.3.2.3.2 uint32_t maple_ftpe_job_t::first_flags

User should provide a mask of the following defines; Should be initialized to 0; Minimal sanity check for input parameters; FTPE_BD_CFG0_FC_CG, FTPE_BD_CFG0_INT_EN, FTPE_BD_CFG0_DST↩ ZP(K), FTPE_BD_CFG0_OFFKB, FTPE_BD_CFG0_OFF16, FTPE_BD_CFG0_SOUS, FTPE_BD_C↩ FG0_PRBIE, FTPE_PRE_MULTIPLICATION_OPERATION, FTPE_PRE_ADDITION_OPERATIO↩ N, FTPE_PRE_SUBTRACTION_OPERATION.

3.5.6.3.2.3.3 uint32_t maple_ftpe_job_t::second_flags

User should provide a mask of the following defines; Should be initialized to 0; Minimal sanity check for input parameters; FTPE_BD_CFG1_AIUS, FTPE_BD_CFG1_PRMUR, FTPE_BD_CFG1_PSMU↩ R, FTPE_BD_CFG1_ZERO_PAD, FTPE_BD_CFG1_SINS, FTPE_BD_CFG1_CPRE, FTPE_BD_CF↩ G1_GRDC, FTPE_BD_CFG1_AFS, FTPE_BD_CFG1_PSTME, FTPE_BD_CFG1_CPIE, FTPE_BD↩ _CFG1_GR_EN, FTPE_BD_CFG1_OVA_SCL, FTPE_BD_CFG1_SCL_TYPE, FTPE_BD_CFG1_IT↩ E, FTPE_BD_CFG1_GI_EN, maple_ftpe_post_vec_t, maple_ftpe_pre_vec_t, maple_ftpe_post_scale_t, maple_ftpe_extra_scale_t, maple_ftpe_data_size_set_t.

3.5.6.3.2.3.4 uint32_t maple_ftpe_job_t::third_flags

User should provide a mask of the following defines; Should be initialized to 0; Minimal sanity check for input parameters; maple_ftpe_mbus_pr_t.

3.5.6.3.2.3.5 uint32_t maple_ftpe_job_t::fourth_flags

User should provide a mask of the following defines; Should be initialized to 0; Minimal sanity check for input parameters; FTPE_BD_CFG3_PM_SCL, FTPE_BD_CFG3_USR_SCL_DWN_STG0(BITS), FTPE_BD_CFG3_USR_SCL_DWN_STG1(BITS), FTPE_BD_CFG3_USR_SCL_DWN_STG2(BITS), FTPE_BD_CFG3_USR_SCL_DWN_STG3(BITS), FTPE_BD_CFG3_USR_SCL_DWN_STG4(BITS), FTPE_BD_CFG3_USR_SCL_DWN_STG5(BITS), FTPE_BD_CFG3_IN_SCL(N), FTPE_BD_CFG3_ADP_OVA_SCL(N)

3.5.6.3.2.3.6 uint32_t maple_ftpe_job_t::fifth_flags

User should provide a mask of the following defines; Should be initialized to 0; Minimal sanity check for input parameters; FTPE_BD_CFG4_CPS(N), FTPE_BD_CFG4_BD_RPT(N)

3.5.6.3.2.3.7 uint32_t maple_ftpe_job_t::buffer_size

Transform Length ID.

3.5.6.3.2.3.8 void* maple_ftpe_job_t::post_multiplication

Valid only if the PSTME bit is enabled and BD_RPT $>$ 0.

3.5.6.3.2.3.9 uint16_t maple_ftpe_job_t::post_mult_real

Post Multiplication Imaginary.

3.5.6.3.2.3.10 uint16_t maple_ftpe_job_t::post_mult_img

Post Multiplication Real.

3.5.6.3.2.3.11 void* maple_ftpe_job_t::input

Address of input.

3.5.6.3.2.3.12 void* maple_ftpe_job_t::output

Address output.

3.5.6.3.2.3.13 void* maple_ftpe_job_t::bd_rpt_stat_ptr

BD Repeat Status Pointer.

3.5.6.3.2.3.14 void* maple_ftpe_job_t::pre_operation_ptr

Pre–Operation Pointer.

Points to the address in the system memory where the new data for pre-operation memory is located.

3.5.6.3.2.3.15 void* maple_ftpe_job_t::post_multiplier_ptr

Post-Multiplier Pointer.

Points to the address in the system memory where the new data for post-multiplier memory is located.

3.5.6.3.2.3.16 void* maple_ftpe_job_t::in_exp_ptr

INP_EXP_PTR field in BD.

3.5.6.3.2.3.17 uint8_t maple_ftpe_job_t::input_exponent

Input Exponent value (scale)

3.5.6.3.2.3.18 void* maple_ftpe_job_t::per_resb_exp_ptr

Per Resource Block Input Exponent Pointer.

3.5.6.3.2.3.19 maple_ftpe_job_extension_t* maple_ftpe_job_t::extension_ptr

A pointer to the extension needed for Code Generation feature.

3.5.6.3.2.3.20 uint8_t maple_ftpe_job_t::bd_index

Job's index in the BD ring.

Will be assigned by the driver

3.5.6.3.3 Macro Definition Documentation

3.5.6.3.3.1 #define MAPLE_FTPE_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL

Poll the channel for finished jobs.

Use MAPLE_PE_CH_CMD_RX_POLL instead

3.5.6.3.3.2 #define MAPLE_FTPE_CMD_PRE_POST_UPDATE (0x00000300 | COP_LLD_COMMAND)

Pre and Post Multiplication Memory update, use MAPLE_CMD_PCR_ACTIVATE_WITH_POLL right after it; Pass pointer to maple_ftpe_mult_buff_params_t as a parameter.

3.5.6.3.3.3 #define MAPLE_FTPE_CMD_GET_PE_ACCUMULATOR (0x00000400 | COP_LLD_COMMAND)

Sample a PE accumulator, The delta between two samples of the PEs accumulator indicates the number of maple cycles the specific PE was in processing state (including IO u-code and HW duration).

3.5.6.3.3.4 #define FTPE_BD_MANUAL_ACTIVATION 0x80000000

Prevent the driver from dispatching the job, MAPLE will wait until the user calls a Channel Ctrl Command to activate the job.

3.5.6.3.3.5 #define FTPE_BD_CFG0_FC_CG 0x00080000

Frequency Correction or Code Generation extension.

3.5.6.3.3.6 #define FTPE_BD_CFG0_DSTZP(*K*) ((K) << 6)

Data Size to Zero Padding.

3.5.6.3.3.7 #define FTPE_BD_CFG0_OFFKB (0x1 << 0)

Offset in KBytes; Valid only for FFT operation and when the BD repeat option is enabled; Will be copied by driver to BD[OBO].

3.5.6.3.3.8 #define FTPE_BD_CFG0_OFF16 (0x1 << 1)

Offset in 16 Bytes; Valid only for DFT/iDFT operation and when the BD repeat option is enabled; Will be copied by driver to BD[OBO].

3.5.6.3.3.9 #define FTPE_BD_PACKED_EXP_STAT (0x1 << 2)

Packed structure arrangement of the CMP_RSN and ADP_OVA_SCL_ST.

3.5.6.3.3.10 #define FTPE_BD_CFG0_PRBIE (0x1 << 3)

Per Resource Block Input Exponent; Will be copied by driver to BD[PRBIE_PTR].

3.5.6.3.3.11 #define FTPE_BD_CFG0_SOUS (0x1 << 4)

Short Output Structure indication; Will be copied by driver to BD[SOUS].

3.5.6.3.3.12 #define FTPE_BD_COHERENCY_MODE (0x1 << 5)

In coherency mode the uCode resets the owner bit only after the data is written to external buffer (unlike in regular mode in which the owner bit is reset after the DMA command, but the data may not be ready in the external buffet yet).

Will be copied by driver to BD[COHERENCY_MODE]

3.5.6.3.3.13 #define FTPE_BD_CFG0_INT_EN 0x10000000

Interrupt enable.

3.5.6.3.3.14 #define FTPE_BD_CFG1_AIUS 0x80000000

Adaptive Input Scaling.

Valid only in Adaptive Scale mode (SCL_TYPE=0).

3.5.6.3.3.15 #define FTPE_BD_CFG1_PROUR 0x40000000

Pre Operation Repeat.

Valid only if BD repeat (BD_RPT > 0).

3.5.6.3.3.16 #define FTPE_BD_CFG1_PSMUR 0x20000000

Post Multiplier Repeat.

Valid only if BD repeat (BD_RPT > 0).

3.5.6.3.3.17 #define FTPE_BD_CFG1_ZERO_PAD 0x04000000

Zero Padding.

Indicates the eFTPE to add zero padding to the input data.

3.5.6.3.3.18 #define FTPE_BD_CFG1_SINS 0x02000000

Short Input Structure.

3.5.6.3.3.19 #define FTPE_BD_CFG1_CPRE 0x01000000

Cyclic Prefix Removal Enable.

Valid only if BD Repeat option is enabled.

3.5.6.3.3.20 #define FTPE_BD_CFG1_GRDC 0x00080000

Guard Removal of DC.

Valid only if the GR bit of the eFTPE BD is set.

3.5.6.3.3.21 #define FTPE_BD_CFG1_AFS 0x00040000

Align to First Scale.

Valid only if BD_RPT > 0, SCL_TYPE = 0

3.5.6.3.3.22 #define FTPE_BD_CFG1_PSTME 0x00008000

Post Multiplication Enable.

If set, the eFTPE multiplies all FFT/DFT output samples with the complex value as described in PSTM↩ _REAL and PSTM_IMG of the eFTPE BD.

3.5.6.3.3.23 #define FTPE_BD_CFG1_CPIE 0x00004000

Cyclic Prefix Insertion Enable.

3.5.6.3.3.24 #define FTPE_BD_CFG1_GR_EN 0x00002000

Guard Removal.

Guard Removal output data mode enable. Valid only if (ITE = 0).

3.5.6.3.3.25 #define FTPE_BD_CFG1_OVA_SCL 0x00000400

Overall Scaling.

Valid only for adaptive scaling (SCL_TYPE=0).

3.5.6.3.3.26 #define FTPE_BD_CFG1_SCL_TYPE 0x00000200

Scaling Type.

User defined scaling is performed.

3.5.6.3.3.27 #define FTPE_BD_CFG1_ITE 0x00000100

Inverse Transform Enable.

3.5.6.3.3.28 #define FTPE_BD_CFG1_GI_EN 0x00000080

Guard Insertion.

Guard Insertion input data mode enable.

3.5.6.3.3.29 #define FTPE_BD_CFG3_PM_SCL 0x80000000

User scaling after pre-multiplication.

Valid only if the SCL_TYPE field of the BD is set.

3.5.6.3.3.30 #define FTPE_BD_CFG3_USR_SCL_DWN_STG0(*BITS*) ((BITS) << 28)

The results of stage 0 are scaled down (shift right) by BITS.

3.5.6.3.3.31 #define FTPE_BD_CFG3_USR_SCL_DWN_STG1(*BITS*) ((BITS) << 25)

The results of stage 1 are scaled down (shift right) by BITS.

3.5.6.3.3.32 #define FTPE_BD_CFG3_USR_SCL_DWN_STG2(*BITS*) ((BITS) << 22)

The results of stage 2 are scaled down (shift right) by BITS.

3.5.6.3.3.33 #define FTPE_BD_CFG3_USR_SCL_DWN_STG3(*BITS*) ((BITS) << 19)

The results of stage 3 are scaled down (shift right) by BITS.

3.5.6.3.3.34 #define FTPE_BD_CFG3_USR_SCL_DWN_STG4(*BITS*) ((BITS) << 16)

The results of stage 4 are scaled down (shift right) by BITS.

3.5.6.3.3.35 #define FTPE_BD_CFG3_USR_SCL_DWN_STG5(*BITS*) ((BITS) << 13)

The results of stage 5 are scaled down (shift right) by BITS.

3.5.6.3.3.36 #define FTPE_BD_CFG3_IN_SCL(*N*) ((N) << 9)

the input data is scaled up (shift left) by N before stage 0

3.5.6.3.3.37 #define FTPE_BD_CFG3_ADP_OVA_SCL(*N*) (N)

The overall needed scaling is N.

3.5.6.3.3.38 #define FTPE_BD_CFG4_CPS(*N*) ((N) << 23)

Cyclic Prefix Size.

Valid only if the CPIE bit or CPRE bit are enabled.

3.5.6.3.3.39 #define FTPE_BD_CFG4_BD_RPT(*N*) ((N) << 16)

BD Repeat.

This field allows the use of the same BD, for multiple jobs.

3.5.6.3.4 Enumeration Type Documentation

3.5.6.3.4.1 anonymous enum

Pre-Operation Type.

Will be copied by driver to BD[INP_EXP_PTR]

3.5.6.3.4.2 enum maple_ftpe_post_vec_t

Post-Multiplier Vector enable / disable.

Enumerator

FTPE_POST_MULTI_DISABLE Post-Multiplier is disabled.

FTPE_POST_MULTI_EN_NO_UPDATE Post-Multiplier is enabled. The eFTPE uses the post-multiplier memory without updating it.

FTPE_POST_MULTI_EN_AND_UPDATE Post-Multiplier is enabled. The MAPLE update the post-multiplier memory with new data.

3.5.6.3.4.3 enum maple_ftpe_pre_vec_t

Pre Operation enable/ disable.

Enumerator

FTPE_PRE_OPERATION_DISABLE Pre-Operation is disabled.

FTPE_PRE_OPERATION_EN_NO_UPDATE Pre-Operation is enabled. The eFTPE uses the pre-operation memory without updating it.

FTPE_PRE_OPERATION_EN_AND_UPDATE Pre-Operation is enabled. The MAPLE update the pre-operation memory with new data.

3.5.6.3.4.4 enum maple_ftpe_post_scale_t

Post Multiplier Scaling.

Valid during only if (SCL_TYPE=1) and (PSTMV=1).

3.5.6.3.4.5 enum maple_ftpe_extra_scale_t

Extra Scaling.

Valid only if Adaptive Scaling is enabled and no if Overall Scaling is applied.

3.5.6.3.4.6 enum maple_ftpe_data_size_set_t

Data Size Set.

Indicates the MAPLE which set of Data Size parameters.

Enumerator

FTPE_DATA_SIZE_SET_GI Data Size register in eFTPE are assumed to be updated. Valid only if GI is set.

3.5.6.3.4.7 enum maple_ftpe_axi_pr_t

MBus Priority.

Enumerator

FTPE_BD_AXI_PR_0 The AXI accesses related to that BD are initiated with priority 0.

FTPE_BD_AXI_PR_1 The AXI accesses related to that BD are initiated with priority 1.

FTPE_BD_AXI_PR_2 The AXI accesses related to that BD are initiated with priority 2.

FTPE_BD_AXI_PR_3 The AXI accesses related to that BD are initiated with priority 3.

3.5.7 Maple EQPE API

3.5.7.1 Overview

```
MAPLE EQPE Initialization and Runtime API
```

Modules

- EQPE Initialization
- EQPE Runtime

3.5.7.2 EQPE Initialization

3.5.7.2.1 Overview

EQPE device initialization API

Data Structures

- struct eqpe_open_params_t

Typedefs

- typedef maple_pe_init_params_t maple_eqpe_init_params_t
- typedef struct maple_pe_init_params_s maple_eqpe_init_params_s
- typedef maple_pe_ch_open_params_t maple_eqpe_ch_open_params_t

Enumerations

- enum

Functions

- os_status mapleEqpeInitialize (maple_eqpe_init_params_s *init_params, unsigned int num_devices, os_status(*channel_dispatch)(void *channel, void *jobs, int *num_jobs), void(*channel_reap)(void *channel, void *maple))

MAPLE EQPE Device Names and IDs

- #define EQPE_DEV_ID_0 0
- #define EQPE_DEV_ID_1 1
- #define EQPE_DEV_ID_2 2
- #define EQPE_DEV_ID_3 3
- #define EQPE_DEV_NAME_0 "EQPE0"
- #define EQPE_DEV_NAME_1 "EQPE1"
- #define EQPE_DEV_NAME_2 "EQPE2"
- #define EQPE_DEV_NAME_3 "EQPE3"

3.5.7.2.2 Data Structure Documentation

3.5.7.2.2.1 struct eqpe_open_params_t

EQPE open device LLD parameters.

Data Fields

- void * maple_handle
- maple_pe_bd_priority_t maple_pe_bd_priority
- maple_pe_num_bd_t maple_pe_num_bd

3.5.7.2.2.1.1 Field Documentation

3.5.7.2.2.1.2 void* eqpe_open_params_t::maple_handle

Handle returned from osCopDeviceOpen() for MAPLE controller.

3.5.7.2.2.1.3 maple_pe_bd_priority_t eqpe_open_params_t::maple_pe_bd_priority

BD rings priority scheduling - only configured by MAPLE master.

3.5.7.2.2.1.4 maple_pe_num_bd_t eqpe_open_params_t::maple_pe_num_bd

The number of BD rings for each priority - only configured by MAPLE master.

3.5.7.2.3 Macro Definition Documentation

3.5.7.2.3.1 #define EQPE_DEV_ID_0 0

EQPE 0 ID.

3.5.7.2.3.2 #define EQPE_DEV_ID_1 1

EQPE 1 ID.

3.5.7.2.3.3 #define EQPE_DEV_ID_2 2

EQPE 2 ID.

3.5.7.2.3.4 #define EQPE_DEV_ID_3 3

EQPE 3 ID.

3.5.7.2.3.5 #define EQPE_DEV_NAME_0 "EQPE0"

Maple EQPE 0 device name.

3.5.7.2.3.6 #define EQPE_DEV_NAME_1 "EQPE1"

Maple EQPE 1 device name.

3.5.7.2.3.7 #define EQPE_DEV_NAME_2 "EQPE2"

Maple EQPE 2 device name.

3.5.7.2.3.8 #define EQPE_DEV_NAME_3 "EQPE3"

Maple EQPE 3 device name.

3.5.7.2.4 Typedef Documentation

3.5.7.2.4.1 typedef maple_pe_init_params_t maple_eqpe_init_params_t

MAPLE EQPE initialization parameters type for multiple PEs.

3.5.7.2.4.2 typedef struct maple_pe_init_params_s maple_eqpe_init_params_s

MAPLE EQPE initialization parameters type for one PE.

3.5.7.2.4.3 typedef maple_pe_ch_open_params_t maple_eqpe_ch_open_params_t

MAPLE EQPE channel open parameters type.

3.5.7.2.5 Enumeration Type Documentation

3.5.7.2.5.1 anonymous enum

MAPLE EQPE steering bits pointers mapping.

Use it for accessing relevant maple_pe_ch_open_params_t.steering_bits[x]

3.5.7.2.6 Function Documentation

3.5.7.2.6.1 os_status mapleEqpeInitialize (maple_eqpe_init_params_s * *init_params,* unsigned int *num_devices,* os_status(*)(void *channel, void *jobs, int *num_jobs) *channel_dispatch,* void(*)(void *channel, void *maple) *channel_reap*)

Initializes the EQPE driver's structures

Parameters

in	*init_params*	- EQPE Initialization parameters
in	*num_devices*	- Number of EQPE devices
in	*channel_dispatch*	- Pointer to channel dispatch function

in	*channel_reap*	- Pointer to channel reap function

Returns

OS_SUCCESS

Warning

This function is generally called by osArchInitialize() as part of the kernel and drivers

3.5.7.3 EQPE Runtime

3.5.7.3.1 Overview

EQPE Runtime API

Modules

- MAPLE EQPE BD

Macros

- #define EQPE_MAX_NUM_BD_FOR_DISPACTH 8

EQPE Device Control Commands

- enum maple_eqpe_ecc_event_flag_t {
  MAPLE_EQPE_ECC_ERR = 0x80000000, MAPLE_EQPE_ECC_SCU = 0x00002000, MAPLE↩
  _EQPE_ECC_IDU = 0x00001000,
  MAPLE_EQPE_ECC_NBB = 0x00000800, MAPLE_EQPE_ECC_MOS = 0x00000400, MAPL↩
  E_EQPE_ECC_MO = 0x00000200,
  MAPLE_EQPE_ECC_CMD = 0x00000100, MAPLE_EQPE_ECC_ISCL = 0x00000080, MAPL↩
  E_EQPE_ECC_Y = 0x00000040,
  MAPLE_EQPE_ECC_F = 0x00000020, MAPLE_EQPE_ECC_S = 0x00000010, MAPLE_EQP↩
  E_ECC_W = 0x00000008,
  MAPLE_EQPE_ECC_H = 0x00000004, MAPLE_EQPE_ECC_OSCL = 0x00000002, MAPLE_↩
  EQPE_ECC_OUT = 0x00000001 }
- #define MAPLE_EQPE_CMD_GET_PE_ACCUMULATOR (0x00000001 | COP_LLD_COMM↩
  AND)
- #define MAPLE_EQPE_CMD_GET_ECC_EVENT_REG (0x00000002 | COP_LLD_COMMA↩
  ND)

MAPLE EQPE Channel Control Commands.

- #define MAPLE_EQPE_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL

3.5.7.3.2 Macro Definition Documentation

3.5.7.3.2.1 #define EQPE_MAX_NUM_BD_FOR_DISPACTH 8

Maximal number of EQPE BD per dispatch.

Defines the maximal number of BD that can be dispatched with a single call to osCopChannelDispatch(). The larger the number the higher the stack consumption is in the driver. Users may change this value, however it requires recompiling the drivers.

3.5.7.3.2.2 #define MAPLE_EQPE_CMD_GET_PE_ACCUMULATOR (0x00000001 | COP_LLD_COMMAND)

Sample a PE accumulator, The delta between two samples of the PEs accumulator indicates the number of maple cycles the specific PE was in processing state (including IO u-code and HW duration).

3.5.7.3.2.3 #define MAPLE_EQPE_CMD_GET_ECC_EVENT_REG (0x00000002 | COP_LLD_COMMAND)

Returns ECC events of the EQPE2 memories.

param should be a pointer to a uint32_t that will hold the return value. Use maple_eqpe_ecc_event_flag_t enum to retrieve information from the register

3.5.7.3.2.4 #define MAPLE_EQPE_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL

Pool the channel for finished jobs.

Use MAPLE_PE_CH_CMD_RX_POLL instead

3.5.7.3.3 Enumeration Type Documentation

3.5.7.3.3.1 enum maple_eqpe_ecc_event_flag_t

Enumerator

MAPLE_EQPE_ECC_ERR ECC Error. When set this bit indicates that a multiple-bit error has occurred in one of the EQPE2 memories.

MAPLE_EQPE_ECC_SCU SCU E Buffer ECC Error. When set this bit indicates that a multiple-bit error has occurred in this EQPE2 memory.

MAPLE_EQPE_ECC_IDU IDU-MIB FIFO ECC Error. When set this bit indicates that a multiple-bit error has occurred in this EQPE2 memory.

MAPLE_EQPE_ECC_NBB NV/BETA Buffer internal ECC Error or Output Scale Buffer external ECC Error. When set this bit indicates that a multiple-bit error has occurred at the NV/beta memory by EQPE2 read access or at the Output Scale memory by host read.

MAPLE_EQPE_ECC_MOS MIV Output Scale Buffer internal ECC Error or NV/BETA Buffer external ECC Error. When set this bit indicates that a multiple-bit error has occurred at the MIV output scale memory by EQPE2 read access or at the NV/beta memory by host read.

MAPLE_EQPE_ECC_MO MIV Output Buffer Internal ECC Error or MIV Output Scale Buffer

external ECC Error. When set this bit indicates that a multiple-bit error has occurred at the MIV output memory by EQPE2 read access or at the MIV output scale memory by host read.

MAPLE_EQPE_ECC_CMD CMD Buffer Internal ECC Error or MIV Output Scale Buffer external ECC Error. When set this bit indicates that a multiple-bit error has occurred at the Command memory by EQPE2 read access or at the MIV output memory by host read.

MAPLE_EQPE_ECC_ISCL Input Scale Buffer internal ECC Error or CMD Buffer external ECC Error. When set this bit indicates that a multiple-bit error has occurred at the Input Scale memory by EQPE2 read access or at the Command memory by host read.

MAPLE_EQPE_ECC_Y Y Buffer internal ECC Error or Input Scale Buffer external ECC Error. When set this bit indicates that a multiple-bit error has occurred at the Y memory by EQP↩ E2read access or at the Input Scale memory by host read.

MAPLE_EQPE_ECC_F F Buffer internal ECC Error or Y Buffer external ECC Error. When set this bit indicates that a multiple-bit error has occurred at the F memory by EQPE2 read access or at the Y memory by host read.

MAPLE_EQPE_ECC_S S Buffer internal ECC Error or F Buffer external ECC Error. When set this bit indicates that a multiple-bit error has occurred at the S memory by EQPE2 read access or at the F memory by host read.

MAPLE_EQPE_ECC_W W Buffer internal ECC Error or S Buffer external ECC Error. When set this bit indicates that a multiple-bit error has occurred at the W memory by EQPE2 read access or at the S memory by host read.

MAPLE_EQPE_ECC_H H Buffer internal ECC Error or W Buffer external ECC Error. When set this bit indicates that a multiple-bit error has occurred at the H memory by EQPE2 read access or at the W memory by host read.

MAPLE_EQPE_ECC_OSCL Output Scale Buffer internal ECC Error or H Buffer external EC↩ C Error. When set this bit indicates that a multiple-bit error has occurred at the Output Scale memory by EQPE2 read access or at the Y memory by host read.

MAPLE_EQPE_ECC_OUT Output Buffer ECC Error. When set this bit indicates that a multiple-bit error has occurred at the output memory.

3.5.7.3.4 MAPLE EQPE BD

3.5.7.3.4.1 Overview

MAPLE EQPE Buffer Descriptor.

Data Structures

- struct maple_eqpe_job_t

Macros

- #define EQPE_NUM_H_PTRS 2
- #define EQPE_NUM_F_PTRS 4
- #define EQPE_NUM_OUT_PTRS 4

Enumerations

- enum {
  EQPE_S_SCL_OFFSET, EQPE_H_SCL_OFFSET, EQPE_F0_SCL_OFFSET,
  EQPE_F1_SCL_OFFSET, EQPE_F2_SCL_OFFSET, EQPE_F3_SCL_OFFSET,
  EQPE_SCL_OFFSET_LAST }

Functions

- INLINE uint32_t EQPE_BD_ALIGN_VAL (int8_t align, unsigned int layer)

Flags and definitions used in maple_eqpe_job_t.first_flags

- enum { EQPE_BD_ALG_LNEQ = 0x00000000, EQPE_BD_ALG_MIV = 0x04000000 }
- enum { EQPE_BD_MATRIX_M_TYPE_HERMITIAN = 0x00000000, EQPE_BD_MATRIX_↩
  M_TYPE_COMPLEX_FULL = 0x02000000 }
- enum { EQPE_BD_M_SCL_TYPE_SINGLE_SHARED = 0x00000000, EQPE_BD_M_SCL_T↩
  YPE_SEPARATE = 0x01000000 }
- INLINE uint32_t EQPE_BD_LX (unsigned int num)
- INLINE uint32_t EQPE_BD_RX (unsigned int num)
- INLINE uint32_t EQPE_BD_COLS (unsigned int num)
- INLINE uint32_t EQPE_BD_ROWS (unsigned int rows)
- #define EQPE_BD_MANUAL_ACTIVATION 0x80000000
- #define EQPE_BD_INT_EN 0x10000000

Flags and definitions used in maple_eqpe_job_t.second_flags

- enum {
  EQPE_BD_LTC_0 = 0x00080000, EQPE_BD_LTC_1 = 0x00040000, EQPE_BD_LTC_2 = 0x00020000,
  EQPE_BD_LTC_3 = 0x00010000 }
- enum {
  EQPE_BD_LV0 = 0x00800000, EQPE_BD_LV1 = 0x00400000, EQPE_BD_LV2 = 0x00200000,
  EQPE_BD_LV3 = 0x00100000 }
- enum {
  EQPE_BD_ALIGN0 = 0x00008000, EQPE_BD_ALIGN1 = 0x00004000, EQPE_BD_ALIGN2 = 0x00002000,
  EQPE_BD_ALIGN3 = 0x00001000 }
- INLINE uint32_t EQPE_BD_C_SCL (int8_t scale)
- #define EQPE_BD_C_EN 0x00000100

Flags and definitions used in maple_eqpe_job_t.matrix_dim

- enum { EQPE_BD_S_SCL_TYPE_SINGLE_SHARED = 0x00000000, EQPE_BD_S_SCL_TY↩
  PE_SEPARATE = 0x00002000 }
- enum { EQPE_BD_MATRIX_S_TYPE_HERMITIAN = 0x00000000, EQPE_BD_MATRIX_S↩
  _TYPE_COMPLEX_FULL = 0x00001000 }
- enum {
  EQPE_AXI_PR_0 = 0x00000000, EQPE_AXI_PR_1 = 0x00000001, EQPE_AXI_PR_2 = 0x00000002,
  EQPE_AXI_PR_3 = 0x00000003 }
- enum { EQPE_BD_INTRP_II2 = 0x00000000, EQPE_BD_INTRP_EXT = 0x20000000, EQPE_↩

BD_INTRP_NONE = 0x30000000 }
- INLINE uint32_t EQPE_BD_W_COL (uint8_t cols)
- INLINE uint32_t EQPE_BD_W_ROW (uint8_t rows)
- INLINE uint32_t EQPE_BD_S_COL (uint8_t cols)
- INLINE uint32_t EQPE_BD_S_ROW (uint8_t rows)
- #define EQPE_BD_W_PER_RX 0x40000000

Flags and definitions used in maple_eqpe_job_t.low_rr/high_rr

- INLINE uint16_t EQPE_RR_SET_COLUMN (bool reduce, unsigned short column)
- INLINE uint32_t EQPE_BD_RR0 (uint16_t rr)
- INLINE uint32_t EQPE_BD_RR1 (uint16_t rr)
- #define EQPE_BD_RR2(rr) EQPE_BD_RR0(rr)
- #define EQPE_BD_RR3(rr) EQPE_BD_RR1(rr)

Flags and definitions used in maple_eqpe_job_t.matrix_size

- INLINE uint32_t EQPE_BD_W_SIZE (uint16_t size)
- INLINE uint32_t EQPE_BD_S_SIZE (uint16_t size)

Flags and definitions used in maple_eqpe_job_t.y_params

- INLINE uint32_t EQPE_BD_Y_OFFSET (uint16_t offset)
- INLINE uint32_t EQPE_BD_Y_GAP (uint16_t gap)

Flags and definitions used in maple_eqpe_job_t.out_offsets

- INLINE uint32_t EQPE_BD_OUT_OFFSET0 (uint8_t offset)
- INLINE uint32_t EQPE_BD_OUT_OFFSET1 (uint8_t offset)
- INLINE uint32_t EQPE_BD_OUT_OFFSET2 (uint8_t offset)
- INLINE uint32_t EQPE_BD_OUT_OFFSET3 (uint8_t offset)

Flags and definitions used in maple_eqpe_job_t.h_params

- INLINE uint32_t EQPE_BD_H_OFFSET (uint16_t offset)
- INLINE uint32_t EQPE_BD_H_GAP (uint16_t gap)
- #define EQPE_CONSECUTIVE_H 0x00000001

Flags and definitions used in maple_eqpe_job_t.w_params

- INLINE uint32_t EQPE_BD_W_OFFSET (uint16_t offset)
- INLINE uint32_t EQPE_BD_W_GAP (uint16_t gap)

Flags and definitions used in maple_eqpe_job_t.f_params

- INLINE uint32_t EQPE_BD_F_OFFSET (uint16_t offset)
- INLINE uint32_t EQPE_BD_F_GAP (uint16_t gap)

3.5.7.3.4.2 Data Structure Documentation

3.5.7.3.4.2.1 struct maple_eqpe_job_t

EQPE Job Descriptor.

This structure should be passed to the LLD on the cop_job_handle.device_specific field in order for the LLD to build the necessary BD

Data Fields

- uint32_t first_flags
- uint32_t second_flags
- uint32_t matrix_dim
- uint32_t low_rr
- uint32_t high_rr
- uint32_t align_val
- uint32_t axi_pr
- os_virt_ptr y_ptr
- uint32_t y_params
- os_virt_ptr h_ptr [EQPE_NUM_H_PTRS]
- uint32_t h_params
- os_virt_ptr s_ptr
- os_virt_ptr w_ptr
- uint32_t w_params
- os_virt_ptr f_ptr
- uint32_t f_params
- os_virt_ptr nb_ptr
- os_virt_ptr c_ptr
- os_virt_ptr scl_ba_ptr
- uint16_t scl_offsets [EQPE_SCL_OFFSET_LAST]
- os_virt_ptr y_scl_ptr
- os_virt_ptr out_ptr [EQPE_NUM_OUT_PTRS]
- os_virt_ptr out_scl_ptr
- uint8_t bd_index

3.5.7.3.4.2.2 Field Documentation

3.5.7.3.4.2.3 uint32_t maple_eqpe_job_t::first_flags

User should provide a mask of the following defines and enumerations; EQPE_BD_INT_EN, EQPE_↩ BD_M_TYPE, EQPE_BD_M_SCL_TYPE, EQPE_BD_ALG_XX, EQPE_BD_LX(), EQPE_BD_RX(), EQPE_BD_COLS(), EQPE_BD_ROWS(); LLD will provide a mask for relevant bits.

Minimal sanity check for input parameters.

3.5.7.3.4.2.4 uint32_t maple_eqpe_job_t::second_flags

User should provide a mask of the following defines and enumerations; EQPE_BD_C_EN, EQPE_BD↩ _LTVX, EQPE_BD_LTC_X, EQPE_BD_ALIGN(), EQPE_BD_C_SCL(); LLD will provide a mask for relevant bits.

Minimal sanity check for input parameters.

3.5.7.3.4.2.5 uint32_t maple_eqpe_job_t::matrix_dim

User should provide a mask of the following defines and enumerations; EQPE_AXI_PR_X, EQPE_BD↩ _INTRP_XX, EQPE_BD_S_TYPE, EQPE_BD_S_SCL_TYPE, EQPE_BD_W_ROW(), EQPE_BD_S↩ _COL(), EQPE_BD_S_ROW(); LLD will provide a mask for relevant bits.

Minimal sanity check for input parameters.

3.5.7.3.4.2.6 uint32_t maple_eqpe_job_t::low_rr

User should provide a mask of the following defines and enumerations; EQPE_BD_RR0(), EQPE_BD_↩ RR1(); Any bit can be set with EQPE_RR_SET_COLUMN()

3.5.7.3.4.2.7 uint32_t maple_eqpe_job_t::high_rr

User should provide a mask of the following defines and enumerations; EQPE_BD_RR2(), EQPE_BD_↩ RR3(); Any bit can be set with EQPE_RR_SET_COLUMN()

3.5.7.3.4.2.8 uint32_t maple_eqpe_job_t::align_val

User should provide a value based on EQPE_BD_ALIGN_VAL() for all layers.

3.5.7.3.4.2.9 uint32_t maple_eqpe_job_t::axi_pr

Task ID and AXI Bus Priority.

3.5.7.3.4.2.10 os_virt_ptr maple_eqpe_job_t::y_ptr

Y pointer, valid for LNEQ, MLEQ and TS modes only; must be 16 byte aligned.

3.5.7.3.4.2.11 uint32_t maple_eqpe_job_t::y_params

User should provide a value based on EQPE_BD_Y_OFFSET(), EQPE_BD_Y_GAP()

3.5.7.3.4.2.12 os_virt_ptr maple_eqpe_job_t::h_ptr[EQPE_NUM_H_PTRS]

For LNEQ modes H pointer #0: H pointer; A pointer to the base address in the system memory where the MAPLE-B3 expects to find the first reference input data; In case of external interpolation or non interpolation modes (INTRP=2,3), this pointer points to the only reference input data buffer.

For MIV mode H pointer #0: M pointer; A pointer to the base address in the system memory where the MAPLE-B3 expects to find the input matrices data buffer

H pointer #1: Valid only for Internal interpolation mode (INTRP=0); A pointer to the base address in the system memory where the MAPLE-B3 expects to find the second reference input data

3.5.7.3.4.2.13 uint32_t maple_eqpe_job_t::h_params

User should provide a value based on EQPE_BD_H_OFFSET(), EQPE_BD_H_GAP(), EQPE_CONS↩ ECUTIVE_H.

3.5.7.3.4.2.14 os_virt_ptr maple_eqpe_job_t::s_ptr

S pointer, valid only for LNEQ.

3.5.7.3.4.2.15 os_virt_ptr maple_eqpe_job_t::w_ptr

W Pointer, valid only for LNEQ and only when internal interpolation (INTRP=0) is used; must be 16 byte aligned.

3.5.7.3.4.2.16 uint32_t maple_eqpe_job_t::w_params

User should provide a value based on EQPE_BD_W_OFFSET(), EQPE_BD_W_GAP()

3.5.7.3.4.2.17 os_virt_ptr maple_eqpe_job_t::f_ptr

F Pointer; f_ptr is valid only for LNEQ mode (ALG=0) and only when LTCx is set; Feedback input data buffer of layer x to be canceled by the EQPE.

3.5.7.3.4.2.18 uint32_t maple_eqpe_job_t::f_params

User should provide a value based on EQPE_BD_F_OFFSET(), EQPE_BD_F_GAP()

3.5.7.3.4.2.19 os_virt_ptr maple_eqpe_job_t::nb_ptr

NV/BETA Pointer must be 4 byte aligned.

Valid only for LNEQ mode (ALG=0). A pointer to the base address in the system memory where the MAPLE-B3 should write the NV/BETA statistics output data buffer to be used by the demapper function

3.5.7.3.4.2.20 os_virt_ptr maple_eqpe_job_t::c_ptr

C Pointer, valid only for LNEQ mode and only when EQPE_BD_C_EN is set; must be 16 byte aligned.

3.5.7.3.4.2.21 os_virt_ptr maple_eqpe_job_t::scl_ba_ptr

Scale Pointers Base Address; must be 16 byte aligned.

3.5.7.3.4.2.22 uint16_t maple_eqpe_job_t::scl_offsets[EQPE_SCL_OFFSET_LAST]

Scale offsets, use the following enumerations for accessing it: EQPE_S_SCL_OFFSET, EQPE_H_SCL↩ _OFFSET, EQPE_F0_SCL_OFFSET, EQPE_F1_SCL_OFFSET, EQPE_F2_SCL_OFFSET, EQPE_F3↩ _SCL_OFFSET.

3.5.7.3.4.2.23 os_virt_ptr maple_eqpe_job_t::y_scl_ptr

MAPLE expects to find the Y scale samples vector at the address.

3.5.7.3.4.2.24 os_virt_ptr maple_eqpe_job_t::out_ptr[EQPE_NUM_OUT_PTRS]

For LNEQ mode OUT_POINTER #0: Output Pointer for output samples of layer #0.

Valid only if LV0 = 1; Output samples (mantissa) of layer #0; For MIV mode OUT_POINTER #0: Output pointer; Output samples (mantissa) of MIV processing

OUT_POINTER n (1 - 3): Output Pointer for output samples of layer n; Valid only for LNEQ mode and only if LV1 = 1;

3.5.7.3.4.2.25 os_virt_ptr maple_eqpe_job_t::out_scl_ptr

For LNEQ processing: Output Pointer for output scale values.

A pointer to the base address in the system memory where the MAPLE-B3 is to write the output scale values (exponents) . For MIV processing: Output scale pointer.A pointer to the base address in the system memory where the MAPLE-B3 is to write the output scale values (exponents) of MIV processing.

3.5.7.3.4.2.26 uint8_t maple_eqpe_job_t::bd_index

Job's index in the BD ring.

Will be assigned by the driver

3.5.7.3.4.3 Macro Definition Documentation

3.5.7.3.4.3.1 #define EQPE_BD_MANUAL_ACTIVATION 0x80000000

Prevent the driver from dispatching the job, MAPLE will wait until the user calls a Channel Ctrl Command to activate the job.

3.5.7.3.4.3.2 #define EQPE_BD_INT_EN 0x10000000

BD[INT_EN].

3.5.7.3.4.3.3 #define EQPE_BD_C_EN 0x00000100

BD[C_EN] - when used, enables EQPE use of external C matrix.

3.5.7.3.4.3.4 #define EQPE_BD_W_PER_RX 0x40000000

Interpolation Weight per RX.

Distinct weight for each received antenna.

3.5.7.3.4.3.5 #define EQPE_BD_RR2(*rr*) EQPE_BD_RR0(rr)

Calculates 32 bits word for setting BD[RR2].

3.5.7.3.4.3.6 #define EQPE_BD_RR3(*rr*) EQPE_BD_RR1(rr)

Calculates 32 bits word for setting BD[RR3].

3.5.7.3.4.3.7 #define EQPE_CONSECUTIVE_H 0x00000001

The H input vectors offsets are calculated according to H_OFFSET and H_GAP; Valid for LNEQ mode only.

3.5.7.3.4.3.8 #define EQPE_NUM_H_PTRS 2

The number of EQPE H pointers inside EQPE BD.

3.5.7.3.4.3.9 #define EQPE_NUM_F_PTRS 4

The number of EQPE F pointers inside EQPE BD.

3.5.7.3.4.3.10 #define EQPE_NUM_OUT_PTRS 4

The number of EQPE output pointers inside EQPE BD.

3.5.7.3.4.4 Enumeration Type Documentation

3.5.7.3.4.4.1 anonymous enum

Values for BD[ALG] - Indicates the required processing type of the job.

Enumerator

EQPE_BD_ALG_LNEQ Linear Equalization.
EQPE_BD_ALG_MIV Matrix Inversion processing.

3.5.7.3.4.4.2 anonymous enum

Values for BD[M_TYPE] - Matrix Type.

Valid only for MIV processing (ALG=1). Indicates if the input and output matrices are of Hermitian or Complex Full type.

Enumerator

EQPE_BD_MATRIX_M_TYPE_HERMITIAN The input and output matrices are Hermitian.
EQPE_BD_MATRIX_M_TYPE_COMPLEX_FULL The input and output matrices are Complex Full.

3.5.7.3.4.4.3 anonymous enum

Values for BD[M_SCL_TYPE] - M Matrix Scale Type.

Valid only for MIV processing (ALG=1). Indicates the scale type of the input matrices.

Enumerator

EQPE_BD_M_SCL_TYPE_SINGLE_SHARED Single shared scale value for all matrices in the job.
EQPE_BD_M_SCL_TYPE_SEPARATE Separate scale value for each input matrix.

3.5.7.3.4.4.4 anonymous enum

Values for BD[LTC] - Layer To Cancel.

Valid only for LNEQ processing

Enumerator

EQPE_BD_LTC_0 Perform cancellation of layer 0.
EQPE_BD_LTC_1 Perform cancellation of layer 1.
EQPE_BD_LTC_2 Perform cancellation of layer 2.
EQPE_BD_LTC_3 Perform cancellation of layer 3.

3.5.7.3.4.4.5 anonymous enum

Values for BD[LVx] - Valid Layer.

Enumerator

EQPE_BD_LV0 Indicates that layer #0 is valid and contains data in the current segment.
EQPE_BD_LV1 Indicates that layer #1 is valid and contains data in the current segment.
EQPE_BD_LV2 Indicates that layer #2 is valid and contains data in the current segment.
EQPE_BD_LV3 Indicates that layer #3 is valid and contains data in the current segment.

3.5.7.3.4.4.6 anonymous enum

Values for BD[ALIGNx] - Indicates the output alignment type for LNEQ processing, or the output alignment type for MIV processing.

Enumerator

EQPE_BD_ALIGN0 Indicates that layer #0 samples are aligned to a given exponent value, according to ALIGN_VAL0 field.
EQPE_BD_ALIGN1 Indicates that layer #1 samples are aligned to a given exponent value, according to ALIGN_VAL1 field.
EQPE_BD_ALIGN2 Indicates that layer #2 samples are aligned to a given exponent value, according to ALIGN_VAL2 field.
EQPE_BD_ALIGN3 Indicates that layer #3 samples are aligned to a given exponent value, according to ALIGN_VAL3 field.

3.5.7.3.4.4.7 anonymous enum

Values for BD[S_SCL_TYPE] - S Matrix Scale Type.

Valid for LNEQ processing only (ALG=0). Indicates the scale granularity of the S matrices.

Enumerator

EQPE_BD_S_SCL_TYPE_SINGLE_SHARED Single shared scale value for all input S matrices.
EQPE_BD_S_SCL_TYPE_SEPARATE Separate scale value for each input S matrix.

3.5.7.3.4.4.8 anonymous enum

Values for BD[S_TYPE] - Matrix Type.

Valid for LNEQ processing only (ALG=0). Indicates if the input and output matrices are of Hermitian or Complex Full type.

Enumerator

EQPE_BD_MATRIX_S_TYPE_HERMITIAN The input and output matrices are Hermitian.
EQPE_BD_MATRIX_S_TYPE_COMPLEX_FULL The input and output matrices are Complex Full.

3.5.7.3.4.4.9 anonymous enum

Axi Bus Priority BD[AXI_PR].

Valid only if the [AXI_PR_SCH] of the MAPLE_MODE parameter is set to 0x1 or 0x2.

Enumerator

EQPE_AXI_PR_0 The AXI Bus accesses related to that BD are initiated with priority 0x0.
EQPE_AXI_PR_1 The AXI Bus accesses related to that BD are initiated with priority 0x1.
EQPE_AXI_PR_2 The AXI Bus accesses related to that BD are initiated with priority 0x2.
EQPE_AXI_PR_3 The AXI Bus accesses related to that BD are initiated with priority 0x3.

3.5.7.3.4.4.10 anonymous enum

Values for BD[INTRP] - Interpolation mode.

Valid only for LNEQ mode only; Indicates the interpolation mode for the current job

Enumerator

EQPE_BD_INTRP_II2 Internal Interpolation with 2 references.
EQPE_BD_INTRP_EXT External Interpolation.
EQPE_BD_INTRP_NONE No Interpolation.

3.5.7.3.4.4.11 anonymous enum

Use it for accessing relevant maple_eqpe_job_t.scl_offsets[x].

Enumerator

EQPE_S_SCL_OFFSET Use it for accessing maple_eqpe_job_t.scl_offsets[EQPE_S_SCL_OFFSET].
EQPE_H_SCL_OFFSET Use it for accessing maple_eqpe_job_t.scl_offsets[EQPE_H_SCL_OFFSET].
EQPE_F0_SCL_OFFSET Use it for accessing maple_eqpe_job_t.scl_offsets[EQPE_F0_SCL_OFFSET].
EQPE_F1_SCL_OFFSET Use it for accessing maple_eqpe_job_t.scl_offsets[EQPE_F1_SCL_OFFSET].
EQPE_F2_SCL_OFFSET Use it for accessing maple_eqpe_job_t.scl_offsets[EQPE_F2_SCL_OFFSET].
EQPE_F3_SCL_OFFSET Use it for accessing maple_eqpe_job_t.scl_offsets[EQPE_F3_SCL_OFFSET].
EQPE_SCL_OFFSET_LAST Number of existing offsets.

3.5.7.3.4.5 Function Documentation

3.5.7.3.4.5.1 INLINE uint32_t EQPE_BD_LX (unsigned int *num*)

Calculates BD[LX]

For LNEQ indicates the number of the transmit layers. For MIV processing indicates the dimensions of the input matrix to invert

Parameters

in	*num*	- Number of the transmit layers or dimension of the input matrix.

Returns

32 bits word with BD[LX] calculated value

3.5.7.3.4.5.2 INLINE uint32_t EQPE_BD_RX (unsigned int *num*)

Calculates BD[RX]; Valid only for LNEQ and MLEQ processing.

Parameters

in	*num*	- Number of Receive antennas.

Returns

32 bits word with BD[RX] culculated value

3.5.7.3.4.5.3 INLINE uint32_t EQPE_BD_COLS (unsigned int *num*)

Calculates BD[COLS]; Valid only for LNEQ, MLEQ and TS modes.

Parameters

in	*num*	- Number of Columns.

Returns

32 bits word with BD[COLS] culculated value

3.5.7.3.4.5.4 INLINE uint32_t EQPE_BD_ROWS (unsigned int *rows*)

Calculates BD[ROWS]; Valid only for LNEQ, MLEQ and TS modes.

For LNEQ processing mode: number of rows in the sub-carrier grid. For MIV processing: number of matrix to invert in current job Note For LNEQ processing mode: (ROWS+1) must be a multiplication of 12 Note For LNEQ processing mode: Single column is not supported to command of 12 rows Note For MIV processing: If M_TYPE = 0 (Hermitian) - 1 to 509 indicating 2 to 510 matrices respectively. Note For MIV processing: If M_TYPE = 1 (Complex Full) - 1 to 253 indicating 2 to 254 matrices respectively.

Parameters

in	*rows*	- Number of rows/ number of matrix (Zero indexed, 1 means 2 rows).

Returns

32 bits word with BD[ROWS] culculated value

3.5.7.3.4.5.5 INLINE uint32_t EQPE_BD_C_SCL (int8_t *scale*)

Calculates 32 bits word for setting BD[C_SCL]

Valid only for LNEQ processing (ALG=0) and only if C_EN=1

Parameters

in	*scale*	- BD[C_SCL] value

Returns

32 bits word for setting maple_eqpe_job_t.scaling

3.5.7.3.4.5.6 INLINE uint32_t EQPE_BD_W_COL (uint8_t *cols*)

Calculates 32 bits word for setting BD[W_COL]

Parameters

in	*cols*	- BD[W_COL] value

Returns

32 bits word for setting maple_eqpe_job_t.matrix_dim

3.5.7.3.4.5.7 INLINE uint32_t EQPE_BD_W_ROW (uint8_t *rows*)

Calculates 32 bits word for setting BD[W_ROW]

Parameters

in	*rows*	- BD[W_ROW] value

Returns

32 bits word for setting maple_eqpe_job_t.matrix_dim

3.5.7.3.4.5.8 INLINE uint32_t EQPE_BD_S_COL (uint8_t *cols*)

Calculates 32 bits word for setting BD[S_COL]

Parameters

in	*cols*	- BD[S_COL] value

Returns

32 bits word for setting maple_eqpe_job_t.matrix_dim

3.5.7.3.4.5.9 INLINE uint32_t EQPE_BD_S_ROW (uint8_t *rows*)

Calculates 32 bits word for setting BD[S_ROW]

Parameters

in	*rows*	- BD[S_ROW] value

Returns

32 bits word for setting maple_eqpe_job_t.matrix_dim

3.5.7.3.4.5.10 INLINE uint16_t EQPE_RR_SET_COLUMN (bool *reduce,* unsigned short *column*)

Set rank reduction

Parameters

in	*reduce*	- reduction to the matching column
in	*column*	- column for rank reduction

Returns

16 bits word for setting column rank reduction

3.5.7.3.4.5.11 INLINE uint32_t EQPE_BD_RR0 (uint16_t *rr*)

Calculates 32 bits word for setting BD[RR0]

Parameters

in	*rr*	- BD[RR0] value

Returns

32 bits word for setting maple_eqpe_job_t.low_rr/high_rr

3.5.7.3.4.5.12 INLINE uint32_t EQPE_BD_RR1 (uint16_t *rr*)

Calculates 32 bits word for setting BD[RR1]

Parameters

in	*rr*	- BD[RR1] value

Returns

32 bits word for setting maple_eqpe_job_t.low_rr/high_rr

3.5.7.3.4.5.13 INLINE uint32_t EQPE_BD_ALIGN_VAL (int8_t *align*, unsigned int *layer*)

Used in maple_eqpe_job_t.align_val

Parameters

in	*align*	- Alignment required.
in	*layer*	- Layer number.

Returns

32 bits word for setting maple_eqpe_job_t.align_val

3.5.7.3.4.5.14 INLINE uint32_t EQPE_BD_W_SIZE (uint16_t *size*)

Calculates 32 bits word for setting BD[W_SIZE]

Parameters

in	*size*	- BD[W_SIZE] value

Returns

32 bits word for setting maple_eqpe_job_t.matrix_size

3.5.7.3.4.5.15 INLINE uint32_t EQPE_BD_S_SIZE (uint16_t *size*)

Calculates 32 bits word for setting BD[S_SIZE]

Parameters

in	*size*	- BD[S_SIZE] value

Returns

32 bits word for setting maple_eqpe_job_t.matrix_size

3.5.7.3.4.5.16 INLINE uint32_t EQPE_BD_Y_OFFSET (uint16_t *offset*)

Calculates 32 bits word for setting BD[Y_OFFSET]

Parameters

in	*offset*	- BD[Y_OFFSET] value

Returns

32 bits word for setting maple_eqpe_job_t.y_params

3.5.7.3.4.5.17 INLINE uint32_t EQPE_BD_Y_GAP (uint16_t *gap*)

Calculates 32 bits word for setting BD[Y_GAP]

Parameters

in	*gap*	- BD[Y_GAP] value

Returns

32 bits word for setting maple_eqpe_job_t.y_params

3.5.7.3.4.5.18 INLINE uint32_t EQPE_BD_OUT_OFFSET0 (uint8_t *offset*)

Calculates 32 bits word for setting BD[OUT_OFFSET0]

Parameters

in	*offset*	- BD[OUT_OFFSET0] value

Returns

32 bits word for setting maple_eqpe_job_t.out_offsets

3.5.7.3.4.5.19 INLINE uint32_t EQPE_BD_OUT_OFFSET1 (uint8_t *offset*)

Calculates 32 bits word for setting BD[OUT_OFFSET1]

Parameters

in	*offset*	- BD[OUT_OFFSET1] value

Returns

32 bits word for setting maple_eqpe_job_t.out_offsets

3.5.7.3.4.5.20 INLINE uint32_t EQPE_BD_OUT_OFFSET2 (uint8_t *offset*)

Calculates 32 bits word for setting BD[OUT_OFFSET2]

Parameters

in	*offset*	- BD[OUT_OFFSET2] value

Returns

32 bits word for setting maple_eqpe_job_t.out_offsets

3.5.7.3.4.5.21 INLINE uint32_t EQPE_BD_OUT_OFFSET3 (uint8_t *offset*)

Calculates 32 bits word for setting BD[OUT_OFFSET3]

Parameters

in	*offset*	- BD[OUT_OFFSET3] value

Returns

32 bits word for setting maple_eqpe_job_t.out_offsets

3.5.7.3.4.5.22 INLINE uint32_t EQPE_BD_H_OFFSET (uint16_t *offset*)

Calculates 32 bits word for setting BD[H_OFFSET]

Parameters

in	*offset*	- BD[H_OFFSET] value

Returns

32 bits word for setting maple_eqpe_job_t.h_params

3.5.7.3.4.5.23 INLINE uint32_t EQPE_BD_H_GAP (uint16_t *gap*)

Calculates 32 bits word for setting BD[H_GAP]

Parameters

in	*gap*	- BD[H_GAP] value

Returns

32 bits word for setting maple_eqpe_job_t.h_params

3.5.7.3.4.5.24 INLINE uint32_t EQPE_BD_W_OFFSET (uint16_t *offset*)

Calculates 32 bits word for setting BD[W_OFFSET]

Parameters

in	*offset*	- BD[W_OFFSET] value

Returns

32 bits word for setting maple_eqpe_job_t.w_params

3.5.7.3.4.5.25 INLINE uint32_t EQPE_BD_W_GAP (uint16_t *gap*)

Calculates 32 bits word for setting BD[W_GAP]

Parameters

in	*gap*	- BD[W_GAP] value

Returns

32 bits word for setting maple_eqpe_job_t.w_params

3.5.7.3.4.5.26 INLINE uint32_t EQPE_BD_F_OFFSET (uint16_t *offset*)

Calculates 32 bits word for setting BD[F_OFFSET]

Parameters

in	*offset*	- BD[F_OFFSET] value

Returns

32 bits word for setting maple_eqpe_job_t.f_params

3.5.7.3.4.5.27 INLINE uint32_t EQPE_BD_F_GAP (uint16_t *gap*)

Calculates 32 bits word for setting BD[F_GAP]

Parameters

in	*gap*	- BD[F_GAP] value

Returns

32 bits word for setting maple_eqpe_job_t.f_params

3.5.8 Maple DEPE API

3.5.8.1 Overview

MAPLE DEPE Initialization and Runtime API

Modules

- DEPE Initialization
- DEPE Runtime

3.5.8.2 DEPE Initialization

3.5.8.2.1 Overview

DEPE device initialization API

Data Structures

- struct depe_open_params_t

Typedefs

- typedef maple_pe_init_params_t maple_depe_init_params_t
- typedef struct maple_pe_init_params_s maple_depe_init_params_s
- typedef maple_pe_ch_open_params_t maple_depe_ch_open_params_t

Enumerations

- enum

Functions

- os_status mapleDepeInitialize (maple_depe_init_params_s *init_params, unsigned int num_devices, os_status(*channel_dispatch)(void *channel, void *jobs, int *num_jobs), void(*channel_reap)(void *channel, void *maple))

MAPLE DEPE Device Names and IDs

- #define DEPE_DEV_ID_0 0
- #define DEPE_DEV_ID_1 1
- #define MAPLE_0_DEPE_NAME "DEPE0"
- #define MAPLE_1_DEPE_NAME "DEPE1"

3.5.8.2.2 Data Structure Documentation

3.5.8.2.2.1 struct depe_open_params_t

DEPE open device LLD parameters.

Data Fields

- void * maple_handle
- maple_pe_bd_priority_t maple_pe_bd_priority
- maple_pe_num_bd_t maple_pe_num_bd

3.5.8.2.2.1.1 Field Documentation

3.5.8.2.2.1.2 void* depe_open_params_t::maple_handle

Handle returned from osCopDeviceOpen() for MAPLE controller.

3.5.8.2.2.1.3 maple_pe_bd_priority_t depe_open_params_t::maple_pe_bd_priority

BD rings priority scheduling - only configured by MAPLE master.

3.5.8.2.2.1.4 maple_pe_num_bd_t depe_open_params_t::maple_pe_num_bd

The number of BD rings for each priority - only configured by MAPLE master.

3.5.8.2.3 Macro Definition Documentation

3.5.8.2.3.1 #define DEPE_DEV_ID_0 0

DEPE 3 ID.

3.5.8.2.3.2 #define DEPE_DEV_ID_1 1

DEPE 1 ID.

3.5.8.2.3.3 #define MAPLE_0_DEPE_NAME "DEPE0"

Maple 0 DEPE device name.

3.5.8.2.3.4 #define MAPLE_1_DEPE_NAME "DEPE1"

Maple 1 DEPE device name.

3.5.8.2.4 Typedef Documentation

3.5.8.2.4.1 typedef maple_pe_init_params_t maple_depe_init_params_t

MAPLE DEPE initialization parameters type for multiple PEs.

3.5.8.2.4.2 typedef struct maple_pe_init_params_s maple_depe_init_params_s

MAPLE DEPE initialization parameters type for one PE.

3.5.8.2.4.3 typedef maple_pe_ch_open_params_t maple_depe_ch_open_params_t

MAPLE DEPE channel open parameters type.

3.5.8.2.5 Enumeration Type Documentation

3.5.8.2.5.1 anonymous enum

MAPLE DEPE steering bits pointers mapping.

Use it for accessing relevant maple_pe_ch_open_params_t.steering_bits[x]

3.5.8.2.6 Function Documentation

3.5.8.2.6.1 os_status mapleDepeInitialize (maple_depe_init_params_s * *init_params,* unsigned int *num_devices,* os_status(*)(void *channel, void *jobs, int *num_jobs) *channel_dispatch,* void(*)(void *channel, void *maple) *channel_reap*)

Initializes the DEPE driver's structures

Parameters

in	*init_params*	- DEPE Initialization parameters
in	*num_devices*	- Number of DEPE devices
in	*channel_↩ dispatch*	- Pointer to channel dispatch function
in	*channel_reap*	- Pointer to channel reap function

Returns

OS_SUCCESS

Warning

This function is generally called by osArchInitialize() as part of the kernel and drivers

3.5.8.3 DEPE Runtime

3.5.8.3.1 Overview

DEPE Runtime API

Modules

- MAPLE DEPE BD
- MAPLE DEPE Headers

Macros

- #define DEPE_MAX_NUM_BD_FOR_DISPACTH 8

DEPE Device Control Commands

- #define MAPLE_DEPE_CMD_GET_PE_ACCUMULATOR (0x00000001 | COP_LLD_COMM↩ AND)

MAPLE DEPE Channel Control Commands

- #define MAPLE_DEPE_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL

3.5.8.3.2 Macro Definition Documentation

3.5.8.3.2.1 #define DEPE_MAX_NUM_BD_FOR_DISPACTH 8

Maximal number of DEPE BD per dispatch.

Defines the maximal number of BD that can be dispatched with a single call to osCopChannelDispatch(). The larger the number the higher the stack consumption is in the driver.

Warning

Users may change this value, however it requires recompiling the drivers.

3.5.8.3.2.2 #define MAPLE_DEPE_CMD_GET_PE_ACCUMULATOR (0x00000001 | COP_LLD_COMMAND)

Sample a PE accumulator, The delta between two samples of the PEs accumulator indicates the number of maple cycles the specific PE was in processing state (including IO u-code and HW duration).

3.5.8.3.2.3 #define MAPLE_DEPE_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL

Pool the channel for finished jobs.

Use MAPLE_PE_CH_CMD_RX_POLL instead.

3.5.8.3.3 MAPLE DEPE BD

3.5.8.3.3.1 Overview

MAPLE DEPE Buffer Descriptor.

Modules

- MAPLE DEPE Headers

Data Structures

- struct maple_depe_job_t

Macros

- #define DEPE_BD_MANUAL_ACTIVATION 0x80000000
- #define DEPE_BD_INT_EN 0x10000000
- #define DEPE_BD_COE 0x00000001
- #define DEPE_TRANSPORT_BLOCK 0x00004000
- #define DFPE_SVE_SVE 0x00008000

Enumerations

- enum { DEPE_STD_WIMAX = 0x00000000, DEPE_STD_UMTS = 0x00000000, DEPE_STD_↩ 3GLTE = 0x02000000 }
- enum {

DEPE_NOH_1 = 0x00000000, DEPE_NOH_2 = 0x00000800, DEPE_NOH_3 = 0x00001000,
DEPE_NOH_4 = 0x00001800, DEPE_NOH_5 = 0x00002000, DEPE_NOH_6 = 0x00002800,
DEPE_NOH_7 = 0x00003000, DEPE_NOH_8 = 0x00003800, DEPE_NOH_9 = 0x00004000,
DEPE_NOH_10 = 0x00004800, DEPE_NOH_11 = 0x00005000, DEPE_NOH_12 = 0x00005800,
DEPE_NOH_13 = 0x00006000, DEPE_NOH_14 = 0x00006800, DEPE_NOH_15 = 0x00007000,
DEPE_NOH_16 = 0x00007800, DEPE_NOH_17 = 0x00008000, DEPE_NOH_18 = 0x00008800,
DEPE_NOH_19 = 0x00009000, DEPE_NOH_20 = 0x00009800, DEPE_NOH_21 = 0x0000A000,
DEPE_NOH_22 = 0x0000A800, DEPE_NOH_23 = 0x0000B000, DEPE_NOH_24 = 0x0000B800,
DEPE_NOH_25 = 0x0000C000, DEPE_NOH_26 = 0x0000C800, DEPE_NOH_27 = 0x0000D000,
DEPE_NOH_28 = 0x0000D800, DEPE_NOH_29 = 0x0000E000, DEPE_NOH_30 = 0x0000E800,
DEPE_NOH_31 = 0x0000F000, DEPE_NOH_32 = 0x0000F800 }

- enum { , DEPE_MB_PR_1 = 0x40000000, DEPE_MB_PR_2 = 0x80000000, DEPE_MB_PR_3 = 0xC0000000 }
- enum {
  DEPE_ESVO_128 = 0x00000000, DEPE_ESVO_256 = 0x10000000, DEPE_ESVO_512 = 0x20000000,
  DEPE_ESVO_1K = 0x30000000, DEPE_ESVO_2K = 0x40000000, DEPE_ESVO_4K = 0x50000000,
  DEPE_ESVO_8K = 0x60000000, DEPE_ESVO_16K = 0x70000000, DEPE_ESVO_32K = 0x80000000,
  DEPE_ESVO_64K = 0x90000000, DEPE_ESVO_128K = 0xA0000000 }

Functions

- INLINE uint32_t DEPE_IBS (unsigned int num)
- INLINE uint32_t DEPE_NOH (unsigned int num)
- INLINE uint32_t DEPE_OBS (unsigned int num)
- INLINE uint32_t DEPE_PFS (unsigned int num)
- INLINE uint32_t DEPE_PSS (unsigned int num)
- INLINE uint32_t DEPE_OBO (unsigned int obo)

3.5.8.3.3.2 Data Structure Documentation

3.5.8.3.3.2.1 struct maple_depe_job_t

DEPE Job Descriptor.

This structure should be passed to the LLD on the cop_job_handle.device_specific field in order for the LLD to build the necessary BD

Data Fields

- uint32_t first_flags
- void * input
- void * output
- void * header
- void * sts
- uint32_t aux_flags
- uint32_t sep_vector
- uint32_t rmna
- union {

} headers

- uint8_t bd_index

3.5.8.3.3.2.2 Field Documentation

3.5.8.3.3.2.3 uint32_t maple_depe_job_t::first_flags

User should provide an or mask of the following defines and enumerations; LLD will provide a mask for relevant bits; Minimal sanity check for input parameters; DEPE_BD_INT_EN, DEPE_STD_XXX, DEPE_IBS(), DEPE_NOH_XX, DEPE_OBS()

3.5.8.3.3.2.4 void* maple_depe_job_t::input

Input Buffer Address.

This field points to the input buffer location in system memory, where MAPLE is to fetch the data into the DEPE input buffer. The address pointer to system memory must be 4 bytes aligned

3.5.8.3.3.2.5 void* maple_depe_job_t::output

Output Buffer Address.

This field points to the output buffer location in system memory, where MAPLE is to write the data from the DEPE output buffer; The address pointer to the output buffer in system memory must be 4 bytes aligned

3.5.8.3.3.2.6 void* maple_depe_job_t::header

Header Buffer Address.

This field points to the DEPE Headers buffer location in the system memory if multiple Headers are included in this job ([NOH]>0); If single Header is included in this job ([NOH]=0), this field is optional: by clearing it to zero the MAPLE takes the Header from the BD depe_header; The address pointer to the output buffer in system memory must be 8 bytes aligned

3.5.8.3.3.2.7 void* maple_depe_job_t::sts

BD Status Pointer.

Points to an address in system memory where MAPLE is to write the RMNA status fields in case [NOH] > 0 or in case [NOH] = 0 and this field is not zeroed

3.5.8.3.3.2.8 uint32_t maple_depe_job_t::aux_flags

User should provide an or mask of the following defines and enumerations; LLD will provide a mask for relevant bits; Minimal sanity check for input parameters; DEPE_BD_COE, DEPE_MB_PR_X, DEPE_↩ LTE_OBO.

3.5.8.3.3.2.9 uint32_t maple_depe_job_t::sep_vector

User should provide an or mask of the following defines and enumerations; LLD will provide a mask for relevant bits; Minimal sanity check for input parameters; DEPE_ESVO_X, DFPE_SVE_SVE, DEPE_PFS(), DEPE_PSS(), DEPE_TRANSPORT_BLOCK.

3.5.8.3.3.2.10 uint32_t maple_depe_job_t::rmna

RMNA for LTE.

3.5.8.3.3.2.11 union { ... } maple_depe_job_t::headers

If BD[NOH] is set to 0 and BD[HBA] is set to NULL, the DEPE header can be included in the BD; Otherwise is is read from the address pointed at by BD[HBA].

3.5.8.3.3.2.12 uint8_t maple_depe_job_t::bd_index

Job's index in the BD ring.

Will be assigned by the driver

3.5.8.3.3.3 Macro Definition Documentation

3.5.8.3.3.3.1 #define DEPE_BD_MANUAL_ACTIVATION 0x80000000

used in maple_depe_job_t.first_flags to Prevent the driver from dispatching the job, MAPLE will wait until the user calls a Channel Ctrl Command to activate the job

3.5.8.3.3.3.2 #define DEPE_BD_INT_EN 0x10000000

BD[INT_EN] - used in maple_depe_job_t.first_flags.

3.5.8.3.3.3.3 #define DEPE_BD_COE 0x00000001

Concatenate Output Enable BD[COE] - used in maple_depe_job_t.aux_flags.

Valid only for single Header job (BD[NOH]=0).If set, the MAPLE merges the first 32 bits word of the DEPE output buffer with the first 32 bits word of the output buffer in the system memory pointed by BD[OBA], thus allowing concatenating the current job to a previous job in the system memory

3.5.8.3.3.3.4 #define DEPE_TRANSPORT_BLOCK 0x00004000

Transport Block BD[TB] - used in maple_depe_job_t.sep_vector.

3.5.8.3.3.3.5 #define DFPE_SVE_SVE 0x00008000

Separate Vectors Enable BD[SVE] - used in maple_depe_job_t.sep_vector.

Valid only for UMTS standard. Indicates the MAPLE that the Bit Collection is disabled hence the DEPE outputs 3 separate vectors and not single stream

3.5.8.3.3.4 Enumeration Type Documentation

3.5.8.3.3.4.1 anonymous enum

Values for BD[STD] - used in maple_depe_job_t.first_flags.

The selection between DEPE_STD_WIMAX and DEPE_STD_UMTS is based on MAPLE Mode Configuration 0 Parameter (MMC0P)

Enumerator

DEPE_STD_WIMAX BD[STD] is cleared (WiMAX)
DEPE_STD_UMTS BD[STD] is UMTS.
DEPE_STD_3GLTE BS[STD] is 3GLTE.

3.5.8.3.3.4.2 anonymous enum

Values for BD[NOH] - used in maple_depe_job_t.first_flags.

Describes the number of Headers related to this BD. If set to 0 and BD[HBA] is also set to 0, the MAPLE expect to find the DEPE Header inside the BD, else the MAPLE fetches the Header from system memory address pointed by BD[HBA]

Enumerator

DEPE_NOH_1 1 Header, may be in the BD provided that BD[HBA] is also set to 0
DEPE_NOH_2 2 Headers
DEPE_NOH_3 3 Headers
DEPE_NOH_4 4 Headers
DEPE_NOH_5 5 Headers
DEPE_NOH_6 6 Headers
DEPE_NOH_7 7 Headers
DEPE_NOH_8 8 Headers
DEPE_NOH_9 9 Headers
DEPE_NOH_10 10 Headers
DEPE_NOH_11 11 Headers
DEPE_NOH_12 12 Headers
DEPE_NOH_13 13 Headers
DEPE_NOH_14 14 Headers
DEPE_NOH_15 15 Headers
DEPE_NOH_16 16 Headers
DEPE_NOH_17 17 Headers
DEPE_NOH_18 18 Headers
DEPE_NOH_19 19 Headers
DEPE_NOH_20 20 Headers
DEPE_NOH_21 21 Headers
DEPE_NOH_22 22 Headers
DEPE_NOH_23 23 Headers

DEPE_NOH_24 24 Headers
DEPE_NOH_25 25 Headers
DEPE_NOH_26 26 Headers
DEPE_NOH_27 27 Headers
DEPE_NOH_28 28 Headers
DEPE_NOH_29 29 Headers
DEPE_NOH_30 30 Headers
DEPE_NOH_31 31 Headers
DEPE_NOH_32 32 Headers

3.5.8.3.3.4.3 anonymous enum

MBus Priority BD[MB_PR] - used in maple_depe_job_t.aux_flags.

Valid only if the [AXI_PR_SCH] of the MAPLE_MODE parameter is set to 0x1 or 0x2.

Enumerator

DEPE_MB_PR_1 The MBus accesses related to that BD are initiated with priority 0x0.
DEPE_MB_PR_2 The MBus accesses related to that BD are initiated with priority 0x1.
DEPE_MB_PR_3 The MBus accesses related to that BD are initiated with priority 0x2.

3.5.8.3.3.4.4 anonymous enum

Encoded Separate Vectors Offset BD[SVEO] - used in maple_depe_job_t.sep_vector.

Valid only for UMTS standard and when Separate vectors output data structure is enabled ([SVE] = 1). Indicates the MAPLE the offset between the 3 output vectors in the system memory

Enumerator

DEPE_ESVO_128 The offset between the vectors in system memory is 128 bits.
DEPE_ESVO_256 The offset between the vectors in system memory is 256 bits.
DEPE_ESVO_512 The offset between the vectors in system memory is 512 bits.
DEPE_ESVO_1K The offset between the vectors in system memory is 1K bits.
DEPE_ESVO_2K The offset between the vectors in system memory is 2K bits.
DEPE_ESVO_4K The offset between the vectors in system memory is 4K bits.
DEPE_ESVO_8K The offset between the vectors in system memory is 8K bits.
DEPE_ESVO_16K The offset between the vectors in system memory is 16K bits.
DEPE_ESVO_32K The offset between the vectors in system memory is 32K bits.
DEPE_ESVO_64K The offset between the vectors in system memory is 64K bits.
DEPE_ESVO_128K The offset between the vectors in system memory is 128K bits.

3.5.8.3.3.5 Function Documentation

3.5.8.3.3.5.1 INLINE uint32_t DEPE_IBS (unsigned int *num*)

Calculates BD[IBS] - used in maple_depe_job_t.first_flags

Takes any number between 8-1020 [bytes] and returns the proper setting for BD[IBS] and returns the number of 32 bits words which should be fetched by MAPLE in order to execute the current BD. If the current BD includes multiple Headers (BD[NOH]>0), the IBS should describe the input data size of all Headers

Parameters

in	*num*	- Any number between 8-1020 used for BD[IBS] calculation

Returns

32 bits word with BD[IBS] calculated value

3.5.8.3.3.5.2 INLINE uint32_t DEPE_NOH (unsigned int *num*)

Calculates BD[NOH] - used in maple_depe_job_t.first_flags

Takes any number between 1-32 and returns the proper setting for BD[NOH]

Parameters

in	*num*	- Any number between 1-32 used for BD[NOH] calculation

Returns

32 bits word with BD[IBS] calculated value

3.5.8.3.3.5.3 INLINE uint32_t DEPE_OBS (unsigned int *num*)

Calculates BD[OBS] - used in maple_depe_job_t.first_flags

Describes the total number of 32 bits words which should be written by MAPLE from DEPE output buffer and into the system memory on job completion. If the current BD includes multiple Headers (BD[NO↩H]>0), the BD[OBS] should describe the output data size of all Headers. For 3GLTE standard this field is ignored. For UMTS Separate Vectors this field indicates the size of the systematic vector only. Clearing it to zero indicates MAPLE that this BD is part of UMTS TB processing

Parameters

in	*num*	- Any number between 0x8-0x1600 used for BD[OBS] calculation, input in bytes.

Returns

32 bits word with BD[OBS] culculated value

3.5.8.3.3.5.4 INLINE uint32_t DEPE_PFS (unsigned int *num*)

Calculates Parity First Size BD[PFS] - used in maple_depe_job_t.sep_vector

Valid only for UMTS standard and when Separate vectors output data structure is enabled ([SVE] = 1). Describes the total number of 32 bits words which should be written by MAPLE from DEPE output buffer and into the system memory in order to output the parity first vector from DEPE output buffer

Parameters

in	*num*	- Parity First Size in bytes.

Returns

32 bits word with BD[PFS] calculated value

3.5.8.3.3.5.5 INLINE uint32_t DEPE_PSS (unsigned int *num*)

Calculates Parity Second Size BD[PSS] - used in maple_depe_job_t.sep_vector

Valid only for UMTS standard and when Separate vectors output data structure is enabled ([SVE] = 1). Describes the total number of 32 bits words which should be written by MAPLE from DEPE output buffer and into the system memory in order to output the parity second vector from DEPE output buffer

Parameters

in	*num*	- Parity Second Size in bytes.

Returns

32 bits word with BD[PSS] calculated value

3.5.8.3.3.5.6 INLINE uint32_t DEPE_OBO (unsigned int *obo*)

Calculates BD[OBO] - used in maple_depe_job_t.aux_flags

Output Buffer Offset. Offset of the first bit of the encoded block in the DEPE output buffer. Used for concatenating the current CB with previous CB from the same transport block in system memory

Parameters

in	*obo*	- Output Buffer Offset.

Returns

32 bits word with BD[OBO] calculated value

Warning

Should be initialized with the same value as the header OBO in case of LTE only

3.5.8.3.3.5.7 MAPLE DEPE Headers

3.5.8.3.3.5.8 Overview

MAPLE DEPE Header Structures.

Data Structures

- union depe_3glte_header_t
- union depe_wimax_header_t
- union depe_wimax16m_header_t
- union depe_umts_header_t

Enumerations

- enum {
  DEPE_HEADER_1, DEPE_HEADER_2, DEPE_HEADER_3,
  DEPE_HEADER_4, DEPE_HEADER_5, DEPE_HEADER_6,
  DEPE_HEADER_7, DEPE_HEADER_8, DEPE_NUM_HEADERS }

3.5.8.3.3.5.9 Data Structure Documentation

3.5.8.3.3.5.10 union depe_3glte_header_t

DEPE Header Structure for 3GLTE.

The DEPE Header Structure for 3GLTE. Can be used in the BD itself if BD[NOH] is set to 0 and BD[H↩ BA] is set to NULL. Each 32 bits in the header can be accessed either as a whole or by sub-field.

Warning

Users should write reserved fields to 0 for forward compatibility

3.5.8.3.3.5.11 Field Documentation

3.5.8.3.3.5.12 uint32_t depe_3glte_header_t::nof

Number of Filler bits.

Number of filler bits needed to be added to the current data block

3.5.8.3.3.5.13 uint32_t depe_3glte_header_t::obo

Output Buffer Offset.

Offset of the first bit of the encoded block in the DEPE output buffer. Used for concatenating the current CB with previous CB from the same transport block in system memory

3.5.8.3.3.5.14 uint32_t depe_3glte_header_t::ibo

Input Buffer Offset.

Offset of the first bit of the input CB in the MAPLE input buffer pointed by [IBA] field of the DEPE BD. Used for supporting CB segmentation.Required due to the lack of filler bits in expected input

3.5.8.3.3.5.15 uint32_t depe_3glte_header_t::cbsi

Code Block Size Index.

Select one of 188 CB sizes supported in 3GLTE. Legal values 0-187

3.5.8.3.3.5.16 uint32_t depe_3glte_header_t::cm

CRC Mode.

Select the CRC polynomial. Valid only if [CD] = 0. 0 CB CRC polynomial (D24 + D23 + D6 + D5 + D + 1) 1 Transport block CRC polynomial (D24 + D23 + D18 + D17 + D14 + D11 + D10 + D7 + D6 + D5 + D4 + D3 + D + 1)

3.5.8.3.3.5.17 uint32_t depe_3glte_header_t::lh

Last Header indication.

Indicated the MAPLE that this Header is the last Header in the BD. If [NOH] $>$ 0, this bit should be set only for the last Header. If [NOH] == 0, this bit should be set for the single Header

3.5.8.3.3.5.18 uint32_t depe_3glte_header_t::cd

CRC Disable.

Indicates the DEPE whether to perform CRC calculation. If enabled, the CRC polynomial is determined using [CM].

3.5.8.3.3.5.19 uint32_t depe_3glte_header_t::noh

Number of Headers.

Number of Headers included in this job

3.5.8.3.3.5.20 uint32_t depe_3glte_header_t::rmno

Rate Matching Number of Output bits.

The required number of encoded bits after rate matching. Since the DEPE does not support repetition, the

value of this field must not be greater than the value of [RMNI]

3.5.8.3.3.5.21 uint32_t depe_3glte_header_t::rms

Rate Matching Start bit.

The starting bit in the virtual circular buffer, that is, the virtual circular buffer index of the first bit in the output stream

3.5.8.3.3.5.22 uint32_t depe_3glte_header_t::rmni

Rate Matching Number of Input bits.

Number of encoded bits in the virtual circular buffer

3.5.8.3.3.5.23 union depe_wimax_header_t

DEPE Header Structure for WiMAX (802.16e)

The DEPE Header Structure for WiMAX. Can be used in the BD itself if BD[NOH] is set to 0 and B↩ D[HBA] is set to NULL. Each 32 bits in the header can be accessed either as a whole or by sub-field.

Warning

Users should write reserved fields to 0 for forward compatibility

3.5.8.3.3.5.24 Field Documentation

3.5.8.3.3.5.25 uint32_t depe_wimax_header_t::rmno

Rate Matching Number of Output bits.

The required number of encoded bits after rate matching. Since the DEPE does not support repetition, the value of this field must not be greater than the value of [RMNI]

3.5.8.3.3.5.26 uint32_t depe_wimax_header_t::rms

Rate Matching Start bit.

The starting bit in the virtual circular buffer, that is, the virtual circular buffer index of the first bit in the output stream

3.5.8.3.3.5.27 uint32_t depe_wimax_header_t::cbsi

Code Block Size Index.

3.5.8.3.3.5.28 uint32_t depe_wimax_header_t::rd

Randomizer Disable.

Set to 1 to disable

3.5.8.3.3.5.29 uint32_t depe_wimax_header_t::lh

Last Header indication.

Indicated the MAPLE that this Header is the last Header in the BD. If [NOH] > 0, this bit should be set only for the last Header. If [NOH] == 0, this bit should be set for the single Header

3.5.8.3.3.5.30 uint32_t depe_wimax_header_t::cd

CRC Disable.

Set to 1 to disable CRC calculation.

3.5.8.3.3.5.31 uint32_t depe_wimax_header_t::noh

Number of Headers.

Number of Headers included in this job

3.5.8.3.3.5.32 union depe_wimax16m_header_t

DEPE Header Structure for WiMAX (802.16m)

The DEPE Header Structure for WiMAX. Can be used in the BD itself if BD[NOH] is set to 0 and B↩ D[HBA] is set to NULL. Each 32 bits in the header can be accessed either as a whole or by sub-field.

Warning

Users should write reserved fields to 0 for forward compatibility

3.5.8.3.3.5.33 Field Documentation

3.5.8.3.3.5.34 uint32_t depe_wimax16m_header_t::clsb

Circularly Left Shift B Sub-block.

3.5.8.3.3.5.35 uint32_t depe_wimax16m_header_t::rmno

Rate Matching Number of Output bits.

3.5.8.3.3.5.36 uint32_t depe_wimax16m_header_t::clsy

Circularly Left Shift Y Sub-block.

3.5.8.3.3.5.37 uint32_t depe_wimax16m_header_t::clsw

Circularly Left Shift W Sub-block.

3.5.8.3.3.5.38 uint32_t depe_wimax16m_header_t::cbsi

Code Block Size Index.

3.5.8.3.3.5.39 uint32_t depe_wimax16m_header_t::rd

Randomizer Disable.

Set to 1 to disable

3.5.8.3.3.5.40 uint32_t depe_wimax16m_header_t::lh

Last Header indication.

Indicated the MAPLE that this Header is the last Header in the BD. If [NOH] $>$ 0, this bit should be set only for the last Header. If [NOH] == 0, this bit should be set for the single Header

3.5.8.3.3.5.41 uint32_t depe_wimax16m_header_t::cd

CRC Disable.

Set to 1 to disable CRC calculation.

3.5.8.3.3.5.42 uint32_t depe_wimax16m_header_t::noh

Number of Headers.

Number of Headers included in this job

3.5.8.3.3.5.43 union depe_umts_header_t

DEPE Header Structure for UMTS.

The DEPE Header Structure for UMTS. Can be used in the BD itself if BD[NOH] is set to 0 and BD[HBA] is set to NULL. Each 32 bits in the header can be accessed either as a whole or by sub-field.

Warning

Users should write reserved fields to 0 for forward compatibility

3.5.8.3.3.5.44 Field Documentation

3.5.8.3.3.5.45 uint32_t depe_umts_header_t::pfem2

Parity First Eminus #2.

Error decrement variable (eminus) for second rate matching stage of the parity first stream

3.5.8.3.3.5.46 uint32_t depe_umts_header_t::cbs

Code Block Size.

The number of bits in the current CB. The size can vary from 40 to 5114. Other values are reserved

3.5.8.3.3.5.47 uint32_t depe_umts_header_t::bct

Bit Collection Type.

00 Systematic, first parity and second parity are collected in separate vectors. 01 Systematic, first parity and second parity bits are interleaved by a rectangular interleaver with 2 rows 10 Systematic, first parity and second parity bits are interleaved by a rectangular interleaver with 4 rows 11 Systematic, first parity and second parity bits are interleaved by a rectangular interleaver with 6 rows

3.5.8.3.3.5.48 uint32_t depe_umts_header_t::obo

Output Buffer Offset.

Offset of the first bit of the encoded block in the DEPE output buffer. Used for concatenating the current CB with previous CB from the same transport block in system memory

3.5.8.3.3.5.49 uint32_t depe_umts_header_t::ibo

Input Buffer Offset.

Offset of the first bit of the input CB in the MAPLE input buffer pointed by [IBA] field of the DEPE BD. Used for supporting CB segmentation.Required due to the lack of filler bits in expected input

3.5.8.3.3.5.50 uint32_t depe_umts_header_t::cbc

Continuous Bit Collection.

Indicates the DEPE that the current CB is a following CB in a current TB executed successively and in order in the DEPE in order to support full TB bit collection

3.5.8.3.3.5.51 uint32_t depe_umts_header_t::p

Number of Physical channels.

This field indicates the number of physical channel allocated for the current job. This is ignored if bit collection is not done by a rectangular interleaver (that is, if BCM=0 or BCT=0). 0 is a reserved value

3.5.8.3.3.5.52 uint32_t depe_umts_header_t::bcm

Bit Collection Mode.

0 - Systematic, first parity and second parity are interlaced. 1 - according to BCT

3.5.8.3.3.5.53 uint32_t depe_umts_header_t::rep

Repetition.

0 - RM2 implements puncturing. 1 - RM2 implements repetition

3.5.8.3.3.5.54 uint32_t depe_umts_header_t::rrm

Reset Rate Matching machine.

Indicates the DEPE whether to reset the error parameters or to resume with the current internal error parameters values assuming the current CB is the successive CB in the transport block of the previous task

3.5.8.3.3.5.55 uint32_t depe_umts_header_t::sd

Scrambling Disable.

Set to 1 to disable

3.5.8.3.3.5.56 uint32_t depe_umts_header_t::lh

Last Header indication.

Indicated the MAPLE that this Header is the last Header in the BD. If [NOH] $>$ 0, this bit should be set only for the last Header. If [NOH] == 0, this bit should be set for the single Header

3.5.8.3.3.5.57 uint32_t depe_umts_header_t::cd

CRC Disable.

Set to 1 to disable CRC calculation.

3.5.8.3.3.5.58 uint32_t depe_umts_header_t::noh

Number of Headers.

Number of Headers included in this job

3.5.8.3.3.5.59 uint32_t depe_umts_header_t::psem2

Parity Second Eminus #2.

Error decrement variable (eminus) for the second RM stage of the second parity stream

3.5.8.3.3.5.60 uint32_t depe_umts_header_t::pem1

Parity Eminus #1.

This parameter is used to calculate the error decrement variable (eminus) for the first RM stage of the parity streams. For E-DCH processing must be disabled by clearing to zero

3.5.8.3.3.5.61 uint32_t depe_umts_header_t::sei

Systematic Eini.

Initial error variable (eini) for systematic stream RM

3.5.8.3.3.5.62 uint32_t depe_umts_header_t::sem

Systematic Eminus.

Error decrement variable (eminus) for systematic stream RM

3.5.8.3.3.5.63 uint32_t depe_umts_header_t::psei2

Parity Second Eini #2.

Initial error variable (eini) for the second rate matching stage of the second parity stream

3.5.8.3.3.5.64 uint32_t depe_umts_header_t::pfei2

Parity First Eini #2.

Initial error variable (eini) for the second rate matching stage of the first parity stream

3.5.8.3.3.5.65 uint32_t depe_umts_header_t::psei1

Parity Second Eini #1.

Initial error variable (eini) for the first rate matching stage of the second parity stream. For E-DCH processing must be disabled by assigning to any value bigger than zero

3.5.8.3.3.5.66 uint32_t depe_umts_header_t::pfei1

Parity First Eini #1.

Initial error variable (eini) for the first rate matching stage of the first parity stream. For E-DCH processing must be disabled by assigning to any value bigger than zero.

3.5.8.3.3.5.67 uint32_t depe_umts_header_t::pso

Parity Second Offset.

This field is valid when [BCM] is set, and indicates the offset of the parity second bits in the output vector

3.5.8.3.3.5.68 uint32_t depe_umts_header_t::pfo

Parity First Offset.

This field is valid when [BCM] is set, and indicates the offset of the parity first bits in the output vector

3.5.8.3.3.5.69 uint32_t depe_umts_header_t::ncb

Number of Code Block.

Indicates the number of CBs in the current TB. Used for calculating the X parameter

3.5.8.3.3.5.70 uint32_t depe_umts_header_t::epfei2

Extension for Parity First Eini #2.

3 bits extension for [PFEI2]

3.5.8.3.3.5.71 uint32_t depe_umts_header_t::epsei2

Extension for Parity Second Eini #2.

2 bits extension for [PSEI2].

3.5.8.3.3.5.72 uint32_t depe_umts_header_t::epsei1

Extension for Parity Second Eini #1.

2 bits extension for [PSEI1]

3.5.8.3.3.5.73 uint32_t depe_umts_header_t::epfei1

Extension for Parity First Eini #1.

3 bits extension for [PFEI1]

3.5.8.3.3.5.74 uint32_t depe_umts_header_t::epsem2

Extension for Parity Second Eminus #2.

2 bits extension for [PSEM2]

3.5.8.3.3.5.75 uint32_t depe_umts_header_t::epem1

Extension for Parity Eminus #1.

3 bits extension for [PEM1]. For E-DCH processing must be disabled by clearing to zero

3.5.8.3.3.5.76 uint32_t depe_umts_header_t::esei

Extension for systematic Eini.

2 bits extension for [SEI]

3.5.8.3.3.5.77 uint32_t depe_umts_header_t::esem

Extension for Systematic Eminus.

2 bits extension for [SEM]

3.5.8.3.3.5.78 Enumeration Type Documentation

3.5.8.3.3.5.79 anonymous enum

DEPE Header Enumeration.

Enumerator

DEPE_HEADER_1 Access to BD[DEPE_HEADER_1] - Relevant for all Standards.
DEPE_HEADER_2 Access to BD[DEPE_HEADER_2] - Relevant for all Standards.
DEPE_HEADER_3 Access to BD[DEPE_HEADER_3] - Relevant for 3GLTE and UMTS Standards only.
DEPE_HEADER_4 Access to BD[DEPE_HEADER_4] - Relevant for 3GLTE and UMTS Standards only.
DEPE_HEADER_5 Access to BD[DEPE_HEADER_5] - Relevant for UMTS Standard only.
DEPE_HEADER_6 Access to BD[DEPE_HEADER_6] - Relevant for UMTS Standard only.
DEPE_HEADER_7 Access to BD[DEPE_HEADER_7] - Relevant for UMTS Standard only.

DEPE_HEADER_8 Access to BD[DEPE_HEADER_8] - Relevant for UMTS Standard only.
DEPE_NUM_HEADERS Number of DEPE headers in BD.

3.5.9 Maple PUSCH API

3.5.9.1 Overview

MAPLE PUSCH Initialization and Runtime API

Modules

- PUSCH Initialization
- PUSCH Runtime

3.5.9.1.1 PUSCH Initialization

3.5.9.1.1.1 Overview

PUSCH device initialization API

Data Structures

- struct pusch_open_params_t
- struct maple_pusch_ch_open_additional_interrupt
- struct maple_pusch_ch_open_term_additional_interrupt
- struct maple_pusch_ch_open_params_t

Macros

- #define PUSCH_DEV_ID_0 0
- #define PUSCH_DEV_NAME_0 "PUSCH0"
- #define PUSCH_DEV_NAME_1 "PUSCH1"

Typedefs

- typedef struct maple_pe_init_params_s maple_pusch_init_params_s
- typedef maple_pe_init_params_t maple_pusch_init_params_t
- typedef void(∗ add_int_cb)(void ∗job_handle, void ∗param)
- typedef void(∗ maple_pusch_int_cb_t)(void ∗job_handle, void ∗user)

Enumerations

- enum

Functions

- os_status maplePuschInitialize (maple_pusch_init_params_s *init_params, unsigned int num_devices, os_status(*channel_dispatch)(void *channel, void *jobs, int *num_jobs), void(*channel_reap)(void *channel, void *maple))

3.5.9.1.1.2 Data Structure Documentation

3.5.9.1.1.2.1 struct pusch_open_params_t

MAPLE PUSCH LLD Device Open paramaters.

Data Fields

- void * maple_handle
- maple_pe_bd_priority_t maple_pe_bd_priority
- maple_pe_num_bd_t maple_pe_max_num_bd_rings

3.5.9.1.1.2.2 Field Documentation

3.5.9.1.1.2.3 void* pusch_open_params_t::maple_handle

Handle returned from osCopDeviceOpen() for MAPLE controller.

3.5.9.1.1.2.4 maple_pe_bd_priority_t pusch_open_params_t::maple_pe_bd_priority

BD rings priority scheduling - only configured by MAPLE master.

3.5.9.1.1.2.5 maple_pe_num_bd_t pusch_open_params_t::maple_pe_max_num_bd_rings

The max number of BD rings for each priority - only configured by MAPLE master.

3.5.9.1.1.2.6 struct maple_pusch_ch_open_additional_interrupt

MAPLE PUSCH channel open parameters for additional interrupt.

Data Fields

- uint32_t int_enable:1
- uint32_t multiplexed_int:1
- os_hwi_dispatcher int_dispatcher
- maple_pusch_int_cb_t int_callback
- os_hwi_handle int_num
- os_hwi_priority int_priority

3.5.9.1.1.2.7 Field Documentation

3.5.9.1.1.2.8 uint32_t maple_pusch_ch_open_additional_interrupt::int_enable

Set to 1 to enable symbol interrupt for this channel.

3.5.9.1.1.2.9 uint32_t maple_pusch_ch_open_additional_interrupt::multiplexed_int

Set to 1 for the driver if this is a multiplexed interrupt line, for example if this interrupt line is used for multiple channels.

3.5.9.1.1.2.10 os_hwi_dispatcher maple_pusch_ch_open_additional_interrupt::int_dispatcher

Interrupt dispatcher to be called when interrupt occurs.

3.5.9.1.1.2.11 maple_pusch_int_cb_t maple_pusch_ch_open_additional_interrupt::int_callback

User call back function to be called on interrupt assertion.

3.5.9.1.1.2.12 os_hwi_handle maple_pusch_ch_open_additional_interrupt::int_num

Which interrupt line on this device should the interrupt handler register to.

3.5.9.1.1.2.13 os_hwi_priority maple_pusch_ch_open_additional_interrupt::int_priority

Interrupt priority.

3.5.9.1.1.2.14 struct maple_pusch_ch_open_term_additional_interrupt

MAPLE PUSCH channel open parameters for additional termination interrupts.

Data Fields

- uint32_t int_enable:1
- uint32_t multiplexed_int:1
- os_hwi_dispatcher int_dispatcher
- add_int_cb int_callback
- os_hwi_handle int_num
- os_hwi_priority int_priority

3.5.9.1.1.2.15 Field Documentation

3.5.9.1.1.2.16 uint32_t maple_pusch_ch_open_term_additional_interrupt::int_enable

Set to 1 to enable symbol interrupt for this channel.

3.5.9.1.1.2.17 uint32_t maple_pusch_ch_open_term_additional_interrupt::multiplexed_int

Set to 1 for the driver if this is a multiplexed interrupt line, for example if this interrupt line is used for multiple channels.

3.5.9.1.1.2.18 os_hwi_dispatcher maple_pusch_ch_open_term_additional_interrupt::int_↩ dispatcher

Interrupt dispatcher to be called when interrupt occurs.

3.5.9.1.1.2.19 add_int_cb maple_pusch_ch_open_term_additional_interrupt::int_callback

Call back function to be called on interrupt assertion.

3.5.9.1.1.2.20 os_hwi_handle maple_pusch_ch_open_term_additional_interrupt::int_num

Which interrupt line on this device should the interrupt handler register to.

3.5.9.1.1.2.21 os_hwi_priority maple_pusch_ch_open_term_additional_interrupt::int_priority

Interrupt priority.

3.5.9.1.1.2.22 struct maple_pusch_ch_open_params_t

MAPLE PUSCH channel open parameters type.

Data Fields

- maple_pe_ch_open_params_t pe_ch_params
- maple_pusch_ch_open_additional_interrupt ctrl_interrupt
- add_int_cb done_cb
- add_int_cb all_ctrl_done_cb
- maple_pusch_ch_open_term_additional_interrupt eq_term_interrupt
- maple_pusch_ch_open_term_additional_interrupt pu_term_interrupt

3.5.9.1.1.2.23 Field Documentation

3.5.9.1.1.2.24 maple_pe_ch_open_params_t maple_pusch_ch_open_params_t::pe_ch_params

PE channel - common parameters.

3.5.9.1.1.2.25 maple_pusch_ch_open_additional_interrupt maple_pusch_ch_open_params_t↩ ::ctrl_interrupt

Control interrupt initialization.

3.5.9.1.1.2.26 add_int_cb maple_pusch_ch_open_params_t::done_cb

Callback for PUSCH User Done.

3.5.9.1.1.2.27 add_int_cb maple_pusch_ch_open_params_t::all_ctrl_done_cb

Callback for PUSCH All Users Ctrl Done.

3.5.9.1.1.2.28 maple_pusch_ch_open_term_additional_interrupt maple_pusch_ch_open_↩ params_t::eq_term_interrupt

All EQ Termination users interrupt initialization.

3.5.9.1.1.2.29 maple_pusch_ch_open_term_additional_interrupt maple_pusch_ch_open_↩ params_t::pu_term_interrupt

All PU Termination users interrupt initialization.

3.5.9.1.1.3 Macro Definition Documentation

3.5.9.1.1.3.1 #define PUSCH_DEV_ID_0 0

PUSCH number 0 in MAPLE.

3.5.9.1.1.3.2 #define PUSCH_DEV_NAME_0 "PUSCH0"

MAPLE0 PUSCH0 device name.

3.5.9.1.1.3.3 #define PUSCH_DEV_NAME_1 "PUSCH1"

MAPLE1 PUSCH1 device name.

3.5.9.1.1.4 Typedef Documentation

3.5.9.1.1.4.1 typedef struct maple_pe_init_params_s maple_pusch_init_params_s

MAPLE PUSCH initialization parameters type for one PE.

3.5.9.1.1.4.2 typedef maple_pe_init_params_t maple_pusch_init_params_t

MAPLE PUSCH initialization parameters type for multiple PEs.

3.5.9.1.1.4.3 typedef void(∗ add_int_cb)(void ∗job_handle, void ∗param)

Interrupt Callbacks function typedef.

Pointer to BD call back function to be called on interrupt assertion

3.5.9.1.1.4.4 typedef void(∗ maple_pusch_int_cb_t)(void ∗job_handle, void ∗user)

Pointer to user call back function to be called on interrupt assertion.

3.5.9.1.1.5 Enumeration Type Documentation

3.5.9.1.1.5.1 anonymous enum

MAPLE PUSCH steering bits pointers mapping.

Use it for accessing relevant maple_pe_ch_open_params_t.steering_bits[x]

3.5.9.1.1.6 Function Documentation

3.5.9.1.1.6.1 os_status maplePuschInitialize (maple_pusch_init_params_s * *init_params,* unsigned int *num_devices,* os_status(*)(void *channel, void *jobs, int *num_jobs) *channel_dispatch,* void(*)(void *channel, void *maple) *channel_reap*)

Initializes the PUSCH driver's structures

The driver can supply default MAPLE parameters for initialization. The user can override these parameters by specifying an alternative MAPLE parameters structure. Although this function is called by all cores, only the master core performs the initialization of the MAPLE registers.

Parameters

in	*init_params*	- MAPLE Initialization parameters. if NULL, default MAPLE parameters will be used.
in	*num_devices*	- Number of PUSCH devices
in	*channel_dispatch*	- Pointer to channel dispatch function.
in	*channel_reap*	- Pointer to channel reap function.

Returns

OS_SUCCESS

Warning

This function is generally called by osArchInitialize() as part of the kernel and drivers

3.5.9.1.2 PUSCH Runtime

3.5.9.1.2.1 Overview

PUSCH Runtime API

Macros

- #define PUSCH_MAX_NUM_BD_FOR_DISPACTH 8
- #define PUSCH_NUM_OF_ROWS(x) ((x) - 1)

Functions

- INLINE uint32_t PUSCH_SEGMENT_HEADER_USER_TYPE (pusch_segment_header_user_type_t user_type_0, pusch_segment_header_user_type_t user_type_1, pusch_segment_header_user_type_t user_type_2, pusch_segment_header_user_type_t user_type_3)
- INLINE uint32_t PUSCH_SEGMENT_RB_START (uint8_t rb_start)

PUSCH Channel Control Commands

```
Used in osCopChannelCtrl()
```

- #define MAPLE_PUSCH_CMD_CTRL_POLL (0x00001000 | COP_LLD_COMMAND)
- #define PUSCH_CH_CMD_GET_EQPE_STATUS (0x00002000 | COP_LLD_COMMAND)
- #define PUSCH_CH_CMD_GET_TERM_STATUS (0x00003000 | COP_LLD_COMMAND)
- #define PUSCH_CH_CMD_SET_USER_NEIGHBOR_READY (0x00004000 | COP_LLD_CO↩ MMAND)

PUSCH_CH_CMD_GET_TERM_STATUS flags

Used to retrieve data from all_done_status after PUSCH_CH_CMD_GET_TERM_STATUS was executed

- enum {
  PUSCH_TERM_STATUS_ALL_CTRL_DONE = 0x1, PUSCH_TERM_STATUS_ALL_EQ_T↩ ERM_DONE = 0x2, PUSCH_TERM_STATUS_ALL_PU_TERM_DONE = 0x4,
  PUSCH_TERM_STATUS_PU_TERM_NUM_MASK = 0x0000FF00, PUSCH_TERM_STATU↩ S_EQ_TERM_NUM_MASK = 0x00FF0000 }

Flags for maple_pusch_job_t.flags

- #define PUSCH_BD_MANUAL_ACTIVATION 0x80000000
- #define PUSCH_BD_ALL_USERS_DONE_INT_EN 0x10000000
- #define PUSCH_BD_ALL_CTRL_INT_EN 0x08000000
- #define PUSCH_BD_ALL_EQ_TERM_INT_EN 0x04000000
- #define PUSCH_BD_ALL_PU_TERM_INT_EN 0x02000000
- #define PUSCH_BD_EXTENDED_JOB 0x01000000
- #define PUSCH_NV_BETA_OUT_EN 0x00800000
- #define PUSCH_COMP_EN 0x00400000
- #define PUSCH_JOINT_EQ_EN 0x00200000
- #define PUSCH_BD_EXCP_EN 0x00100000

Flags for maple_pusch_job_t.config0

- #define PUSCH_BD_NV_BETA_SRC 0x00100000
- #define PUSCH_BD_DECOUPLE 0x00010000

Number of Columns

- enum pusch_num_of_cols {
  PUSCH_9_COLUMNS = (8 << 12), PUSCH_10_COLUMNS = (9 << 12), PUSCH_11_COL↩ UMNS = (10 << 12),
  PUSCH_12_COLUMNS = (11 << 12) }

CQI maximal output - should be set in cqi_max

- #define OUT_CQI_RI_DECODED 0x0
- #define OUT_CQI_AMOUNT1 0x1

Modulation

- #define MODULATION_QPSK 0x0

- #define MODULATION_16QAM 0x1
- #define MODULATION_64QAM 0x2

Number of HARQ ACK/NACK bits - should be set in ACK_BITS

- enum pusch_num_ack_bits_t {
  PUSCH_1_HARQ_ACK_NACK = (0 << 26), PUSCH_2_HARQ_ACK_NACK = (1 << 26), P↩
  USCH_3_HARQ_ACK_NACK = (2 << 26),
  PUSCH_4_HARQ_ACK_NACK = (3 << 26), PUSCH_5_HARQ_ACK_NACK = (4 << 26), P↩
  USCH_6_HARQ_ACK_NACK = (5 << 26),
  PUSCH_7_HARQ_ACK_NACK = (6 << 26), PUSCH_8_HARQ_ACK_NACK = (7 << 26), P↩
  USCH_9_HARQ_ACK_NACK = (8 << 26),
  PUSCH_10_HARQ_ACK_NACK = (9 << 26), PUSCH_11_HARQ_ACK_NACK = (10 << 26),
  PUSCH_12_HARQ_ACK_NACK = (11 << 26),
  PUSCH_13_HARQ_ACK_NACK = (12 << 26), PUSCH_14_HARQ_ACK_NACK = (13 << 26), PUSCH_15_HARQ_ACK_NACK = (14 << 26),
  PUSCH_16_HARQ_ACK_NACK = (15 << 26), PUSCH_17_HARQ_ACK_NACK = (16 << 26), PUSCH_18_HARQ_ACK_NACK = (17 << 26),
  PUSCH_19_HARQ_ACK_NACK = (18 << 26), PUSCH_20_HARQ_ACK_NACK = (19 << 26)
  }

Segment Header Bits

- #define PUSCH_SEGMENT_HEADER_LV0 0x00800000
- #define PUSCH_SEGMENT_HEADER_LV1 0x00400000
- #define PUSCH_SEGMENT_HEADER_LV2 0x00200000
- #define PUSCH_SEGMENT_HEADER_LV3 0x00100000
- #define PUSCH_SEGMENT_HEADER_C_EN 0x00000100
- #define PUSCH_SEGMENT_HEADER_SN 0x80000000
- #define PUSCH_SEGMENT_HEADER_LTC_0 0x00080000
- #define PUSCH_SEGMENT_HEADER_LTC_1 0x00040000
- #define PUSCH_SEGMENT_HEADER_LTC_2 0x00020000
- #define PUSCH_SEGMENT_HEADER_LTC_3 0x00010000
- #define PUSCH_SEGMENT_HEADER_USERS_DONE_0 0x00080000
- #define PUSCH_SEGMENT_HEADER_USERS_DONE_1 0x00040000
- #define PUSCH_SEGMENT_HEADER_USERS_DONE_2 0x00020000
- #define PUSCH_SEGMENT_HEADER_USERS_DONE_3 0x00010000
- #define PUSCH_SEGMENT_HEADER_USERS_DONE_BITMASK(MASK) (MASK << 16)
- #define PUSCH_SEGMENT_HEADER_W_OFFSET(OFFSET) (((uint32_t)((uint16_t)OFFSET)) << 18)
- #define PUSCH_SEGMENT_HEADER_W_GAP(GAP) ((uint16_t)GAP << 2)
- #define PUSCH_SEGMENT_HEADER_F_OFFSET(OFFSET) (((uint32_t)((uint16_t)OFFSET)) << 18)
- #define PUSCH_SEGMENT_HEADER_F_GAP(GAP) ((uint16_t)GAP << 2)
- #define PUSCH_SEGMENT_HEADER_SRS_EN(LAYER_BITS_MASK) ((uint32_t)(LAYER↩
  _BITS_MASK << 12))
- #define PUSCH_SEGMENT_HEADER_USER_RB_START(LAYER, RBSTART) ((uint32_↩
  t)((RBSTART) << (8 * (3 - LAYER))))
- #define PUSCH_SEGMENT_HEADER_USER_RB_AMOUNT(LAYER, AMOUNT) (((uint32↩
  _t)AMOUNT) << (8 * (3 - LAYER)))
- #define PUSCH_SEGMENT_HEADER_USER_ID_IN_LAYER(LAYER, USER) (((uint32_t)U↩

SER) << (8 * (3 - LAYER)))

User Header User type, Should be set in maple_pusch_sh_t.config[8] for groups #0 to #3 and maple_pusch_sh_t.config[9] for groups #4 to #7 .

- #define PUSCH_SEGMENT_HEADER_GROUP0_MASK(MASK) (((uint32_t)(MASK)) << 24)
- #define PUSCH_SEGMENT_HEADER_GROUP1_MASK(MASK) (((uint32_t)(MASK)) << 16)
- #define PUSCH_SEGMENT_HEADER_GROUP2_MASK(MASK) (((uint32_t)(MASK)) << 8)
- #define PUSCH_SEGMENT_HEADER_GROUP3_MASK(MASK) ((uint32_t)(MASK))
- #define PUSCH_SEGMENT_HEADER_GROUP4_MASK(MASK) PUSCH_SEGMENT_HEADER_GROUP0_MASK(MASK)
- #define PUSCH_SEGMENT_HEADER_GROUP5_MASK(MASK) PUSCH_SEGMENT_HEADER_GROUP1_MASK(MASK)
- #define PUSCH_SEGMENT_HEADER_GROUP6_MASK(MASK) PUSCH_SEGMENT_HEADER_GROUP2_MASK(MASK)
- #define PUSCH_SEGMENT_HEADER_GROUP7_MASK(MASK) PUSCH_SEGMENT_HEADER_GROUP3_MASK(MASK)

User Header User type, Should be set in using PUSCH_SEGMENT_HEADER_USER_TYPE().

- enum pusch_segment_header_user_type_t { PUSCH_SEGMENT_HEADER_USER_TYPE_NON_COMP = 0x00000000, PUSCH_SEGMENT_HEADER_USER_TYPE_EQPE_TERMINATION = 0x00000001, PUSCH_SEGMENT_HEADER_USER_TYPE_PUPE_TERMINATION = 0x00000002, PUSCH_SEGMENT_HEADER_USER_TYPE_LLR_COMBINED = 0x00000003, PUSCH_SEGMENT_HEADER_USER_TYPE_EQPE_COMBINED = 0x00000004 }

Segment Header number of layers

- enum **pusch_segment_header_lx_t**

Segment Header number of columns

- enum pusch_segment_header_cols_t { PUSCH_SEGMENT_HEADER_COLS_9 = 0x00008000, PUSCH_SEGMENT_HEADER_COLS_10 = 0x00009000, PUSCH_SEGMENT_HEADER_COLS_11 = 0x0000a000, PUSCH_SEGMENT_HEADER_COLS_12 = 0x0000b000 }

Segment Header number of rows

- enum pusch_segment_header_rows_t { PUSCH_SEGMENT_HEADER_ROWS_12 = 0x0000000b, PUSCH_SEGMENT_HEADER_ROWS_24 = 0x00000017, PUSCH_SEGMENT_HEADER_ROWS_36 = 0x00000023, PUSCH_SEGMENT_HEADER_ROWS_48 = 0x0000002f, PUSCH_SEGMENT_HEADER_ROWS_60 = 0x0000003b, PUSCH_SEGMENT_HEADER_ROWS_72 = 0x00000047, PUSCH_SEGMENT_HEADER_ROWS_96 = 0x0000005f, PUSCH_SEGMENT_HEADER_ROWS_108 = 0x0000006d, PUSCH_SEGMENT_HEADER_ROWS_120 = 0x00000077, PUSCH_SEGMENT_HEADER_ROWS_144 = 0x0000008f, PUSCH_SEGMENT_HEADER_ROWS_180 = 0x000000b3, PUSCH_SEGMENT_HEADER_ROWS_192 = 0x000000bf, PUSCH_SEGMENT_HEADER_ROWS_216 = 0x000000d7, PUSCH_SEGMENT_HEADER_R

OWS_240 = 0x000000ef, PUSCH_SEGMENT_HEADER_ROWS_288 = 0x0000011f, PUSCH_SEGMENT_HEADER_ROWS_300 = 0x0000012d, PUSCH_SEGMENT_HEADER_ROWS_324 = 0x00000143, PUSCH_SEGMENT_HEADER_ROWS_360 = 0x00000167, PUSCH_SEGMENT_HEADER_ROWS_384 = 0x0000017f, PUSCH_SEGMENT_HEADER_ROWS_432 = 0x000001af, PUSCH_SEGMENT_HEADER_ROWS_480 = 0x000001bf, PUSCH_SEGMENT_HEADER_ROWS_540 = 0x0000021d, PUSCH_SEGMENT_HEADER_ROWS_576 = 0x0000023f, PUSCH_SEGMENT_HEADER_ROWS_600 = 0x00000257, PUSCH_SEGMENT_HEADER_ROWS_648 = 0x00000287, PUSCH_SEGMENT_HEADER_ROWS_720 = 0x000002cf, PUSCH_SEGMENT_HEADER_ROWS_768 = 0x000002ff, PUSCH_SEGMENT_HEADER_ROWS_900 = 0x00000383, PUSCH_SEGMENT_HEADER_ROWS_960 = 0x000003bf, PUSCH_SEGMENT_HEADER_ROWS_972 = 0x000003cb, PUSCH_SEGMENT_HEADER_ROWS_1080 = 0x00000437, PUSCH_SEGMENT_HEADER_ROWS_1152 = 0x0000047f, PUSCH_SEGMENT_HEADER_ROWS_1200 = 0x000004af }

Segment Header Shared columns for W Matrix

- enum pusch_segment_header_weights_shared_columns_t { PUSCH_SEGMENT_HEADER_W_COL_ALL = 0x0, PUSCH_SEGMENT_HEADER_W_COL_1 = 0x01000000, PUSCH_SEGMENT_HEADER_W_COL_2 = 0x02000000, PUSCH_SEGMENT_HEADER_W_COL_3 = 0x03000000, PUSCH_SEGMENT_HEADER_W_COL_4 = 0x04000000, PUSCH_SEGMENT_HEADER_W_COL_5 = 0x05000000, PUSCH_SEGMENT_HEADER_W_COL_6 = 0x06000000 }

Segment Header Interpolation Weight per RX - Valid for internal interpolation only (INTRP=0).

- enum pusch_segment_header_w_per_rx_t { PUSCH_SEGMENT_HEADER_COMMON_WEIGHT_ALL_RX = 0x00000000, PUSCH_SEGMENT_HEADER_DISTINCT_WEIGHT_PER_RX = 0x40000000 }

Segment Header weight shared rows

- enum pusch_segment_header_w_row_t { PUSCH_SEGMENT_HEADER_WROW_0 = 0x0, PUSCH_SEGMENT_HEADER_WROW_1 = 0x00010000, PUSCH_SEGMENT_HEADER_WROW_2 = 0x00020000, PUSCH_SEGMENT_HEADER_WROW_3 = 0x00030000, PUSCH_SEGMENT_HEADER_WROW_4 = 0x00040000, PUSCH_SEGMENT_HEADER_WROW_6 = 0x00060000, PUSCH_SEGMENT_HEADER_WROW_12 = 0x000c0000 }

Segment Header Shared columns for S Matrix

- enum pusch_segment_header_shared_columns_t { PUSCH_SEGMENT_HEADER_S_COL_ALL = 0x0, PUSCH_SEGMENT_HEADER_S_COL_1 = 0x00000100, PUSCH_SEGMENT_HEADER_S_COL_2 = 0x00000200, PUSCH_SEGMENT_HEADER_S_COL_3 = 0x00000300, PUSCH_SEGMENT_HEADER_S_COL_4 = 0x00000400, PUSCH_SEGMENT_HEADER_S_COL_5 = 0x00000500, PUSCH_SEGMENT_HEADER_S_COL_6 = 0x00000600 }
- enum pusch_segment_header_shared_rows_t { PUSCH_SEGMENT_HEADER_S_ROW_ALL = 0x0, PUSCH_SEGMENT_HEADER_S_RO

W_1 = 0x0000001, PUSCH_SEGMENT_HEADER_S_ROW_2 = 0x0000002, PUSCH_SEGMENT_HEADER_S_ROW_3 = 0x0000003, PUSCH_SEGMENT_HEADER_S_↩ ROW_4 = 0x0000004, PUSCH_SEGMENT_HEADER_S_ROW_6 = 0x0000006, PUSCH_SEGMENT_HEADER_S_ROW_12 = 0x000000c }

Segment Header S Matrix scale type

- enum pusch_segment_header_s_scl_type_t { PUSCH_SEGMENT_HEADER_S_SCL_TYPE_SI↩ NGLE = 0x0, PUSCH_SEGMENT_HEADER_S_SCL_TYPE_DIFFERENT = 0x00002000 }

Segment Header Interpolation mode

- enum pusch_segment_header_intrp_type_t { PUSCH_SEGMENT_HEADER_INTRP_II2 = 0x0, PUSCH_SEGMENT_HEADER_INTRP_EXTERNAL = 0x20000000, PUSCH_SEGMENT_HE↩ ADER_INTRP_NONE = 0x30000000 }

User Header Bits

- #define PUSCH_USER_HEADER_DONE_INT_EN 0x80000000
- #define PUSCH_USER_HEADER_CTRL_INT_EN 0x40000000
- #define PUSCH_USER_HEADER_DISCARD_CW1 0x10000000
- #define PUSCH_USER_HEADER_DISCARD_CW0 0x08000000
- #define PUSCH_USER_HEADER_FH_ENABLE 0x04000000
- #define PUSCH_USER_NV_BETA_OUT_EN 0x00100000
- #define PUSCH_USER_HEADER_LAYER_0 0x00010000
- #define PUSCH_USER_HEADER_LAYER_1 0x00020000
- #define PUSCH_USER_HEADER_LAYER_2 0x00040000
- #define PUSCH_USER_HEADER_LAYER_3 0x00080000
- #define PUSCH_USER_HEADER_SRS_EN 0x80000000
- #define PUSCH_USER_HEADER_CARRIER_0 0x00100000
- #define PUSCH_USER_HEADER_CARRIER_1 0x00200000
- #define PUSCH_USER_HEADER_CARRIER_2 0x00400000
- #define PUSCH_USER_HEADER_CARRIER_3 0x00800000
- #define PUSCH_USER_HEADER_CARRIER_4 0x01000000
- #define PUSCH_USER_HEADER_HOE_CW_EN 0x80000000
- #define PUSCH_USER_HEADER_LLMAP_CW_HYBRID 0x08000000
- #define PUSCH_USER_HEADER_HAOE_CW_EN 0x00020000
- #define PUSCH_USER_HEADER_FTH_CW_EN 0x00008000
- #define PUSCH_USER_HEADER_CRC_A_STP_CW_EN 0x00002000
- #define PUSCH_USER_HEADER_W1E_CW 0x04000000
- #define PUSCH_USER_HEADER_W2E_CW 0x02000000
- #define PUSCH_USER_HEADER_CQI_MAX_OUT_EN 0x80000000

3.5.9.1.2.2 Macro Definition Documentation

3.5.9.1.2.2.1 #define MAPLE_PUSCH_CMD_CTRL_POLL (0x00001000 | COP_LLD_COMMAND)

Polling the channel for finished jobs.

3.5.9.1.2.2.2 #define PUSCH_CH_CMD_GET_EQPE_STATUS (0x00002000 | COP_LLD_COMMAND)

This command expects to receive as a parameter a PUSCH job, and will return the EQPE status through the job's segment_in_done_ptr field.

3.5.9.1.2.2.3 #define PUSCH_CH_CMD_GET_TERM_STATUS (0x00003000 | COP_LLD_COMMAND)

This command expects to receive as a parameter a PUSCH job, and will return the done status through the job's term_status field.

3.5.9.1.2.2.4 #define PUSCH_CH_CMD_SET_USER_NEIGHBOR_READY (0x00004000 | COP_LLD_COMMAND)

This command expects to receive as a parameter a PUSCH job, and will set the neighbor ready field.

3.5.9.1.2.2.5 #define PUSCH_MAX_NUM_BD_FOR_DISPACTH 8

Maximum number of BDs that are allowed to dispatch all at once.

3.5.9.1.2.2.6 #define PUSCH_BD_MANUAL_ACTIVATION 0x80000000

Prevent the driver from dispatching the job, MAPLE will wait until the user calls a Channel Ctrl Command to activate the job.

3.5.9.1.2.2.7 #define PUSCH_BD_ALL_USERS_DONE_INT_EN 0x10000000

if set PUSCH will assert User Done Interrupt when all users in the BD are done.

3.5.9.1.2.2.8 #define PUSCH_BD_ALL_CTRL_INT_EN 0x08000000

If set PUSCH will assert Ctrl Interrupt when control of all users in the BD is done.

3.5.9.1.2.2.9 #define PUSCH_BD_ALL_EQ_TERM_INT_EN 0x04000000

If if set PUSCH will assert ALL_EQ_TERM_DONE_INT when all EQ termination users in the BD are done.

3.5.9.1.2.2.10 #define PUSCH_BD_ALL_PU_TERM_INT_EN 0x02000000

If set PUSCH will assert ALL_PU_TERM_DONE_INT when all PU termination users in the BD are done.

3.5.9.1.2.2.11 #define PUSCH_BD_EXTENDED_JOB 0x01000000

Buffer Descriptor Extension.

Indicates if either Joint Equalization (i.e. JEQ_EN == 1) or Post Equalization/LLR combining or termination of users after EQPE or PUPE stages (COMP_EN == 1) enabled hence BD extension is applicable.

3.5.9.1.2.2.12 #define PUSCH_NV_BETA_OUT_EN 0x00800000

NV BETA out enable.

When this bit and internal NV Beta and EQPE debug are all enabled maple will write the NV Beta result for the user after the equalization data and scale in the EQPE debug result buffer. (in the same way as for EQPE termination users).

3.5.9.1.2.2.13 #define PUSCH_COMP_EN 0x00400000

PUSCH EQPE COMP enable bit.

3.5.9.1.2.2.14 #define PUSCH_JOINT_EQ_EN 0x00200000

PUSCH Joint Equalization enable bit.

3.5.9.1.2.2.15 #define PUSCH_BD_EXCP_EN 0x00100000

Extended CP.

Indicate if this sub-frame is with normal Cyclic Prefix or extended CP. Used by the MAPLE-B3 to identify the number of columns of the users.

3.5.9.1.2.2.16 #define PUSCH_BD_NV_BETA_SRC 0x00100000

Generated NV & BETA samples internally during EQPE2 processing, else NV & BETA samples calculated by the host.

The MAPLE-B3 reads the samples from the address specified by NV_BETA_POINTER in the user header

3.5.9.1.2.2.17 #define PUSCH_BD_DECOUPLE 0x00010000

Enable Decouple mode.

3.5.9.1.2.2.18 #define PUSCH_NUM_OF_ROWS(*x*) ((x) - 1)

Number of Rows Indicates the number of rows in the resource element grid.

configure using macro or set number of rows - 1, for example for 12 rows should be 11.

3.5.9.1.2.2.19 #define OUT_CQI_RI_DECODED 0x0

Output CQI according to the decoded RI value (CQI_AMOUNT0_x or CQI_AMOUNT1_x).

3.5.9.1.2.2.20 #define OUT_CQI_AMOUNT1 0x1

Always output the CQI according to CQI_AMOUNT1_x.

3.5.9.1.2.2.21 #define MODULATION_QPSK 0x0

QPSK Modulation.

3.5.9.1.2.2.22 #define MODULATION_16QAM 0x1

16QAM Modulation

3.5.9.1.2.2.23 #define MODULATION_64QAM 0x2

64QAM Modulation

3.5.9.1.2.2.24 #define PUSCH_SEGMENT_HEADER_LV0 0x00800000

Layer #0 Valid.

Indicates that layer #0 is valid and contains data in the current segment

3.5.9.1.2.2.25 #define PUSCH_SEGMENT_HEADER_LV1 0x00400000

Layer #1 Valid.

Indicates that layer #1 is valid and contains data in the current segment

3.5.9.1.2.2.26 #define PUSCH_SEGMENT_HEADER_LV2 0x00200000

Layer #2 Valid.

Indicates that layer #2 is valid and contains data in the current segment

3.5.9.1.2.2.27 #define PUSCH_SEGMENT_HEADER_LV3 0x00100000

Layer #3 Valid.

Indicates that layer #3 is valid and contains data in the current segment

3.5.9.1.2.2.28 #define PUSCH_SEGMENT_HEADER_C_EN 0x00000100

C Matrix Enable.

Indicates whether external C matrix is to be used for the current segment or assume internally that C=I (Identity matrix)

3.5.9.1.2.2.29 #define PUSCH_SEGMENT_HEADER_SN 0x80000000

Slot Number.

If set, signals that this segment belongs to the second Slot of the sub-frame, else to first slot on the sub-frame.

3.5.9.1.2.2.30 #define PUSCH_SEGMENT_HEADER_LTC_0 0x00080000

cancel layer 0

3.5.9.1.2.2.31 #define PUSCH_SEGMENT_HEADER_LTC_1 0x00040000

cancel layer 1

3.5.9.1.2.2.32 #define PUSCH_SEGMENT_HEADER_LTC_2 0x00020000

cancel layer 2

3.5.9.1.2.2.33 #define PUSCH_SEGMENT_HEADER_LTC_3 0x00010000

cancel layer 3

3.5.9.1.2.2.34 #define PUSCH_SEGMENT_HEADER_USERS_DONE_0 0x00080000

The user allocated to layer #0 with USER_ID_L0: it is its last processed segment in the sub-frame.

3.5.9.1.2.2.35 #define PUSCH_SEGMENT_HEADER_USERS_DONE_1 0x00040000

The user allocated to layer #1 with USER_ID_L1: it is its last processed segment in the sub-frame.

3.5.9.1.2.2.36 #define PUSCH_SEGMENT_HEADER_USERS_DONE_2 0x00020000

The user allocated to layer #2 with USER_ID_L2: it is its last processed segment in the sub-frame.

3.5.9.1.2.2.37 #define PUSCH_SEGMENT_HEADER_USERS_DONE_3 0x00010000

The user allocated to layer #3 with USER_ID_L3: it is its last processed segment in the sub-frame.

3.5.9.1.2.2.38 #define PUSCH_SEGMENT_HEADER_USERS_DONE_BITMASK(*MASK*) (MASK << 16)

Set USER_ID_Lx as a mask.

3.5.9.1.2.2.39 #define PUSCH_SEGMENT_HEADER_W_OFFSET(*OFFSET*) (((uint32_t)((uint16_t)OFFSET)) << 18)

Valid for internal interpolation only (INTRP=0).

Indicates the offset from the W pointer base address where the W inputs for the current job are to be fetched by MAPLE-B3.

3.5.9.1.2.2.40 #define PUSCH_SEGMENT_HEADER_W_GAP(*GAP*) ((uint16_t)GAP << 2)

Valid for internal interpolation only (INTRP=0).

Indicates the gap between W vectors to be fetched by MAPLE-B3.

3.5.9.1.2.2.41 #define PUSCH_SEGMENT_HEADER_F_OFFSET(*OFFSET*) (((uint32_t)((uint16_t)OFFSET)) << 18)

Valid only if one or more of the LTCbits are set.

Indicates the offset from the F pointer base address (of the relevant reference symbol)

3.5.9.1.2.2.42 #define PUSCH_SEGMENT_HEADER_F_GAP(*GAP*) ((uint16_t)GAP << 2)

Valid only if one or more of the LTCbits are set.

Indicates the gap between F vectors to be fetched by MAPLE-B3. Used by MAPLE-B3 in order to calculate the F inputs location.

3.5.9.1.2.2.43 #define PUSCH_SEGMENT_HEADER_SRS_EN(*LAYER_BITS_MASK*) ((uint32_t)(LAYER_BITS_MASK << 12))

The user of layer *doesnt transmit on the last column, due to SRS transmission (i = 0..3).*

To be set in segment_params_header.config[3]

3.5.9.1.2.2.44 #define PUSCH_SEGMENT_HEADER_USER_RB_START(*LAYER, RBSTART*) ((uint32_t)((RBSTART) << (8 * (3 - LAYER))))

The layer's User Resource Block Start.

The Layer allocated Users offset from the first row of the entire Sub-Frames RE-grid in RB resolution. RBSTART Supported offsets 0..107.

3.5.9.1.2.2.45 #define PUSCH_SEGMENT_HEADER_USER_RB_AMOUNT(*LAYER, AMOUNT*) (((uint32_t)AMOUNT) << (8 * (3 - LAYER)))

The Layer's User Resource Block Amount.

The entire amount of Resource Blocks in the layer, which are (consecutively) allocated, in the current cluster, for the User which occupies the layer. If amount == 0 than the layer is not valid If amount == 1..108, than the layer have 1..108 resource blocks allocation.

3.5.9.1.2.2.46 #define PUSCH_SEGMENT_HEADER_USER_ID_IN_LAYER(*LAYER, USER*) (((uint32_t)USER) << (8 * (3 - LAYER)))

Indicates the User ID of the user located in layer #x of this segment.

3.5.9.1.2.2.47 #define PUSCH_SEGMENT_HEADER_GROUP0_MASK(*MASK*) (((uint32_t)(MASK)) << 24)

1 byte mask indicating which antenna is active in group #0.

3.5.9.1.2.2.48 #define PUSCH_SEGMENT_HEADER_GROUP1_MASK(*MASK*) (((uint32_t)(MASK)) << 16)

1 byte mask indicating which antenna is active in group #1.

3.5.9.1.2.2.49 #define PUSCH_SEGMENT_HEADER_GROUP2_MASK(*MASK*) (((uint32_t)(MASK)) << 8)

1 byte mask indicating which antenna is active in group #2.

3.5.9.1.2.2.50 #define PUSCH_SEGMENT_HEADER_GROUP3_MASK(*MASK*) ((uint32_t)(MASK))

1 byte mask indicating which antenna is active in group #3.

3.5.9.1.2.2.51 #define PUSCH_SEGMENT_HEADER_GROUP4_MASK(*MASK*) PUSCH_SEGMENT_HEADER_GROUP0_MASK(MASK)

1 byte mask indicating which antenna is active in group #4.

3.5.9.1.2.2.52 #define PUSCH_SEGMENT_HEADER_GROUP5_MASK(*MASK*) PUSCH_SEGMENT_HEADER_GROUP1_MASK(MASK)

1 byte mask indicating which antenna is active in group #5.

3.5.9.1.2.2.53 #define PUSCH_SEGMENT_HEADER_GROUP6_MASK(*MASK*) PUSCH_SEGMENT_HEADER_GROUP2_MASK(MASK)

1 byte mask indicating which antenna is active in group #6.

3.5.9.1.2.2.54 #define PUSCH_SEGMENT_HEADER_GROUP7_MASK(*MASK*) PUSCH_SEGMENT_HEADER_GROUP3_MASK(MASK)

1 byte mask indicating which antenna is active in group #7.

3.5.9.1.2.2.55 #define PUSCH_USER_HEADER_DONE_INT_EN 0x80000000

DONE interrupt enable.

3.5.9.1.2.2.56 #define PUSCH_USER_HEADER_CTRL_INT_EN 0x40000000

CTRL interrupt enable.

3.5.9.1.2.2.57 #define PUSCH_USER_HEADER_DISCARD_CW1 0x10000000

discard CW 1

3.5.9.1.2.2.58 #define PUSCH_USER_HEADER_DISCARD_CW0 0x08000000

discard CW 0

3.5.9.1.2.2.59 #define PUSCH_USER_HEADER_FH_ENABLE 0x04000000

frequency hopping enable

3.5.9.1.2.2.60 #define PUSCH_USER_NV_BETA_OUT_EN 0x00100000

If enabled, NV & BETA source samples are generated internally during EQPE2 processing regardless of what is configured in PUSCH BD[PUSCH_NV_BETA_OUT_EN].

3.5.9.1.2.2.61 #define PUSCH_USER_HEADER_LAYER_0 0x00010000

user layer 0

3.5.9.1.2.2.62 #define PUSCH_USER_HEADER_LAYER_1 0x00020000

user layer 1

3.5.9.1.2.2.63 #define PUSCH_USER_HEADER_LAYER_2 0x00040000

user layer 2

3.5.9.1.2.2.64 #define PUSCH_USER_HEADER_LAYER_3 0x00080000

user layer 3

3.5.9.1.2.2.65 #define PUSCH_USER_HEADER_SRS_EN 0x80000000

SRS enable.

3.5.9.1.2.2.66 #define PUSCH_USER_HEADER_CARRIER_0 0x00100000

carrier 0 is valid

3.5.9.1.2.2.67 #define PUSCH_USER_HEADER_CARRIER_1 0x00200000

carrier 1 is valid

3.5.9.1.2.2.68 #define PUSCH_USER_HEADER_CARRIER_2 0x00400000

carrier 2 is valid

3.5.9.1.2.2.69 #define PUSCH_USER_HEADER_CARRIER_3 0x00800000

carrier 3 is valid

3.5.9.1.2.2.70 #define PUSCH_USER_HEADER_CARRIER_4 0x01000000

carrier 4 is valid

3.5.9.1.2.2.71 #define PUSCH_USER_HEADER_HOE_CW_EN 0x80000000

hard output enable for CW

3.5.9.1.2.2.72 #define PUSCH_USER_HEADER_LLMAP_CW_HYBRID 0x08000000

hybrid linear log map enable

3.5.9.1.2.2.73 #define PUSCH_USER_HEADER_HAOE_CW_EN 0x00020000

HARQ accumulator output enable for CW.

3.5.9.1.2.2.74 #define PUSCH_USER_HEADER_FTH_CW_EN 0x00008000

first HARQ buffer

3.5.9.1.2.2.75 #define PUSCH_USER_HEADER_CRC_A_STP_CW_EN 0x00002000

CRC stop criteria for CW enabled.

3.5.9.1.2.2.76 #define PUSCH_USER_HEADER_W1E_CW 0x04000000

weight 1 enable for CW

3.5.9.1.2.2.77 #define PUSCH_USER_HEADER_W2E_CW 0x02000000

weight 2 enable for CW

3.5.9.1.2.2.78 #define PUSCH_USER_HEADER_CQI_MAX_OUT_EN 0x80000000

Enable CQI max out.

3.5.9.1.2.3 Enumeration Type Documentation

3.5.9.1.2.3.1 anonymous enum

Enumerator

PUSCH_TERM_STATUS_ALL_CTRL_DONE Use to extract the ALL_CTRL_DONE bit from pusch_job.term_status.

PUSCH_TERM_STATUS_ALL_EQ_TERM_DONE Use to extract the ALL_EQ_TERM_DONE bit from pusch_job.term_status.

PUSCH_TERM_STATUS_ALL_PU_TERM_DONE Use to extract the ALL_PU_TERM_DONE bit from pusch_job.term_status.

PUSCH_TERM_STATUS_PU_TERM_NUM_MASK Use to extract the PU_TERM_NUM bit from pusch_job.term_status.

PUSCH_TERM_STATUS_EQ_TERM_NUM_MASK Use to extract the EQ_TERM_NUM bit from pusch_job.term_status.

3.5.9.1.2.3.2 enum pusch_num_of_cols

Enumerator

PUSCH_9_COLUMNS 9 Columns

PUSCH_10_COLUMNS 10 Columns
PUSCH_11_COLUMNS 11 Columns
PUSCH_12_COLUMNS 12 Columns

3.5.9.1.2.3.3 enum pusch_num_ack_bits_t

Enumerator

PUSCH_1_HARQ_ACK_NACK 1 HARQ ACK/NACK Bits
PUSCH_2_HARQ_ACK_NACK 2 HARQ ACK/NACK Bits
PUSCH_3_HARQ_ACK_NACK 3 HARQ ACK/NACK Bits
PUSCH_4_HARQ_ACK_NACK 4 HARQ ACK/NACK Bits
PUSCH_5_HARQ_ACK_NACK 5 HARQ ACK/NACK Bits
PUSCH_6_HARQ_ACK_NACK 6 HARQ ACK/NACK Bits
PUSCH_7_HARQ_ACK_NACK 7 HARQ ACK/NACK Bits
PUSCH_8_HARQ_ACK_NACK 8 HARQ ACK/NACK Bits
PUSCH_9_HARQ_ACK_NACK 9 HARQ ACK/NACK Bits
PUSCH_10_HARQ_ACK_NACK 10 HARQ ACK/NACK Bits
PUSCH_11_HARQ_ACK_NACK 11 HARQ ACK/NACK Bits
PUSCH_12_HARQ_ACK_NACK 12 HARQ ACK/NACK Bits
PUSCH_13_HARQ_ACK_NACK 13 HARQ ACK/NACK Bits
PUSCH_14_HARQ_ACK_NACK 14 HARQ ACK/NACK Bits
PUSCH_15_HARQ_ACK_NACK 15 HARQ ACK/NACK Bits
PUSCH_16_HARQ_ACK_NACK 15 HARQ ACK/NACK Bits
PUSCH_17_HARQ_ACK_NACK 16 HARQ ACK/NACK Bits
PUSCH_18_HARQ_ACK_NACK 17 HARQ ACK/NACK Bits
PUSCH_19_HARQ_ACK_NACK 18 HARQ ACK/NACK Bits
PUSCH_20_HARQ_ACK_NACK 19 HARQ ACK/NACK Bits

3.5.9.1.2.3.4 enum pusch_segment_header_user_type_t

Enumerator

PUSCH_SEGMENT_HEADER_USER_TYPE_NON_COMP User is non-comp (Regular user).
PUSCH_SEGMENT_HEADER_USER_TYPE_EQPE_TERMINATION User process is terminated after EQPE2 stage.
PUSCH_SEGMENT_HEADER_USER_TYPE_PUPE_TERMINATION User process is terminated after PUPE2 stage.
PUSCH_SEGMENT_HEADER_USER_TYPE_LLR_COMBINED User combined after PUPE2 stage.
PUSCH_SEGMENT_HEADER_USER_TYPE_EQPE_COMBINED User combined after EQP↩ E2 stage.

3.5.9.1.2.3.5 enum pusch_segment_header_cols_t

Enumerator

PUSCH_SEGMENT_HEADER_COLS_9 9 columns
PUSCH_SEGMENT_HEADER_COLS_10 10 columns
PUSCH_SEGMENT_HEADER_COLS_11 11 columns
PUSCH_SEGMENT_HEADER_COLS_12 12 columns

3.5.9.1.2.3.6 enum pusch_segment_header_rows_t

Enumerator

PUSCH_SEGMENT_HEADER_ROWS_12 12 rows
PUSCH_SEGMENT_HEADER_ROWS_24 24 rows
PUSCH_SEGMENT_HEADER_ROWS_36 36 rows
PUSCH_SEGMENT_HEADER_ROWS_48 48 rows
PUSCH_SEGMENT_HEADER_ROWS_60 60 rows
PUSCH_SEGMENT_HEADER_ROWS_72 72 rows
PUSCH_SEGMENT_HEADER_ROWS_96 96 rows
PUSCH_SEGMENT_HEADER_ROWS_108 24 rows
PUSCH_SEGMENT_HEADER_ROWS_120 120 rows
PUSCH_SEGMENT_HEADER_ROWS_144 144 rows
PUSCH_SEGMENT_HEADER_ROWS_180 180 rows
PUSCH_SEGMENT_HEADER_ROWS_192 192 rows
PUSCH_SEGMENT_HEADER_ROWS_216 216 rows
PUSCH_SEGMENT_HEADER_ROWS_240 240 rows
PUSCH_SEGMENT_HEADER_ROWS_288 288 rows
PUSCH_SEGMENT_HEADER_ROWS_300 300 rows
PUSCH_SEGMENT_HEADER_ROWS_324 324 rows
PUSCH_SEGMENT_HEADER_ROWS_360 360 rows
PUSCH_SEGMENT_HEADER_ROWS_384 384 rows
PUSCH_SEGMENT_HEADER_ROWS_432 432 rows
PUSCH_SEGMENT_HEADER_ROWS_480 480 rows
PUSCH_SEGMENT_HEADER_ROWS_540 540 rows
PUSCH_SEGMENT_HEADER_ROWS_576 576 rows
PUSCH_SEGMENT_HEADER_ROWS_600 600 rows
PUSCH_SEGMENT_HEADER_ROWS_648 648 rows
PUSCH_SEGMENT_HEADER_ROWS_720 720 rows
PUSCH_SEGMENT_HEADER_ROWS_768 768 rows
PUSCH_SEGMENT_HEADER_ROWS_900 900 rows
PUSCH_SEGMENT_HEADER_ROWS_960 960 rows
PUSCH_SEGMENT_HEADER_ROWS_972 972 rows
PUSCH_SEGMENT_HEADER_ROWS_1080 1080 rows
PUSCH_SEGMENT_HEADER_ROWS_1152 1152 rows
PUSCH_SEGMENT_HEADER_ROWS_1200 1200 rows

3.5.9.1.2.3.7 enum pusch_segment_header_weights_shared_columns_t

Enumerator

PUSCH_SEGMENT_HEADER_W_COL_ALL All the columns in the resource-elements grid of the current job share the same interpolation weights.
PUSCH_SEGMENT_HEADER_W_COL_1 switch the W weights every single column.
PUSCH_SEGMENT_HEADER_W_COL_2 switch the W weights every 2 columns.
PUSCH_SEGMENT_HEADER_W_COL_3 switch the W weights every 3 columns.
PUSCH_SEGMENT_HEADER_W_COL_4 switch the W weights every 4 columns.
PUSCH_SEGMENT_HEADER_W_COL_5 switch the W weights every 5 columns.
PUSCH_SEGMENT_HEADER_W_COL_6 switch the W weights every 6 columns.

3.5.9.1.2.3.8 enum pusch_segment_header_w_per_rx_t

Enumerator

PUSCH_SEGMENT_HEADER_COMMON_WEIGHT_ALL_RX Common interpolation weight for all received antennas.
PUSCH_SEGMENT_HEADER_DISTINCT_WEIGHT_PER_RX Distinct interpolation weight for each received antennas.

3.5.9.1.2.3.9 enum pusch_segment_header_w_row_t

Enumerator

PUSCH_SEGMENT_HEADER_WROW_0 same interpolation weight for all rows
PUSCH_SEGMENT_HEADER_WROW_1 change the W weights every row
PUSCH_SEGMENT_HEADER_WROW_2 change the W weights every 2 rows
PUSCH_SEGMENT_HEADER_WROW_3 change the W weights every 3 rows
PUSCH_SEGMENT_HEADER_WROW_4 change the W weights every 4 rows
PUSCH_SEGMENT_HEADER_WROW_6 change the W weights every 6 rows
PUSCH_SEGMENT_HEADER_WROW_12 change the W weights every 12 rows

3.5.9.1.2.3.10 enum pusch_segment_header_shared_columns_t

Enumerator

PUSCH_SEGMENT_HEADER_S_COL_ALL all columns share the same S-Matrix
PUSCH_SEGMENT_HEADER_S_COL_1 switch the S matrix every single column
PUSCH_SEGMENT_HEADER_S_COL_2 switch the S matrix every 2 columns
PUSCH_SEGMENT_HEADER_S_COL_3 switch the S matrix every 3 columns
PUSCH_SEGMENT_HEADER_S_COL_4 switch the S matrix every 4 columns
PUSCH_SEGMENT_HEADER_S_COL_5 switch the S matrix every 5 columns
PUSCH_SEGMENT_HEADER_S_COL_6 switch the S matrix every 6 columns

3.5.9.1.2.3.11 enum pusch_segment_header_shared_rows_t

Enumerator

PUSCH_SEGMENT_HEADER_S_ROW_ALL all rows share the same S-Matrix
PUSCH_SEGMENT_HEADER_S_ROW_1 switch the S matrix every single row
PUSCH_SEGMENT_HEADER_S_ROW_2 switch the S matrix every 2 rows
PUSCH_SEGMENT_HEADER_S_ROW_3 switch the S matrix every 3 rows
PUSCH_SEGMENT_HEADER_S_ROW_4 switch the S matrix every 4 rows
PUSCH_SEGMENT_HEADER_S_ROW_6 switch the S matrix every 6 rows
PUSCH_SEGMENT_HEADER_S_ROW_12 switch the S matrix every 12 rows

3.5.9.1.2.3.12 enum pusch_segment_header_s_scl_type_t

Enumerator

PUSCH_SEGMENT_HEADER_S_SCL_TYPE_SINGLE single shared value for all input S matrixes
PUSCH_SEGMENT_HEADER_S_SCL_TYPE_DIFFERENT different scale value for each input S matrixes

3.5.9.1.2.3.13 enum pusch_segment_header_intrp_type_t

Enumerator

PUSCH_SEGMENT_HEADER_INTRP_II2 II2 - internal interpolation with 2 referenvces.
PUSCH_SEGMENT_HEADER_INTRP_EXTERNAL external interpolation
PUSCH_SEGMENT_HEADER_INTRP_NONE no interpolation

3.5.9.1.2.4 Function Documentation

3.5.9.1.2.4.1 INLINE uint32_t PUSCH_SEGMENT_HEADER_USER_TYPE (pusch_segment_header_user_type_t *user_type_0*, pusch_segment_header_user_type_t *user_type_1*, pusch_segment_header_user_type_t *user_type_2*, pusch_segment_header_user_type_t *user_type_3*)

calculates SH[USERx_TYPE] value for User #x - used in maple_pusch_sh_t.config[11].users_type

Parameters

in	*user_type_0*	- defines what type of processing a user in layer #0 will go through in PUSCH2_EDF.
in	*user_type_1*	- defines what type of processing a user in layer #1 will go through in PUSCH2_EDF.
in	*user_type_2*	- defines what type of processing a user in layer #2 will go through in PUSCH2_EDF.
in	*user_type_3*	- defines what type of processing a user in layer #3 will go through in PUSCH2_EDF.

Returns

32 bits word with BD[USERx_TYPE] calculated value.

3.5.9.1.2.4.2 INLINE uint32_t PUSCH_SEGMENT_RB_START (uint8_t *rb_start*)

calculates BD[SEG_RB_START] value - used in maple_pusch_sh_t.config[1]

Parameters

in	*rb_start*	- Indicates the RB index of the start point of this segment from the start of the sub-frame

Returns

32 bits word with BD[SEG_RB_START] calculated value.

3.5.10 Maple PUFFT API

3.5.10.1 Overview

MAPLE PUFFT Initialization and Runtime API

MAPLE PUSCH Initialization and Runtime API

Modules

- PUFFT Initialization
- PUFFT Runtime

3.5.10.2 PUFFT Initialization

3.5.10.2.1 Overview

PUFFT device initialization API

Data Structures

- struct pufft_open_params_t
- struct maple_pufft_ch_add_int_params_t
- struct maple_pufft_ch_open_params_t

Typedefs

- typedef void(∗ maple_pufft_int_cb_t)(uint32_t status, os_virt_ptr bd_addr)
- typedef struct maple_pe_init_params_s maple_pufft_init_params_s
- typedef maple_pe_init_params_t maple_pufft_init_params_t

Enumerations

- enum
- enum { MAPLE_PUFFT_ACK_INT, MAPLE_PUFFT_NUM_ADD_INT }

Functions

- os_status maplePufftInitialize (maple_pufft_init_params_s ∗init_params, unsigned int num_devices, os_status(∗channel_dispatch)(void ∗channel, void ∗jobs, int ∗num_jobs), void(∗channel_reap)(void ∗channel, void ∗maple))

Flags for pufft_open_params_t.sector_config

- #define PUFFT_INIT_OVFL_CNT(N) (N)
- #define PUFFT_INIT_FTPE_INDX(N) ((N) << 4)

MAPLE PDSCH Device Names and IDs

- #define PUFFT_DEV_ID_0 0
- #define MAPLE_0_PUFFT_NAME "PUFFT0"
- #define MAPLE_1_PUFFT_NAME "PUFFT1"
- #define MAPLE_2_PUFFT_NAME "PUFFT2"

3.5.10.2.2 Data Structure Documentation

3.5.10.2.2.1 struct pufft_open_params_t

MAPLE PUFFT LLD Device Open paramaters.

Data Fields

- void ∗ maple_handle
- maple_pe_bd_priority_t maple_pe_bd_priority
- maple_pe_num_bd_t maple_pe_num_bd
- void ∗ antx_in_addr [MAPLE_NUM_SECTORS][MAPLE_NUM_ANT]
- uint32_t antx_in_size [MAPLE_NUM_SECTORS]
- uint32_t sector_config [MAPLE_NUM_SECTORS]
- void ∗ pre_mult_addr [MAPLE_NUM_SECTORS]
- struct {
  void ∗ ack_addr
  uint32_t ack_data

} soc_timer_conf [PUFFT_SOC_TIMER_SETS]

- uint8_t pre_mult_steering_bits [MAPLE_NUM_SECTORS]
- void * post_mult_addr [MAPLE_NUM_SECTORS]
- uint32_t fc_shift_real [MAPLE_NUM_SECTORS]
- uint32_t fc_shift_imag [MAPLE_NUM_SECTORS]
- uint16_t fc_base_real [MAPLE_NUM_SECTORS]
- uint16_t fc_base_imag [MAPLE_NUM_SECTORS]

3.5.10.2.2.1.1 Field Documentation

3.5.10.2.2.1.2 void* pufft_open_params_t::maple_handle

Handle returned from osCopDeviceOpen() for MAPLE controller.

3.5.10.2.2.1.3 maple_pe_bd_priority_t pufft_open_params_t::maple_pe_bd_priority

BD rings priority scheduling - only configured by MAPLE master.

3.5.10.2.2.1.4 maple_pe_num_bd_t pufft_open_params_t::maple_pe_num_bd

The max number of BD rings for each priority - only configured by MAPLE master.

3.5.10.2.2.1.5 void* pufft_open_params_t::antx_in_addr[MAPLE_NUM_SECTORS][MAPLE_↩ NUM_ANT]

Base addresses of the input buffer of antenna; Driver won't translate these addresses.

3.5.10.2.2.1.6 uint32_t pufft_open_params_t::antx_in_size[MAPLE_NUM_SECTORS]

Antenna input buffer size.

3.5.10.2.2.1.7 uint32_t pufft_open_params_t::sector_config[MAPLE_NUM_SECTORS]

Sector x(0..3) Initialization Configuration parameter.

Use these flags to set sector_config[x]: PUFFT_INIT_OVFL_CNT(N) PUFFT_INIT_FTPE_INDX(N)

3.5.10.2.2.1.8 void* pufft_open_params_t::pre_mult_addr[MAPLE_NUM_SECTORS]

Sector x Pre-multiplication table Address parameter (x=03); Driver won't translate these addresses.

3.5.10.2.2.1.9 void* pufft_open_params_t::ack_addr

Base address to which the Maple should reply to ack; Driver won't translate these addresses.

3.5.10.2.2.1.10 uint32_t pufft_open_params_t::ack_data

Data to ack with.

3.5.10.2.2.1.11 struct { ... } pufft_open_params_t::soc_timer_conf[PUFFT_SOC_TIMER_SETS]

Maple Sets for data and address to which the Maple should reply to ack.

3.5.10.2.2.1.12 uint8_t pufft_open_params_t::pre_mult_steering_bits[MAPLE_NUM_SECTORS]

PRM_STRPre-Multiplication table steering bits; The MAPLE-B3 uses these steering bits when fetching the pre-multiplication table Use the following flags to set steering_bits field MAPLE_STEERING_D↩SP_CLUSTER_0, MAPLE_STEERING_DSP_CLUSTER_1, MAPLE_STEERING_DSP_CLUSTER↩_2, MAPLE_STEERING_CHB, MAPLE_STEERING_MAPLE_TRG.

3.5.10.2.2.1.13 void* pufft_open_params_t::post_mult_addr[MAPLE_NUM_SECTORS]

PUFFT sector<x> Vector Post-Multiplication table Pointer.

3.5.10.2.2.1.14 uint32_t pufft_open_params_t::fc_shift_real[MAPLE_NUM_SECTORS]

PUFFT sector<x> real part of the frequency correction shift value (x = 0...3).

Values between 1 to 65535.

3.5.10.2.2.1.15 uint32_t pufft_open_params_t::fc_shift_imag[MAPLE_NUM_SECTORS]

PUFFT sector<x> imaginary part of the frequency correction shift value (x = 0...3).

Values between 1 to 65535.

3.5.10.2.2.1.16 uint16_t pufft_open_params_t::fc_base_real[MAPLE_NUM_SECTORS]

PUFFT sector<x> real part of the frequency correction base value (x = 0...3).

Values between 1 to 65535.

3.5.10.2.2.1.17 uint16_t pufft_open_params_t::fc_base_imag[MAPLE_NUM_SECTORS]

PUFFT sector<x> imaginary part of the frequency correction base value (x = 0...3).

Values between 1 to 65535.

3.5.10.2.2.2 struct maple_pufft_ch_add_int_params_t

MAPLE PUFFT channel open parameters for symbol interrupt.

Data Fields

- uint32_t int_enable:1
- uint32_t multiplexed_int:1
- os_hwi_dispatcher int_dispatcher
- maple_pufft_int_cb_t int_callback
- os_hwi_handle int_num
- os_hwi_priority int_priority

3.5.10.2.2.2.1 Field Documentation

3.5.10.2.2.2.2 uint32_t maple_pufft_ch_add_int_params_t::int_enable

Set to 1 to enable symbol interrupt for this channel.

3.5.10.2.2.2.3 uint32_t maple_pufft_ch_add_int_params_t::multiplexed_int

Set to 1 for the driver if this is a multiplexed interrupt line, for example if this interrupt line is used for multiple channels.

3.5.10.2.2.2.4 os_hwi_dispatcher maple_pufft_ch_add_int_params_t::int_dispatcher

Interrupt dispatcher to be called when interrupt occurs.

3.5.10.2.2.2.5 maple_pufft_int_cb_t maple_pufft_ch_add_int_params_t::int_callback

User call back function to be called on interrupt assertion.

3.5.10.2.2.2.6 os_hwi_handle maple_pufft_ch_add_int_params_t::int_num

Which interrupt line on this device should the interrupt handler register to.

3.5.10.2.2.2.7 os_hwi_priority maple_pufft_ch_add_int_params_t::int_priority

Interrupt priority.

3.5.10.2.2.3 struct maple_pufft_ch_open_params_t

MAPLE PUFFT channel open parameters type.

3.5.10.2.3 Macro Definition Documentation

3.5.10.2.3.1 #define PUFFT_INIT_OVFL_CNT(*N*) (N)

Allowed overrun delta.

3.5.10.2.3.2 #define PUFFT_INIT_FTPE_INDX(*N*) ((N) << 4)

FTPE index to be used for PUFFT.

3.5.10.2.3.3 #define PUFFT_DEV_ID_0 0

PUFFT device ID, there is only one PUFFT per Maple.

3.5.10.2.3.4 #define MAPLE_0_PUFFT_NAME "PUFFT0"

PUFFT device name 0 for maple_pufft_init_params_t, one PUFFT per Maple.

3.5.10.2.3.5 #define MAPLE_1_PUFFT_NAME "PUFFT1"

PUFFT device name 1 for maple_pufft_init_params_t, one PUFFT per Maple.

3.5.10.2.3.6 #define MAPLE_2_PUFFT_NAME "PUFFT2"

PUFFT device name 2 for maple_pufft_init_params_t, one PUFFT per Maple.

3.5.10.2.4 Typedef Documentation

3.5.10.2.4.1 typedef void(∗ maple_pufft_int_cb_t)(uint32_t status, os_virt_ptr bd_addr)

Pointer to user call back function to be called on interrupt assertion.

3.5.10.2.4.2 typedef struct maple_pe_init_params_s maple_pufft_init_params_s

MAPLE PUFFT initialization parameters type for one PE.

3.5.10.2.4.3 typedef maple_pe_init_params_t maple_pufft_init_params_t

MAPLE PUFFT initialization parameters type for multiple PEs.

3.5.10.2.5 Enumeration Type Documentation

3.5.10.2.5.1 anonymous enum

MAPLE PUFFT steering bits pointers mapping.

Use it for accessing relevant maple_pe_ch_open_params_t.steering_bits[x]

3.5.10.2.5.2 anonymous enum

MAPLE PUFFT additional interrupts access enumeration.

Enumerator

MAPLE_PUFFT_ACK_INT Index of ack interrupt at maple_pufft_ch_open_params_t.add_int; Callback function will return a pointer to the first cop job in queue which has not been reaped yet.

MAPLE_PUFFT_NUM_ADD_INT Total number of additional PUFFT interrupts.

3.5.10.2.6 Function Documentation

3.5.10.2.6.1 os_status maplePufftInitialize (maple_pufft_init_params_s ∗ *init_params,* unsigned int *num_devices,* os_status(∗)(void ∗channel, void ∗jobs, int ∗num_jobs) *channel_dispatch,* void(∗)(void ∗channel, void ∗maple) *channel_reap*)

Initializes the PUFFT driver's structures

The driver can supply default MAPLE parameters for initialization. The user can override these parameters by specifying an alternative MAPLE parameters structure. Although this function is called by all cores, only the master core performs the initialization of the MAPLE registers.

Parameters

in	*init_params*	- MAPLE Initialization parameters. if NULL, default MAPLE parameters will be used.
in	*num_devices*	- Number of PUFFT devices
in	*channel_dispatch*	- Pointer to channel dispatch function.
in	*channel_reap*	- Pointer to channel reap function.

Returns

OS_SUCCESS

Warning

This function is generally called by osArchInitialize() as part of the kernel and drivers

3.5.10.3 PUFFT Runtime

3.5.10.3.1 Overview

PUFFT Runtime API

Data Structures

- struct maple_pufft_job_t
- union maple_pufft_status_t
- struct pufft_mpisr_ch_params_t

MAPLE PUFFT Device Control Commands.

- #define PUFFT_SET_MPISR_CHANNELS (0x00000100 | COP_LLD_COMMAND)
- #define PUFFT_SET_MPISR_TID (0x00000200 | COP_LLD_COMMAND)
- #define PUFFT_MAX_NUM_BD_FOR_DISPACTH 14

Flags for maple_pufft_job_t.first_flags

- enum { PUFFT_OBA_STEERING_OPT0 = 0x00000000, PUFFT_OBA_STEERING_OPT1 = 0x00100000 }

- enum pufft_axi_priority_t {
  PUFFT_AXI_PRIORITY_0 = 0x00000000, PUFFT_AXI_PRIORITY_1 = 0x00010000, PUFFT_AXI_PRIORITY_2 = 0x00020000,
  PUFFT_AXI_PRIORITY_3 = 0x00030000 }
- enum maple_pufft_pre_vec_t { PUFFT_PRE_MULTI_DISABLE = 0x00000000, PUFFT_PRE_MULTI_EN_NO_UPDATE = 0x00002000, PUFFT_PRE_MULTI_EN_AND_UPDATE = 0x00006000 }
- enum maple_pufft_post_vec_t { PUFFT_POST_MULTI_DISABLE = 0x00000000, PUFFT_POST_MULTI_EN_NO_UPDATE = 0x00040000, PUFFT_POST_MULTI_EN_AND_UPDATE = 0x000C0000 }
- enum {
  PUFFT_TRANS_SIZE_128 = 0, PUFFT_TRANS_SIZE_256 = 1, PUFFT_TRANS_SIZE_512 = 2,
  PUFFT_TRANS_SIZE_1024 = 3, PUFFT_TRANS_SIZE_1536 = 4, PUFFT_TRANS_SIZE_2048 = 5 }
- #define PUFFT_BD_MANUAL_ACTIVATION 0x80000000
- #define PUFFT_INT_EN 0x10000000
- #define PUFFT_FC_GEN_EN 0x00000008
- #define PUFFT_OVA_SCL 0x00001000
- #define PUFFT_ACK_INT_EN 0x00008000
- #define PUFFT_ANT_MASK(MASK) (((uint8_t)(MASK)) << 4)

Flags for maple_pufft_job_t.second_flags

- enum maple_pufft_alt_ubw_t {
  PUFFT_ALT_UBW_OFF = 0x0, PUFFT_ALT_UBW_1_4MHZ = 0x8, PUFFT_ALT_UBW_3MHZ = 0x9,
  PUFFT_ALT_UBW_5MHZ = 0xA, PUFFT_ALT_UBW_10MHZ = 0xB, PUFFT_ALT_UBW_15MHZ = 0xC,
  PUFFT_ALT_UBW_20MHZ = 0xD }
- #define PUFFT_ADP_OVA_SCL(N) (((uint8_t)(N)) << 24)

3.5.10.3.2 Data Structure Documentation

3.5.10.3.2.1 struct maple_pufft_job_t

MAPLE PUFFT job structure.

This structure should be passed to the LLD on the cop_job_handle.device_specific field in order for the LLD to build the necessary BD

Data Fields

- uint32_t first_flags
- os_virt_ptr status_ptr
- uint32_t second_flags
- os_virt_ptr antx_input [MAPLE_NUM_ANT]
- os_virt_ptr output
- uint32_t output_gap

- uint8_t bd_index
- uint32_t ant_status
- uint16_t ant_status_short
- uint16_t timing_adj: 9

3.5.10.3.2.1.1 Field Documentation

3.5.10.3.2.1.2 uint32_t maple_pufft_job_t::first_flags

A mask provided by the user according to the flags defined above.

3.5.10.3.2.1.3 uint32_t maple_pufft_job_t::ant_status

Place holder and Timing Adjustment, always set first 16 bits to 0.

3.5.10.3.2.1.4 uint16_t maple_pufft_job_t::ant_status_short

Place holder ,always set it to 0.

3.5.10.3.2.1.5 uint16_t maple_pufft_job_t::timing_adj

This field indicates the number of Antenna samples (in 4 samples granularity) that are circularly shifted (starting from address pointed by AIBA_<x>) In order to perform a time offset correction on the FFT input buffer.

Valid values are 0... (Transform Size - 1)/4.

3.5.10.3.2.1.6 os_virt_ptr maple_pufft_job_t::status_ptr

BD Status Pointer.

3.5.10.3.2.1.7 uint32_t maple_pufft_job_t::second_flags

A mask provided by the user according to the flags defined above.

3.5.10.3.2.1.8 os_virt_ptr maple_pufft_job_t::antx_input[MAPLE_NUM_ANT]

Input buffer address of antenna x.

3.5.10.3.2.1.9 os_virt_ptr maple_pufft_job_t::output

Output Buffer Address.

3.5.10.3.2.1.10 uint32_t maple_pufft_job_t::output_gap

Output buffer address gap.

3.5.10.3.2.1.11 uint8_t maple_pufft_job_t::bd_index

Job's index in the BD ring.

Will be assigned by the driver

3.5.10.3.2.2 union maple_pufft_status_t

MAPLE PUFFT status returned to the user cb.

3.5.10.3.2.3 struct pufft_mpisr_ch_params_t

PUFFT mpisr channel parameters.

3.5.10.3.3 Macro Definition Documentation

3.5.10.3.3.1 #define PUFFT_SET_MPISR_CHANNELS (0x00000100 | COP_LLD_COMMAND)

set PUFFT MPISRs Link Parameters

3.5.10.3.3.2 #define PUFFT_SET_MPISR_TID (0x00000200 | COP_LLD_COMMAND)

set PUFFT MPISRs MMU task id for acknowledgment of SoC DSP timers pending interrupt.

3.5.10.3.3.3 #define PUFFT_MAX_NUM_BD_FOR_DISPACTH 14

Maximum number of BDs that are allowed to dispatch all at once.

3.5.10.3.3.4 #define PUFFT_BD_MANUAL_ACTIVATION 0x80000000

Prevent the driver from dispatching the job, MAPLE will wait until the user calls a Channel Ctrl Command to activate the job.

3.5.10.3.3.5 #define PUFFT_INT_EN 0x10000000

MAPLE issues an interrupt interrupt at the end of job.

3.5.10.3.3.6 #define PUFFT_FC_GEN_EN 0x00000008

Pre-frequency correction generation enable - Can be set only if Pre-Multiply is disabled.

3.5.10.3.3.7 #define PUFFT_OVA_SCL 0x00001000

Overall Scaling.
Valid only for adaptive scaling.

3.5.10.3.3.8 #define PUFFT_ACK_INT_EN 0x00008000

MAPLE issues an interrupt after receiving a symbol done interrupt.

3.5.10.3.3.9 #define PUFFT_ANT_MASK(*MASK*) (((uint8_t)(MASK)) << 4)

Used to set 8 bit mask of enabled antennas.

3.5.10.3.3.10 #define PUFFT_ADP_OVA_SCL(N) (((uint8_t)(N)) << 24)

The overall needed scaling is N.

3.5.10.3.4 Enumeration Type Documentation

3.5.10.3.4.1 anonymous enum

Output buffer base address Steering bits option.

Enumerator

PUFFT_OBA_STEERING_OPT0 Output buffer base address steering bits set according to MAPLE_PUFFT_STEERING_OBA_PTR field.

PUFFT_OBA_STEERING_OPT1 Output buffer base address steering bits set according to MAPLE_PUFFT_STEERING_ALT_OBA_PTR field.

3.5.10.3.4.2 enum pufft_axi_priority_t

AXI Bus Priority.

Enumerator

PUFFT_AXI_PRIORITY_0 Priority 0 AXI bus accesses.

PUFFT_AXI_PRIORITY_1 Priority 1 AXI bus accesses.

PUFFT_AXI_PRIORITY_2 Priority 2 AXI bus accesses.

PUFFT_AXI_PRIORITY_3 Priority 3 AXI bus accesses.

3.5.10.3.4.3 enum maple_pufft_pre_vec_t

Pre-Multiplier enable / disable.

should be set in maple_pufft_job_t.first_flags

Enumerator

PUFFT_PRE_MULTI_DISABLE Pre-Multiplier is disabled.

PUFFT_PRE_MULTI_EN_NO_UPDATE Pre-Multiplier is enabled. The PUFFT uses the pre-multiplier memory without updating it.

PUFFT_PRE_MULTI_EN_AND_UPDATE Pre-Multiplier is enabled. The MAPLE updates the pre-multiplier memory with new data.

3.5.10.3.4.4 enum maple_pufft_post_vec_t

Vector Post-Multiply Enable \ Disable.

should be set in maple_pufft_job_t.first_flags

Enumerator

PUFFT_POST_MULTI_DISABLE Post-Multiplier is disabled.
PUFFT_POST_MULTI_EN_NO_UPDATE Post-Multiplier is enabled. The PUFFT uses the post-multiplier memory without updating it.
PUFFT_POST_MULTI_EN_AND_UPDATE Post-Multiplier is enabled. The MAPLE updates the post-multiplier memory with new data.

3.5.10.3.4.5 anonymous enum

Use it to set transform size.

Should be set in maple_pufft_job_t.first_flags

Enumerator

PUFFT_TRANS_SIZE_128 Transform size 128.
PUFFT_TRANS_SIZE_256 Transform size 256.
PUFFT_TRANS_SIZE_512 Transform size 512.
PUFFT_TRANS_SIZE_1024 Transform size 1024.
PUFFT_TRANS_SIZE_1536 Transform size 1536.
PUFFT_TRANS_SIZE_2048 Transform size 2048.

3.5.10.3.4.6 enum maple_pufft_alt_ubw_t

Use it to set alternative bandwidth size.

Should be set in maple_pufft_job_t.second_flags

Enumerator

PUFFT_ALT_UBW_OFF ALT_UBW is off.
PUFFT_ALT_UBW_1_4MHZ ALT_UBW is 1.4 MHz.
PUFFT_ALT_UBW_3MHZ ALT_UBW is 3 MHz.
PUFFT_ALT_UBW_5MHZ ALT_UBW is 5 MHz.
PUFFT_ALT_UBW_10MHZ ALT_UBW is 10 MHz.
PUFFT_ALT_UBW_15MHZ ALT_UBW is 15 MHz.
PUFFT_ALT_UBW_20MHZ ALT_UBW is 20 MHz.

3.5.11 Maple PDSCH API

3.5.11.1 Overview

```
MAPLE PDSCH Initialization and Runtime API
```

Modules

- PUSCH Initialization

- PUSCH Runtime
- MAPLE PDSCH BD
- PDSCH Initialization
- PDSCH Runtime

Data Structures

- struct pdsch_sector_queue_param_t
- struct pdsch_dbg_status_param_t
- struct maple_pdsch_pid_t

Macros

- #define PDSCH_MAX_NUM_BD_FOR_DISPACTH 4

Enumerations

- enum pdsch_sbw_to_rb

3.5.11.2 Data Structure Documentation

3.5.11.2.1 struct pdsch_sector_queue_param_t

PDSCH Debug sector queue structure.

Data Fields

- uint32_t queue [8]
- uint32_t write_ptr
- uint32_t read_ptr

3.5.11.2.1.1 Field Documentation

3.5.11.2.1.1.1 uint32_t pdsch_sector_queue_param_t::queue[8]

MAPLE PDSCH2_EDF Sector Queue.

3.5.11.2.1.1.2 uint32_t pdsch_sector_queue_param_t::write_ptr

MAPLE PDSCH2_EDF Sector Queue Write Pointer.

3.5.11.2.1.1.3 uint32_t pdsch_sector_queue_param_t::read_ptr

MAPLE PDSCH2_EDF Sector Queue Read Pointer.

3.5.11.2.2 struct pdsch_dbg_status_param_t

PDSCH Debug status structure.

Data Fields

- uint32_t status [6]
- uint32_t error
- uint8_t sec_num

3.5.11.2.2.1 Field Documentation

3.5.11.2.2.1.1 uint32_t pdsch_dbg_status_param_t::status[6]

MAPLE PDSCH2_EDF Sector Status.

NOTE: User must be aware of coherency issues when providing a pointer to the driver, if the addresses are cacheable, the user must use appropriate barriers to make sure other cores will be able to read the data (if needed)

3.5.11.2.2.1.2 uint32_t pdsch_dbg_status_param_t::error

MAPLE PDSCH2_EDF Sector Error register.

NOTE: User must be aware of coherency issues when providing a pointer to the driver, if the addresses are cacheable, the user must use appropriate barriers to make sure other cores will be able to read the data (if needed)

3.5.11.2.2.1.3 uint8_t pdsch_dbg_status_param_t::sec_num

0 - 3 number indicating sector number to be updated

3.5.11.2.3 struct maple_pdsch_pid_t

PDSCH PDPE Input Dump structure;.

Data Fields

- uint8_t reserved [0x8000]
- uint32_t cw_data_offsets [PDSCH_MAX_USER_PER_SLOT][PDSCH_CW_PER_USER]
- uint8_t data [0x189C0]

3.5.11.2.3.1 Field Documentation

3.5.11.2.3.1.1 uint8_t maple_pdsch_pid_t::reserved[0x8000]

Reserved.

3.5.11.2.3.1.2 uint32_t maple_pdsch_pid_t::cw_data_offsets[PDSCH_MAX_USER_PER_SLOT][PDSCH_CW_PER_USER]

Each element is an unsigned integer pointing to the bit location of the code word relative to the beginning of the Code word data.

3.5.11.2.3.1.3 uint8_t maple_pdsch_pid_t::data[0x189C0]

A concatenation of all the code words in the sub frame starting from the first code word of the first user, followed by the second code word of the first user if exists, followed by the first code word of the second user, etc.

3.5.11.3 Macro Definition Documentation

3.5.11.3.1 #define PDSCH_MAX_NUM_BD_FOR_DISPACTH 4

Maximal number of PDSCH BD per dispatch.

Defines the maximal number of BD that can be dispatched with a single call to osCopChannelDispatch(). The larger the number the higher the stack consumption is in the driver.

Warning

Users may change this value, however it requires recompiling the drivers.

3.5.11.4 Enumeration Type Documentation

3.5.11.4.1 enum pdsch_sbw_to_rb

PDSCH Number of Resource Blocks per S_BW;.

3.5.11.5 MAPLE PDSCH BD

3.5.11.5.1 Overview

MAPLE PDSCH Buffer Descriptor.

Data Structures

- struct maple_pdsch_csi_rs_header_t
- struct maple_pdsch_csrs_paramters_t
- union maple_pdsch_job_extension_t
- struct maple_pdsch_job_t
- struct maple_pdsch_user_header_t
- struct maple_pdsch_cw_header_t
- struct maple_pdsch_sub_frm_ctrl_t

- struct maple_pdsch_ofdm_sym_ctrl_t
- struct maple_pdsch_long_re_ctrl_t
- struct maple_pdsch_short_re_ctrl_t
- struct maple_pdsch_res_block_en_t
- struct maple_pdsch_ctrl_gain_rnti_t
- struct maple_pdsch_sm_w_t
- struct maple_pdsch_ss_header_t

Macros

- #define PDSCH_CSI_HDR_STRUCT_GEN(NOC, N_CSI_RBEN)
- #define PDSCH_UE_RS_HDR_STRUCT_GEN(N_UE_SPEC_PAR, N_UE_SPEC_RBEN)
- #define PDSCH_UE_RS_HDR_STRUCT_GEN(N_UE_SPEC_PAR, N_UE_SPEC_RBEN)
- #define PDSCH_RB_MAP_STRUCT_GEN(SYS_BW)
- #define PDSCH_HDR_RES_BLOCK_EN(NUM, RBEN_ADDR) (RBEN_ADDR)->rben[(128-(NUM))/32] |= ((uint32_t)0x1 << ((NUM-1) % 32))

Enumerations

- enum { PDSCH_QUEUE_TYPE_START = 0x1, PDSCH_QUEUE_TYPE_PDPE = 0x2, PDSCH_QUEUE_TYPE_DONE = 0x4 }
- enum {
  PDSCH_USER_HEADER_ADDRESS, PDSCH_CW_HEADER_ADDRESS, PDSCH_RB_MAP_TABLE_ADDRESS,
  PDSCH_UE_RS_HEADER_ADDRESS, PDSCH_CS_RS_GAIN_CONFIG, PDSCH_MBSFN_RS_GAIN_CONFIG,
  PDSCH_POS_RS_GAIN_CONFIG, PDSCH_EXT_OFDM_SYM_ADDRESS, PDSCH_BF_COEFF_ADDRESS,
  PDSCH_SM_W_ADDRESS, PDSCH_PSS_GAIN_CONFIG, PDSCH_PSS_DATA_ADDRESS,
  PDSCH_SSS_GAIN_CONFIG, PDSCH_SSS_DATA_ADDRESS, PDSCH_ANT0_DATA_ADDRESS,
  PDSCH_ANT1_DATA_ADDRESS, PDSCH_ANT2_DATA_ADDRESS, PDSCH_ANT3_DATA_ADDRESS,
  PDSCH_ANT4_DATA_ADDRESS, PDSCH_ANT5_DATA_ADDRESS, PDSCH_ANT6_DATA_ADDRESS,
  PDSCH_ANT7_DATA_ADDRESS, PDSCH_PDPE_OUTPUT_DUMP_ADDRESS, PDSCH_CSI_HEADER_ADDRESS,
  PDSCH_N_UE_SPECIFIC_CONFIG, PDSCH_EXTERNAL_PADDING_ADDRESS, PDSCH_CSI_RS_CONFIG,
  PDSCH_PDPE_INPUT_DUMP_ADDRESS, PDSCH_PM_ADDRESS, PDSCH_NUM_HEADERS }

MAPLE PDSCH FLAGS for maple_pdsch_job_t.first_flags

- #define PDSCH_BD_MANUAL_ACTIVATION 0x80000000
- #define PDSCH_BD_INT_EN 0x10000000
- #define PDSCH_BD_SYM_INT_EN 0x04000000

- #define PDSCH_BD_EXTND_EN 0x02000000
- #define PDSCH_BD_PH_EN 0x01000000
- #define PDSCH_BD_N_USERS(N) ((N)<<16)
- #define PDSCH_USERS_NUM(JOB_PTR) (((JOB_PTR)->first_flags & 0x00FF0000) >> 16)

MAPLE PDSCH FLAGS for maple_pdsch_job_t.second_flags

- enum { PDSCH_PCS_0 = 0x00000000, PDSCH_PCS_1 = 0x80000000 }
- enum { PDSCH_LTE_FRAME_STR1 = 0x00000000, PDSCH_LTE_FRAME_STR2 = 0x00200000 }
- enum { PDSCH_NORMAL_CP = 0x00000000, PDSCH_EXTENDED_CP = 0x00100000 }
- enum { PDSCH_PBCH_FIRST_Q = 0x00000000, PDSCH_PBCH_SECOND_Q = 0x08000000, PDSCH_PBCH_THIRD_Q = 0x10000000, PDSCH_PBCH_FOURTH_Q = 0x18000000 }
- #define PDSCH_BD_SSF_EN 0x04000000
- #define PDSCH_SSFC(N) ((N) << 22)
- #define PDSCH_BD_SF_NUM(N) ((N) << 16)
- #define PDSCH_BD_NT(N) ((N) << 8)
- #define PDSCH_BD_EXT_SYM_IN_SCALE(N) ((uint8_t)(N))

MAPLE PDSCH FLAGS for maple_pdsch_job_t.third_flags

- enum { PDSCH_SF_SUBCARRIER_15KHZ = 0x00000000, PDSCH_SF_SUBCARRIER_7_5KHZ = 0x01000000 }
- enum { PDSCH_CFI1 = 0x00001000, PDSCH_CFI2 = 0x00002000, PDSCH_CFI3 = 0x00003000 }
- #define PDSCH_BD_UE_EN 0x80000000
- #define PDSCH_BD_CS_EN 0x40000000
- #define PDSCH_BD_MBSFN_EN 0x20000000
- #define PDSCH_BD_POS_EN 0x10000000
- #define PDSCH_BD_PSS_EN 0x08000000
- #define PDSCH_BD_SSS_EN 0x04000000
- #define PDSCH_BD_PBCH_EN 0x02000000
- #define PDSCH_BD_SPM_EN 0x00800000
- #define PDSCH_BD_CSI_EN 0x00400000
- #define PDSCH_BD_POD_EN 0x00200000
- #define PDSCH_BD_PID_EN 0x00100000
- #define PDSCH_BD_PRM_EN 0x00080000
- #define PDSCH_BD_PSM_EN 0x00040000
- #define PDSCH_BD_EXT_PAD_EN 0x00020000
- #define PDSCH_BD_NUM_SYMS(N) ((N)<<8)
- #define PDSCH_BD_ADP_OVA_SCL(N) ((uint8_t)(N))

maple_pdsch_job_t.hdr_config[PDSCH_XXXX_GAIN_CONFIG] defines

```
Use it for below configuration:
 maple_pdsch_job_t.hdr_config[PDSCH_CS_RS_GAIN_CONFIG]
 maple_pdsch_job_t.hdr_config[PDSCH_MBSFN_RS_GAIN_CONFIG],
 maple_pdsch_job_t.hdr_config[PDSCH_POS_RS_GAIN_CONFIG],
 maple_pdsch_job_t.hdr_config[PDSCH_PSS_GAIN_CONFIG],
```

```
maple_pdsch_job_t.hdr_config[PDSCH_SSS_GAIN_CONFIG],
maple_pdsch_job_t.hdr_config[PDSCH_CSI_RS_GAIN_CONFIG],
```

- #define PDSCH_BD_GAIN_EXPONENT(N) (((int8_t)(N) << 16) & 0x00FF0000)
- #define PDSCH_BD_GAIN_MANTISSA(N) ((uint16_t)N)
- #define PDSCH_PDPE_OUTPUT_SCALE(N) (((uint32_t)N) << 24)

maple_pdsch_job_t.hdr_config[PDSCH_PSS_GAIN_CONFIG] defines

- #define PDSCH_BD_NUM_DATA_OFDM_SYM(JOB_PTR) (((JOB_PTR)->hdr_config[PDSCH_PSS_GAIN_CONFIG] & 0xF0000000) >> 28)
- #define PDSCH_BD_NUM_CTRL_OFDM_SYM(JOB_PTR) (((JOB_PTR)->hdr_config[PDSCH_PSS_GAIN_CONFIG] & 0x0F000000) >> 24)

maple_pdsch_job_t.hdr_config[PDSCH_N_UE_SPECIFIC_CONFIG] defines

- #define PDSCH_BD_NUM_UE_SPECIFIC_RBEN(N) (((uint8_t)N) << 24)
- #define PDSCH_BD_NUM_UE_SPECIFIC_PAR(N) (((uint8_t)N) << 16)
- #define PDSCH_BD_N_PRS_RB(N) (((uint8_t)(N)) << 9)

maple_pdsch_job_t.hdr_config[PDSCH_CSI_RS_CONFIG] defines

- #define PDSCH_BD_CSI_NUM_HDR(N) ((N) << 25)
- #define PDSCH_BD_CSI_NUM_RBEN(N) ((N) << 16)
- #define PDSCH_BD_CSRS_NUM_HDR(N) ((N) << 9)
- #define PDSCH_BD_CSRS_NUM_RBEN(N) (N)

MAPLE PDSCH maple_pdsch_user_header_t.tb_params[].num_out_bits

- #define PDSCH_SOSD(MASK) ((MASK) << 24)

MAPLE PDSCH maple_pdsch_user_header_t.first_flags

- #define PDSCH_USER_RB_START(N) ((N) << 16)
- #define PDSCH_USER_ANT_EN(N) ((N) << 8)
- #define PDSCH_USER_NUM_PHYS_RB(N) (N)

MAPLE PDSCH maple_pdsch_user_header_t.second_flags

- enum
- enum
- enum { PDSCH_USER_TRANSMIT_SINGLE_ANT = 0x00000000, PDSCH_USER_TRANSMIT_DIVERSITY = 0x00000002, PDSCH_USER_SPATIAL_MULTIPLEXING = 0x00000004 }
- enum
- #define PDSCH_USER_OSPMS(N) ((N) << 24)
- #define PDSCH_USER_OSPMF(N) ((N) << 16)
- #define PDSCH_USER_LARGE_DELAY_CDD 0x00000010

- #define PDSCH_USER_CSI_OVAL_ACT 0x00000020

MAPLE PDSCH maple_pdsch_user_header_t.third_flags

- enum { PDSCH_USER_TB0_MODULATION_QPSK = 0x00000000, PDSCH_USER_TB0_MODULATION_16QAM = 0x01000000, PDSCH_USER_TB0_MODULATION_64QAM = 0x02000000 }
- enum { PDSCH_USER_TB1_MODULATION_QPSK = 0x00000000, PDSCH_USER_TB1_MODULATION_16QAM = 0x04000000, PDSCH_USER_TB1_MODULATION_64QAM = 0x08000000 }
- enum { PDSCH_USER_TB0_NL_1 = 0x00000000, PDSCH_USER_TB0_NL_2 = 0x00100000, PDSCH_USER_TB0_NL_3 = 0x00200000, PDSCH_USER_TB0_NL_4 = 0x00300000 }
- enum { PDSCH_USER_TB1_NL_1 = 0x00000000, PDSCH_USER_TB1_NL_2 = 0x00400000, PDSCH_USER_TB1_NL_3 = 0x00800000, PDSCH_USER_TB1_NL_4 = 0x00C00000 }
- #define PDSCH_USER_NIR(NUM) (NUM)
- #define PDSCH_USER_RV_IDX_TB0(IND) ((IND) << 28)
- #define PDSCH_USER_RV_IDX_TB1(IND) ((IND) << 30)

MAPLE PDSCH RB Mapping table help methods

- enum
- enum
- INLINE uint16_t PDSCH_RB_MAP_TABLE_SIZE (pdsch_mode_sys_bw_t sys_bw)
- INLINE uint16_t PDSCH_RB_MAP_SLOT_SIZE (pdsch_mode_sys_bw_t sys_bw)

MAPLE PDSCH maple_pdsch_ctrl_gain_rnti_t.ue_rs_hdr_config

- enum { PDSCH_EPDCCH = 0x00008000, PDSCH_RESET_OTHER_ANTS = 0x00004000 }
- enum { PDSCH_CGAIN_SAUET5 = 0x00000000, PDSCH_CGAIN_SAUET7 = 0x00000400, PDSCH_CGAIN_SAUET8 = 0x00000600 }
- #define PDSCH_CGAIN_UOSPMS(N) (((uint8_t)(N)) << 24)
- #define PDSCH_CGAIN_UOSPMF(N) (((uint8_t)(N)) << 16)
- #define PDSCH_CGAIN_NUM_PORTS(N) ((N) << 11)
- #define PDSCH_CGAIN_SP_ANT(N) ((N) << 7)
- #define PDSCH_CGAIN_UE5NPRB(N) (N)

MAPLE PDSCH maple_pdsch_sm_w_t.layer[x] flags

- #define PDSCH_SM_W_RNLV 0x80
- #define PDSCH_SM_W_INLV 0x40
- #define PDSCH_SM_W_SWLV 0x10
- #define PDSCH_SM_W_CNPLV(N) (N)

3.5.11.5.2 Data Structure Documentation

3.5.11.5.2.1 struct maple_pdsch_csi_rs_header_t

BD[CSI_HEADER_ADDRESS] structure; CSI Reference Signal Header Structure.

3.5.11.5.2.1.1 Field Documentation

3.5.11.5.2.1.2 uint32_t maple_pdsch_csi_rs_header_t::csi_rs_header[3]

Used for faster headers setting/ writing.

3.5.11.5.2.1.3 int16_t maple_pdsch_csi_rs_header_t::csi_cge

Signed 8 bit integer specifies the exponent part of the gain of the CSI reference signals.

3.5.11.5.2.1.4 int16_t maple_pdsch_csi_rs_header_t::csi_cgm

Signed 16 bit integer specifies the mantissa part of the gain of the CSI reference signals.

3.5.11.5.2.1.5 uint16_t maple_pdsch_csi_rs_header_t::csi_rs_ports

CSI Reference Signals Ports This field specifies the number of CSI reference signals ports used for Non zero powered CSI assignments: 0 - one CSI port (15).

1- two CSI ports (15 & 16). 2 - four CSI ports (15,16,17 & 18). 3 - eight CSI ports (15, 16, 17, 18, 19, 20, 21 & 22).

3.5.11.5.2.1.6 uint16_t maple_pdsch_csi_rs_header_t::cs_rs_ontop

CSI on top If set, CSI reference signals override REs occupied by data.

3.5.11.5.2.1.7 uint16_t maple_pdsch_csi_rs_header_t::nidcsi

This field is identical to the N ID CSI field as specified in 6.10.5.1 of 36.211.

3.5.11.5.2.1.8 uint32_t maple_pdsch_csi_rs_header_t::csi_rs_conf

CSI Reference Signals Configuration This bitmap mask field specifies the enabled CSI RS configurations.

The MS bit (on the left side) corresponds to configuration 0 while the LS bit (on the right side) corresponds to configuration 31.

3.5.11.5.2.2 struct maple_pdsch_csrs_paramters_t

BD_EXT[CSRS_HEADER_ADDRESS] structure; Cell Specific Reference Signals Headers data structure.

3.5.11.5.2.2.1 Field Documentation

3.5.11.5.2.2.2 uint16_t maple_pdsch_csrs_paramters_t::cs_rs_header[3]

Used for faster headers setting/ writing.

3.5.11.5.2.2.3 uint8_t maple_pdsch_csrs_paramters_t::cs_rs_ontop

Specifies how user data is mapped on a grid where this cell specific reference signal is located.

0 - PDSCH data skips over REs occupied by cell specific reference signals 1 - Cell specific reference signals override REs occupied by PDSCH data.

3.5.11.5.2.2.4 uint8_t maple_pdsch_csrs_paramters_t::cs_rs_ports

Specifies the number of cell specific reference signal ports.

0 - one Cell Specific Reference Signal port (0) 1 - two Cell Specific Reference Signal ports (0 & 1) 2 - four Cell Specific Reference Signal ports (0, 1, 2 & 3)

3.5.11.5.2.2.5 int8_t maple_pdsch_csrs_paramters_t::cs_cge

Signed 8 bit integer specifies the exponent part of the gain of the cell specific reference signals.

3.5.11.5.2.2.6 int16_t maple_pdsch_csrs_paramters_t::cs_cgm

Signed 16 bit integer specifies the mantissa part of the gain of the cell specific reference signals.

3.5.11.5.2.2.7 uint16_t maple_pdsch_csrs_paramters_t::cell_id

Specifies the cell ID according to which the cell specific reference signal should be generated.

3.5.11.5.2.3 union maple_pdsch_job_extension_t

MAPLE PDSCH2-EDF job's extension maple_pdsch_job_t.extension_ptr.

Data Fields

- uint32_t extension_header [16]
- os_virt_ptr import_addr
- uint32_t noit: 4
- uint32_t os_virt_ptr csrs_header_addr: 28
- uint32_t __pad0__: 8
- uint32_t os_virt_ptr fc_address: 23
- uint32_t nprsrben0 [4]
- uint32_t nprsrben1 [4]

3.5.11.5.2.3.1 Field Documentation

3.5.11.5.2.3.2 uint32_t maple_pdsch_job_extension_t::extension_header[16]

Used for faster headers setting/ writing.

3.5.11.5.2.3.3 os_virt_ptr maple_pdsch_job_extension_t::import_addr

This field points to the Import Resource Elements data structure in the system memory.

3.5.11.5.2.3.4 uint32_t maple_pdsch_job_extension_t::noit

This field specifies the number of imported types.

The following Constraint should be met: 35+NOIT+user_tokens <= 45∗(SR+1) Where: users_tokens = UE_EN ? sigma_over_all_users (ue_tokens) : 0. Where: ue_tokens = number_of_ports >= 3 ? 2 : 1

3.5.11.5.2.3.5 uint32_t os_virt_ptr maple_pdsch_job_extension_t::csrs_header_addr

CSRS_HEADER_ADDRESS - this field points to the location of the cell specific reference signals headers data structure in the system memory.

3.5.11.5.2.3.6 uint32_t maple_pdsch_job_extension_t::__pad0__

This bit specifies if the PDSCH performs frequency correction on the data prior to processing.

3.5.11.5.2.3.7 uint32_t os_virt_ptr maple_pdsch_job_extension_t::fc_address

This is the address of the frequency correction data structure.

3.5.11.5.2.3.8 uint32_t maple_pdsch_job_extension_t::nprsrben0[4]

This 100 bit vector specifies on which resource blocks position reference signals should not be placed in slot 0.

NPRSRBEN0[x]=0: Positioning Reference signals are allowed to be placed on resource block #x (subjected to NPRS_RB) NPRSRBEN0[x]=1: Positioning Reference signals are not allowed to be placed on resource block #x.

3.5.11.5.2.3.9 uint32_t maple_pdsch_job_extension_t::nprsrben1[4]

This 100 bit vector specifies on which resource blocks position reference signals should not be placed in slot 1.

NPRSRBEN1[x]=0: Positioning Reference signals are allowed to be placed on resource block #x (subjected to NPRS_RB) NPRSRBEN1[x]=1: Positioning Reference signals are not allowed to be placed on resource block #x.

3.5.11.5.2.4 struct maple_pdsch_job_t

PDSCH Job Descriptor.

This structure should be passed to the LLD on the cop_job_handle.device_specific field in order for the LLD to build the necessary BD

Data Fields

- uint32_t first_flags

- uint32_t second_flags
- uint32_t third_flags
- uint8_t bd_index
- uint8_t syms_done
- maple_pdsch_job_extension_t * extension_ptr
- os_virt_ptr hdr_data_ptrs [PDSCH_NUM_HEADERS]
- uint32_t hdr_config [PDSCH_NUM_HEADERS]

3.5.11.5.2.4.1 Field Documentation

3.5.11.5.2.4.2 uint32_t maple_pdsch_job_t::first_flags

User should provide a mask of the following defines; Should be initialized to 0; Minimal sanity check for input parameters; PDSCH_BD_INT_EN, PDSCH_BD_N_USERS, PDSCH_BD_SYM_INT_EN, PDS↩CH_BD_PH_EN,.

3.5.11.5.2.4.3 uint32_t maple_pdsch_job_t::second_flags

User should provide a mask of the following defines; Should be initialized to 0; Minimal sanity check for input parameters; PDSCH_NON_SPECIAL_SF PDSCH_SPECIAL_SF_1_2_6_7_NORMAL_CP P↩DSCH_SPECIAL_SF_3_4_8_NORMAL_CP PDSCH_SPECIAL_SF_1_2_3_6_EXTENDED_CP PD↩SCH_BD_SF_NUM(N) PDSCH_BD_EXT_SYM_IN_SCALE(N) PDSCH_LTE_FRAME_STR1 PDS↩CH_LTE_FRAME_STR2 PDSCH_NORMAL_CP PDSCH_EXTENDED_CP.

3.5.11.5.2.4.4 uint32_t maple_pdsch_job_t::third_flags

User should provide a mask of the following defines; Should be initialized to 0; Minimal sanity check for input parameters; PDSCH_BD_UE_EN PDSCH_BD_CS_EN PDSCH_BD_MBFN_EN PDSCH_↩BD_POS_EN PDSCH_BD_PSS_EN PDSCH_BD_SSS_EN PDSCH_BD_PBCH_EN PDSCH_BD_S↩PM_EN PDSCH_BD_CSI_EN PDSCH_BD_NUM_SYMS(N) PDSCH_BD_ADP_OVA_SCL(N) PD↩SCH_SF_SUBCARRIER_15KHZ PDSCH_SF_SUBCARRIER_7_5KHZ PDSCH_CFI1 PDSCH_CFI2 PDSCH_CFI3.

3.5.11.5.2.4.5 os_virt_ptr maple_pdsch_job_t::hdr_data_ptrs[PDSCH_NUM_HEADERS]

Pointers to memory locations in which supplemental headers, buffers and input data reside; Use the following access enumerations: PDSCH_USER_HEADER_ADDRESS, PDSCH_CW_HEADER_ADDRES↩S, PDSCH_RB_MAP_TABLE_ADDRESS, PDSCH_UE_RS_HEADER_ADDRESS, PDSCH_EXT_↩OFDM_SYM_ADDRESS, PDSCH_BF_DATA_COEF_ADDRESS, PDSCH_SM_W_ADDRESS, PD↩SCH_PSS_DATA_ADDRESS, PDSCH_SSS_DATA_ADDRESS, PDSCH_ANT0_DATA_ADDRES↩S, PDSCH_ANT1_DATA_ADDRESS, PDSCH_ANT2_DATA_ADDRESS, PDSCH_ANT3_DATA_↩ADDRESS, PDSCH_ANT4_DATA_ADDRESS, PDSCH_ANT5_DATA_ADDRESS, PDSCH_ANT6↩_DATA_ADDRESS, PDSCH_ANT7_DATA_ADDRESS, PDSCH_BF_CSI_COEF_ADDRESS, PDS↩CH_BF_GENRS_COEF_ADDRESS,.

3.5.11.5.2.4.6 uint32_t maple_pdsch_job_t::hdr_config[PDSCH_NUM_HEADERS]

Optional access to headers data configured inside BD; Use the following access enumerations: PDSC↩H_CS_RS_GAIN_CONFIG, PDSCH_MBSFN_RS_GAIN_CONFIG, PDSCH_POS_RS_GAIN_CON↩

FIG, PDSCH_PSS_GAIN_CONFIG, PDSCH_SSS_GAIN_CONFIG, PDSCH_CSI_ZERO_PWR_CO↩
NF, PDSCH_NUM_UE_RS_PORT_7_TILL_10, PDSCH_NUM_UE_RS_PORT_11_TILL_14, PDSC↩
H_CSI_RS_CONFIG,.

3.5.11.5.2.4.7 uint8_t maple_pdsch_job_t::bd_index

Job's index in the BD ring.

Will be assigned by the driver.

3.5.11.5.2.4.8 uint8_t maple_pdsch_job_t::syms_done

Number of symbols done, should be updated using PDSCH Ctrl Command PDSCH_CH_CMD_GET_↩
NUM_SYMS_DONE.

3.5.11.5.2.4.9 maple_pdsch_job_extension_t* maple_pdsch_job_t::extension_ptr

A pointer to the extension needed for Code Generation feature.

3.5.11.5.2.5 struct maple_pdsch_user_header_t

maple_pdsch_job_t.hdr_data_ptrs[PDSCH_USER_HEADER_ADDRESS] compatible with BD[USER↩
_HEADER_ADDRESS] structure;

Number of headers = BD[N_USERS] defined by PDSCH_BD_N_USERS(X) at maple_pdsch_job_t.↩
first_flags

Data Fields

- uint32_t first_flags
- uint32_t second_flags
- uint32_t third_flags
- uint16_t tb_size [PDSCH_CW_PER_USER]
- uint32_t num_out_bits
- os_virt_ptr tb_input

3.5.11.5.2.5.1 Field Documentation

3.5.11.5.2.5.2 uint32_t maple_pdsch_user_header_t::first_flags

Relevant flags: PDSCH_USER_RB_START(N) PDSCH_USER_ANT_EN(N) PDSCH_USER_NUM↩
_PHYS_RB(N)

3.5.11.5.2.5.3 uint32_t maple_pdsch_user_header_t::second_flags

Relevant flags: PDSCH_USER_OSPMS(N) PDSCH_USER_OSPMF(N) PDSCH_SPM_UES PDSCH↩
_USER_2_ANT_PORTS_TRANS_DIVERSITY PDSCH_USER_4_ANT_PORTS_TRANS_DIVERS↩
ITY PDSCH_USER_CDD PDSCH_USER_TRANSMIT_DIVERSITY PDSCH_USER_SPATIAL_M↩
ULTIPLEXING PDSCH_USER_1_CW PDSCH_USER_2_CW.

3.5.11.5.2.5.4 uint32_t maple_pdsch_user_header_t::third_flags

Relevant flags: PDSCH_USER_NIR(NUM) (NUM) PDSCH_USER_RV_IDX_TB0(IND) PDSCH_U↩ SER_RV_IDX_TB1(IND) PDSCH_USER_TB0_MODULATION_QPSK PDSCH_USER_TB0_MOD↩ ULATION_16QAM PDSCH_USER_TB0_MODULATION_64QAM PDSCH_USER_TB1_MODUL↩ ATION_QPSK PDSCH_USER_TB1_MODULATION_16QAM PDSCH_USER_TB1_MODULATIO↩ N_64QAM PDSCH_USER_TB0_NL_1 PDSCH_USER_TB0_NL_2 PDSCH_USER_TB0_NL_3 PD↩ SCH_USER_TB0_NL_4 PDSCH_USER_TB1_NL_1 PDSCH_USER_TB1_NL_2 PDSCH_USER_T↩ B1_NL_3 PDSCH_USER_TB1_NL_4.

3.5.11.5.2.5.5 uint16_t maple_pdsch_user_header_t::tb_size[PDSCH_CW_PER_USER]

Transport Block 0/1 Size.

3.5.11.5.2.5.6 uint32_t maple_pdsch_user_header_t::num_out_bits

Rate Matching Number of Output bits for Transport Block; See also PDSCH_SOSD(MASK)

3.5.11.5.2.5.7 os_virt_ptr maple_pdsch_user_header_t::tb_input

Physical transport block addresses.

3.5.11.5.2.6 struct maple_pdsch_cw_header_t

maple_pdsch_job_t.hdr_data_ptrs[PDSCH_CW_HEADER_ADDRESS] structure compatible with B↩ D[CW_HEADER_ADDRESS]

Number of headers = BD[N_USERS] defined by PDSCH_BD_N_USERS(X) at maple_pdsch_job_t.↩ first_flags

Data Fields

- uint32_t cw_init
- uint8_t cw_mod_type
- uint8_t cw_trgt_layer
- int8_t gain_b_exp
- int8_t gain_a_exp
- uint16_t gain_b_mantisa
- uint16_t gain_a_mantisa

3.5.11.5.2.6.1 Field Documentation

3.5.11.5.2.6.2 uint32_t maple_pdsch_cw_header_t::cw_init

Code Word Scrambling Initialization.

3.5.11.5.2.6.3 uint8_t maple_pdsch_cw_header_t::cw_mod_type

Code Word modulation type: 0- QPSK Modulation, 1- 16QAM Modulation, 2- 64QAM Modulation.

3.5.11.5.2.6.4 uint8_t maple_pdsch_cw_header_t::cw_trgt_layer

Code Word bitmask for targeting layers.

3.5.11.5.2.6.5 int8_t maple_pdsch_cw_header_t::gain_b_exp

Data Gain B exponent.

3.5.11.5.2.6.6 int8_t maple_pdsch_cw_header_t::gain_a_exp

Data Gain A exponent.

3.5.11.5.2.6.7 uint16_t maple_pdsch_cw_header_t::gain_b_mantisa

Data Gain B mantisa.

3.5.11.5.2.6.8 uint16_t maple_pdsch_cw_header_t::gain_a_mantisa

Data Gain A mantisa.

3.5.11.5.2.7 struct maple_pdsch_sub_frm_ctrl_t

MAPLE PDSCH Sub Frame Control.

The sub frame control contains parameters that are fixed for the duration of the sub frame. The number of Important Resource Element Types(NOIT) field specifiesthe number of records in this data structure.

Data Fields

- uint16_t bf_offset
- uint16_t nrbf: 4
- uint16_t roa: 1
- uint16_t bfe: 1
- uint16_t ta: 8
- uint16_t long_elm: 1
- uint16_t int8_t gain_exp:1
- uint16_t gain_mantisa

3.5.11.5.2.7.1 Field Documentation

3.5.11.5.2.7.2 uint16_t maple_pdsch_sub_frm_ctrl_t::bf_offset

Beam forming offset.

3.5.11.5.2.7.3 uint16_t maple_pdsch_sub_frm_ctrl_t::nrbf

Number of resource elements in one beam forming matrix.

3.5.11.5.2.7.4 uint16_t maple_pdsch_sub_frm_ctrl_t::roa

Reset other antennas.

3.5.11.5.2.7.5 uint16_t maple_pdsch_sub_frm_ctrl_t::bfe

Beam Forming Enable.

3.5.11.5.2.7.6 uint16_t maple_pdsch_sub_frm_ctrl_t::ta

Target Antenna.

3.5.11.5.2.7.7 uint16_t maple_pdsch_sub_frm_ctrl_t::long_elm

This bit specifies if resource elements of type <t> are long or short Must equal the OFDM Symbol Control long_elm bit.

3.5.11.5.2.7.8 uint16_t int8_t maple_pdsch_sub_frm_ctrl_t::gain_exp

Control Gain exponent.

3.5.11.5.2.7.9 uint16_t maple_pdsch_sub_frm_ctrl_t::gain_mantisa

Control Gain Mantissa.

3.5.11.5.2.8 struct maple_pdsch_ofdm_sym_ctrl_t

MAPLE PDSCH OFDM Symbol Control.

This data structure contains the number of resource elements, their size (4 or8 bytes) and their location for each OFDM symbol and for each Imported type

Data Fields

- uint32_t long_elm: 1
- uint32_t nor: 11
- uint32_t dp: 19

3.5.11.5.2.8.1 Field Documentation

3.5.11.5.2.8.2 uint32_t maple_pdsch_ofdm_sym_ctrl_t::long_elm

This bit specifies if resource elements of type <t> are long or short.

3.5.11.5.2.8.3 uint32_t maple_pdsch_ofdm_sym_ctrl_t::nor

Number of Import Resource Elements.

3.5.11.5.2.8.4 uint32_t maple_pdsch_ofdm_sym_ctrl_t::dp

IMPORT_ADDR + 72 ∗ NOIT + (LONG ? 8 : 4) ∗ DP is the location of the first resource element control of type <t> targeted to OFDM symbol <o>.

3.5.11.5.2.9 struct maple_pdsch_long_re_ctrl_t

MAPLE PDSCH Long Resource Element.

Data Fields

- uint16_t real
- uint16_t k: 12
- uint16_t imaginary
- uint16_t same_k: 12

3.5.11.5.2.9.1 Field Documentation

3.5.11.5.2.9.2 uint16_t maple_pdsch_long_re_ctrl_t::real

Real Value - The Data Field Consists of 16 Bit real value represented as 4Q12.

3.5.11.5.2.9.3 uint16_t maple_pdsch_long_re_ctrl_t::k

K is the frequency/sub-carrier position in an OFDM Symbol where the Data should be placed.

3.5.11.5.2.9.4 uint16_t maple_pdsch_long_re_ctrl_t::imaginary

Imaginary Value - The Data Field Consists of 16 Bit imaginary value represented as 4Q12.

3.5.11.5.2.9.5 uint16_t maple_pdsch_long_re_ctrl_t::same_k

This field should be identical to the K value.

3.5.11.5.2.10 struct maple_pdsch_short_re_ctrl_t

MAPLE PDSCH Short Resource Element.

Data Fields

- uint16_t real: 8
- uint16_t k: 12

3.5.11.5.2.10.1 Field Documentation

3.5.11.5.2.10.2 uint16_t maple_pdsch_short_re_ctrl_t::real

Real Value - The Data Field Consists of 16 Bit real value represented as 4Q12.

3.5.11.5.2.10.3 uint16_t maple_pdsch_short_re_ctrl_t::k

K is the frequency/sub-carrier position in an OFDM Symbol where the Data should be placed.

3.5.11.5.2.11 struct maple_pdsch_res_block_en_t

Part of BD[UE_RS_HEADER_ADDRESS] structure - resource block enable part.

Use PDSCH_HDR_RES_BLOCK_EN to set enable bits; Refer to PDSCH_UE_RS_HDR_STRUCT_G↩

EN in order to understand the final structure required for BD[UE_RS_HEADER_ADDRESS]

3.5.11.5.2.12 struct maple_pdsch_ctrl_gain_rnti_t

Part of BD[UE_RS_HEADER_ADDRESS] structure - signal gain and rnti part.

Refer to PDSCH_UE_RS_HDR_STRUCT_GEN in order to understand the final structure required for BD[UE_RS_HEADER_ADDRESS]

Data Fields

- int8_t cge0
- int16_t cgm0
- int8_t cge1
- int16_t cgm1
- uint16_t dmrsid: 9
- uint32_t uospms: 7
- uint32_t uospmf: 7
- uint32_t epdcch: 1
- uint32_t roa: 1
- uint32_t nop: 3
- uint32_t sauet: 2
- uint32_t sp_ant: 2
- uint32_t ue5_nprb: 7
- uint16_t rnti
- uint16_t scid: 2

3.5.11.5.2.12.1 Field Documentation

3.5.11.5.2.12.2 int8_t maple_pdsch_ctrl_gain_rnti_t::cge0

Control Gain Exponent for Slot 0.

3.5.11.5.2.12.3 int16_t maple_pdsch_ctrl_gain_rnti_t::cgm0

Control Gain Mantissa for Slot 0.

3.5.11.5.2.12.4 int8_t maple_pdsch_ctrl_gain_rnti_t::cge1

Control Gain Exponent for Slot 1.

3.5.11.5.2.12.5 int16_t maple_pdsch_ctrl_gain_rnti_t::cgm1

Control Gain Mantissa for Slot 1.

3.5.11.5.2.12.6 uint32_t maple_pdsch_ctrl_gain_rnti_t::uospms

UE Specific OFDM Symbol Power Map Second slot.

3.5.11.5.2.12.7 uint32_t maple_pdsch_ctrl_gain_rnti_t::uospmf

UE Specific OFDM Symbol Power Map First slot.

3.5.11.5.2.12.8 uint32_t maple_pdsch_ctrl_gain_rnti_t::epdcch

ePDCCH - The header generates Demodulation reference signals associated with ePDCCH.

This field is valid only if NOP=0.

3.5.11.5.2.12.9 uint32_t maple_pdsch_ctrl_gain_rnti_t::roa

ROA - The UE specific header writes to selected virtual antennas and all other antennas are reset.

This field is valid only if NOP=0 and PDSCH_EPDCCH_EN is set.

3.5.11.5.2.12.10 uint32_t maple_pdsch_ctrl_gain_rnti_t::nop

Number of ports to which this UE specific reference signal is directed.

3.5.11.5.2.12.11 uint32_t maple_pdsch_ctrl_gain_rnti_t::sauet

Single Antenna UE Specific Type.

3.5.11.5.2.12.12 uint32_t maple_pdsch_ctrl_gain_rnti_t::sp_ant

Single Port target Antenna.

This field specifies the target antenna. This field is valid if NOP=0 & (EPDCCH | SAUET=0).

3.5.11.5.2.12.13 uint32_t maple_pdsch_ctrl_gain_rnti_t::ue5_nprb

Number of ports to which this UE specific reference signal is directed.

3.5.11.5.2.12.14 uint16_t maple_pdsch_ctrl_gain_rnti_t::dmrsid

This field is used for calculating the c_init during the scrambling stage when target port is not 5.

3.5.11.5.2.12.15 uint16_t maple_pdsch_ctrl_gain_rnti_t::rnti

Radio Network temporary identifier.

3.5.11.5.2.12.16 uint16_t maple_pdsch_ctrl_gain_rnti_t::scid

This bit is used for calculating c_init during the scrambling stage as specified in 6.10.3.1 in 3GPP TS 36.211 V10.4.0 (2011-12).

It is used only in ports 7-14.

3.5.11.5.2.13 struct maple_pdsch_sm_w_t

maple_pdsch_job_t.hdr_data_ptrs[PDSCH_SM_W_ADDRESS] structure Compatible with one entry of BD[SM_W_ADDRESS] for 1 matrix

3.5.11.5.2.14 struct maple_pdsch_ss_header_t

BD[[PSS][SSS]_DATA_ADDRESS] structure; Synchronization Channel Gain and Data.

3.5.11.5.2.14.1 Field Documentation

3.5.11.5.2.14.2 uint16_t maple_pdsch_ss_header_t::data_re

The Data Field Consists of 16 Bit Re structure, represented in 4Q12.

3.5.11.5.2.14.3 uint16_t maple_pdsch_ss_header_t::k0_re

K is the frequency/sub-carrier position in an OFDM Symbol.

3.5.11.5.2.14.4 uint16_t maple_pdsch_ss_header_t::data_img

The Data Field Consists of 16 Bit Img structure, represented in 4Q12.

3.5.11.5.2.14.5 uint16_t maple_pdsch_ss_header_t::k0_img

K is the frequency/sub-carrier position in an OFDM Symbol.

3.5.11.5.3 Macro Definition Documentation

3.5.11.5.3.1 #define PDSCH_BD_MANUAL_ACTIVATION 0x80000000

Prevent the driver from dispatching the job, MAPLE will wait until the user calls a Channel Ctrl Command to activate the job.

3.5.11.5.3.2 #define PDSCH_BD_INT_EN 0x10000000

BD[INT_EN].

3.5.11.5.3.3 #define PDSCH_BD_SYM_INT_EN 0x04000000

Symbol interrupt enable.

3.5.11.5.3.4 #define PDSCH_BD_EXTND_EN 0x02000000

Extended BD Enabled.

If set then this BD has 256 bytes instead of 128 bytes.

3.5.11.5.3.5 #define PDSCH_BD_PH_EN 0x01000000

Place Holder.

If set, specifies the number of control symbols as well as the number of data symbols is 0.

3.5.11.5.3.6 #define PDSCH_BD_N_USERS(*N*) ((N)<<16)

BD[N_USERS]; Total number of users in subframe allocated in the current cell; PBCH channel is referred as one of users.

3.5.11.5.3.7 #define PDSCH_USERS_NUM(*JOB_PTR*) (((JOB_PTR)->first_flags & 0x00FF0000) >> 16)

Help function to calculate the number of USERS.

3.5.11.5.3.8 #define PDSCH_BD_SSF_EN 0x04000000

BD[SSFC]; Special subframe configuration for UE specific Reference Signals generation in TDD mode.

Enable special subframe configuration for UE specific Reference Signals.

3.5.11.5.3.9 #define PDSCH_SSFC(*N*) ((N) << 22)

Configuration of Special subframe in case of frame structure 2 Valid values are 0..9.

3.5.11.5.3.10 #define PDSCH_BD_SF_NUM(*N*) ((N) << 16)

BD[SF_NUM]; Sub-Frame Number.

3.5.11.5.3.11 #define PDSCH_BD_NT(*N*) ((N) << 8)

BD[NT]; Number of Tail OFDM Symbols.

3.5.11.5.3.12 #define PDSCH_BD_EXT_SYM_IN_SCALE(*N*) ((uint8_t)(N))

BD[EXT_SYM_IN_SCALE]; External Symbols input scaling.

3.5.11.5.3.13 #define PDSCH_BD_UE_EN 0x80000000

BD[UE_EN]; UE Specific Reference Signal Enable.

3.5.11.5.3.14 #define PDSCH_BD_CS_EN 0x40000000

BD[CS_EN]; Cell Specific Reference Signal Enable.

3.5.11.5.3.15 #define PDSCH_BD_MBSFN_EN 0x20000000

BD[MBSFN]; MBSFN Reference Signal Enable.

3.5.11.5.3.16 #define PDSCH_BD_POS_EN 0x10000000

BD[POS_EN]; Positioning Reference Signal Enable.

3.5.11.5.3.17 #define PDSCH_BD_PSS_EN 0x08000000

BD[PSS_EN]; Primary Synchronization Channel Enable.

3.5.11.5.3.18 #define PDSCH_BD_SSS_EN 0x04000000

BD[SSS_EN]; Secondary Synchronization Channel Enable.

3.5.11.5.3.19 #define PDSCH_BD_PBCH_EN 0x02000000

BD[PBCH_EN]; This field indicated whether Physical Broadcast Channel exists at current sub-frame or not.

3.5.11.5.3.20 #define PDSCH_BD_SPM_EN 0x00800000

BD[SPM_EN]; Enable spatial multiplexing.

3.5.11.5.3.21 #define PDSCH_BD_CSI_EN 0x00400000

BD[CSI_EN]; CSI reference signal enable.

3.5.11.5.3.22 #define PDSCH_BD_POD_EN 0x00200000

BD[POD]; PDPE Output Dump Enabled - If set, This field specifies that all the PDPE Output will be written to the system memory starting the address written at PDSCH_PDPE_OUTPUT_DUMP_ADDR↩ ESS field.

If IFFT_DIS == 1, than POD must be set to null

3.5.11.5.3.23 #define PDSCH_BD_PID_EN 0x00100000

BD[PID]; PDPE Input Dump Enabled - If set, This field specifies that all the PDPE Input will be written to the system memory starting the address written at PDSCH_PDPE_INPUT_DUMP_ADDRESS field.

3.5.11.5.3.24 #define PDSCH_BD_PRM_EN 0x00080000

BD[PRM]; Pre-Multiplication Enable - This field specifies if the pre-IFFT samples are multiplied before IFFT.

3.5.11.5.3.25 #define PDSCH_BD_PSM_EN 0x00040000

BD[PSM]; Post-Multiplication Enable - This field specifies if the IFFT results are post multiplied.

3.5.11.5.3.26 #define PDSCH_BD_EXT_PAD_EN 0x00020000

BD[EXTPAD]; External Padding Enabled - If set, This field determines whether padding is originated from a register that is read cyclically or is read from a buffer in the system memory pointed to by PDSC↩ H_PAD_ADDRESS.

3.5.11.5.3.27 #define PDSCH_BD_NUM_SYMS(*N*) ((N)<<8)

BD[NUM_SYMS]; Number of Symbols.

3.5.11.5.3.28 #define PDSCH_BD_ADP_OVA_SCL(*N*) ((uint8_t)(N))

BD[ADP_OVA_SCL]; Adaptive Overall Scaling.

3.5.11.5.3.29 #define PDSCH_BD_GAIN_EXPONENT(*N*) (((int8_t)(N) << 16) & 0x00FF0000)

Gain exponent.

3.5.11.5.3.30 #define PDSCH_BD_GAIN_MANTISSA(*N*) ((uint16_t)N)

Gain mantissa.

3.5.11.5.3.31 #define PDSCH_PDPE_OUTPUT_SCALE(*N*) (((uint32_t)N) << 24)

This field indicates the scaling at the output of the PDPE for the last OFDM symbol processed in this BD. It is a 2s complement number. should be used only PDSCH_SSS_GAIN_CONFIG.

3.5.11.5.3.32 #define PDSCH_BD_NUM_DATA_OFDM_SYM(*JOB_PTR*) (((JOB_↩ PTR)->hdr_config[PDSCH_PSS_GAIN_CONFIG] & 0xF0000000) >> 28)

Help function to get the number of Data OFDM Symbols.

3.5.11.5.3.33 #define PDSCH_BD_NUM_CTRL_OFDM_SYM(*JOB_PTR*) (((JOB_↩ PTR)->hdr_config[PDSCH_PSS_GAIN_CONFIG] & 0x0F000000) >> 24)

Help function to get the number of Control OFDM Symbols.

3.5.11.5.3.34 #define PDSCH_BD_NUM_UE_SPECIFIC_RBEN(*N*) (((uint8_t)N) << 24)

Number of UE specific Reference Signals Parameter headers.

3.5.11.5.3.35 #define PDSCH_BD_NUM_UE_SPECIFIC_PAR(*N*) (((uint8_t)N) << 16)

Number of UE specific Reference Signals Resource Block Enable headers.

3.5.11.5.3.36 #define PDSCH_BD_N_PRS_RB(*N*) (((uint8_t)(N)) << 9)

Number of resource blocks containing Positioning Reference Signals.
Valid only if POS_EN is set to 1. Valid values: 0 to 100.

3.5.11.5.3.37 #define PDSCH_BD_CSI_NUM_HDR(*N*) ((N) << 25)

Number of CSI RS headers.

3.5.11.5.3.38 #define PDSCH_BD_CSI_NUM_RBEN(*N*) ((N) << 16)

Number of CSI RS Resource block enable headers.

3.5.11.5.3.39 #define PDSCH_BD_CSRS_NUM_HDR(*N*) ((N) << 9)

Number of Cell Specific RS headers.

3.5.11.5.3.40 #define PDSCH_BD_CSRS_NUM_RBEN(*N*) (N)

Number of Cell Specific RS Resource block enable headers.

3.5.11.5.3.41 #define PDSCH_CSI_HDR_STRUCT_GEN(*NOC, N_CSI_RBEN*)

maple_pdsch_job_t.hdr_data_ptrs[PDSCH_CSI_HEADER_ADDRESS] structure generated based on N_CSI_RBEN; Compatible with BD[PDSCH_CSI_HEADER_ADDRESS] structure

Use it inside your application for static allocations

3.5.11.5.3.42 #define PDSCH_UE_RS_HDR_STRUCT_GEN(*N_UE_SPEC_PAR, N_UE_SPEC_RBEN*)

maple_pdsch_job_t.hdr_data_ptrs[PDSCH_UE_RS_HEADER_ADDRESS] structure generated based on N_UE_SPEC_PR; Compatible with BD[UE_RS_HEADER_ADDRESS] structure

Use it inside your application for static allocations

3.5.11.5.3.43 #define PDSCH_UE_RS_HDR_STRUCT_GEN(*N_UE_SPEC_PAR, N_UE_SPEC_RBEN*)

maple_pdsch_job_t.hdr_data_ptrs[PDSCH_UE_RS_HEADER_ADDRESS] structure generated based on N_UE_SPEC_PR; Compatible with BD[UE_RS_HEADER_ADDRESS] structure

Use it inside your application for static allocations

3.5.11.5.3.44 #define PDSCH_SOSD(*MASK*) ((MASK) << 24)

This MASK specifies which OFDM symbols in the slot the user data should not be mapped to.

3.5.11.5.3.45 #define PDSCH_USER_RB_START(*N*) ((N) << 16)

Starting Resource Block index of the current user.

3.5.11.5.3.46 #define PDSCH_USER_ANT_EN(*N*) ((N) << 8)

Bits indicating which antennas this user is targeting.

3.5.11.5.3.47 #define PDSCH_USER_NUM_PHYS_RB(*N*) (N)

This field specifies the number of Physical Resource Blocks that are allocated to this user.

3.5.11.5.3.48 #define PDSCH_USER_OSPMS(*N*) ((N) << 24)

OFDM Symbol Power map second slot.

3.5.11.5.3.49 #define PDSCH_USER_OSPMF(*N*) ((N) << 16)

OFDM Symbol Power map first slot.

3.5.11.5.3.50 #define PDSCH_USER_LARGE_DELAY_CDD 0x00000010

Large Delay CDD Mode.

3.5.11.5.3.51 #define PDSCH_USER_CSI_OVAL_ACT 0x00000020

CSI Reference Signal Overlay Action for user.

3.5.11.5.3.52 #define PDSCH_USER_NIR(*NUM*) (NUM)

Maximum number of allowed bits after turbo encoding as defined in TS36.212 5.1.4.1.2.

3.5.11.5.3.53 #define PDSCH_USER_RV_IDX_TB0(*IND*) ((IND) << 28)

Rate Matching redundancy version number.

3.5.11.5.3.54 #define PDSCH_USER_RV_IDX_TB1(*IND*) ((IND) << 30)

Rate Matching redundancy version number.

3.5.11.5.3.55 #define PDSCH_RB_MAP_STRUCT_GEN(*SYS_BW*)

maple_pdsch_job_t.hdr_data_ptrs[PDSCH_RB_MAP_TABLE_ADDRESS] structure compatible B↩ D[RB_MAP_TABLE_ADDRESS]

The RB Map structure is generated based on the System Bandwidth; Use it inside your application for static allocations

3.5.11.5.3.56 #define PDSCH_HDR_RES_BLOCK_EN(*NUM, RBEN_ADDR*) (RBEN_ADDR)->rben[(128-(NUM))/32] |= ((uint32_t)0x1 << ((NUM-1) % 32))

Use this macro to enable resource blocks inside the maple_pdsch_res_block_en_t.

3.5.11.5.3.57 #define PDSCH_CGAIN_UOSPMS(*N*) (((uint8_t)(N)) << 24)

UE Specific OFDM Symbol Power Map Second slot.

3.5.11.5.3.58 #define PDSCH_CGAIN_UOSPMF(*N*) (((uint8_t)(N)) << 16)

UE Specific OFDM Symbol Power Map First slot.

3.5.11.5.3.59 #define PDSCH_CGAIN_NUM_PORTS(*N*) ((N) << 11)

Number of ports to which this UE specific reference signal is directed.

3.5.11.5.3.60 #define PDSCH_CGAIN_SP_ANT(*N*) ((N) << 7)

Single Port target Antenna.

This field specifies the target antenna. This field is valid if NOP=0 & (EPDCCH | SAUET=0).

3.5.11.5.3.61 #define PDSCH_CGAIN_UE5NPRB(*N*) (N)

Number of ports to which this UE specific reference signal is directed.

3.5.11.5.3.62 #define PDSCH_SM_W_RNLV 0x80

The 2s complement of the 16 bit real portion of the complex number pointed to by the CNP field should be used.

3.5.11.5.3.63 #define PDSCH_SM_W_INLV 0x40

The 2s complement of the 16 bit imaginary portion of the complex number pointed to by the CNP field should be used.

3.5.11.5.3.64 #define PDSCH_SM_W_SWLV 0x10

The UDSMRCx & UDSMIC should be used as imaginary and real respectively.

3.5.11.5.3.65 #define PDSCH_SM_W_CNPLV(*N*) (N)

Complex Number Pointer Layer x Virtual Antenna.

3.5.11.5.4 Enumeration Type Documentation

3.5.11.5.4.1 anonymous enum

PDSCH Debug Queue type Enumeration.

Enumerator

PDSCH_QUEUE_TYPE_START MAPLE PDSCH2_EDF PDSCH_START Queue.
PDSCH_QUEUE_TYPE_PDPE MAPLE PDSCH2_EDF PDSCH_PDPE Queue.
PDSCH_QUEUE_TYPE_DONE MAPLE PDSCH2_EDF PDSCH_DONE Queue.

3.5.11.5.4.2 anonymous enum

PDSCH Header Enumeration.

Enumerator

PDSCH_USER_HEADER_ADDRESS Access to BD[USER_HEADER_ADDRESS].
PDSCH_CW_HEADER_ADDRESS Access to BD[CW_HEADER_ADDRESS].
PDSCH_RB_MAP_TABLE_ADDRESS Access to BD[RB_MAP_TABLE_ADDRESS].

PDSCH_UE_RS_HEADER_ADDRESS Access to BD[UE_RS_HEADER_ADDRESS].
PDSCH_CS_RS_GAIN_CONFIG Access to BD[CS_CGE + CS_CGM] data.
PDSCH_MBSFN_RS_GAIN_CONFIG Access to BD[MBSFN_CGE + MBSFN_CGM] data.
PDSCH_POS_RS_GAIN_CONFIG Access to BD[POS_CGE + POS_CGM] data.
PDSCH_EXT_OFDM_SYM_ADDRESS Access to BD[EXT_OFDM_SYM_ADDRESS].
PDSCH_BF_COEFF_ADDRESS Access to BD[PDSCH_BF_COEFF_ADDRESS], supported at Rev2 only.
PDSCH_SM_W_ADDRESS Access to BD[SM_W_ADDRESS].
PDSCH_PSS_GAIN_CONFIG Access to BD[PSS_CGE + PSS_CGM] + Read only BD[ND + NC].
PDSCH_PSS_DATA_ADDRESS Access to BD[PSS_DATA_ADDRESS].
PDSCH_SSS_GAIN_CONFIG Access to BD[POS + SSS_CCE + SSS_CCM].
PDSCH_SSS_DATA_ADDRESS Access to BD[SSS_DATA_ADDRESS].
PDSCH_ANT0_DATA_ADDRESS Access to BD[ANT0_DATA_ADDRESS].
PDSCH_ANT1_DATA_ADDRESS Access to BD[ANT1_DATA_ADDRESS].
PDSCH_ANT2_DATA_ADDRESS Access to BD[ANT2_DATA_ADDRESS].
PDSCH_ANT3_DATA_ADDRESS Access to BD[ANT3_DATA_ADDRESS].
PDSCH_ANT4_DATA_ADDRESS Access to BD[ANT4_DATA_ADDRESS].
PDSCH_ANT5_DATA_ADDRESS Access to BD[ANT5_DATA_ADDRESS].
PDSCH_ANT6_DATA_ADDRESS Access to BD[ANT6_DATA_ADDRESS].
PDSCH_ANT7_DATA_ADDRESS Access to BD[ANT7_DATA_ADDRESS].
PDSCH_PDPE_OUTPUT_DUMP_ADDRESS Access to BD[POD_ADDRESS], supported at Rev2 only.
PDSCH_CSI_HEADER_ADDRESS Access to BD[CSI_HEADER_ADDRESS].
PDSCH_N_UE_SPECIFIC_CONFIG Access to BD[N_UE_SPECIFIC_RBEN + N_UE_SPECIFIC_PAR + N_PRS_RB].
PDSCH_EXTERNAL_PADDING_ADDRESS Access to BD[PAD_ADDRESS], valid only if PDSCH_BD_EXT_PAD_EN is set.
PDSCH_CSI_RS_CONFIG Access to BD[NOC + N_CSI_RBEN + NOCSRS + N_CSRS_RBEN].

PDSCH_PDPE_INPUT_DUMP_ADDRESS Access to BD[PID_ADDRESS].
PDSCH_PM_ADDRESS Pre/Post Multiplication Address.
PDSCH_NUM_HEADERS Number of PDSCH headers in BD.

3.5.11.5.4.3 anonymous enum

BD[PCS]; Primary Configuration Select.

This bit specifies which configuration set should be used for processing the current BD.

Enumerator

PDSCH_PCS_0 Configuration 0 should be used.
PDSCH_PCS_1 Configuration 1 should be used.

3.5.11.5.4.4 anonymous enum

BD[FS]; Frame structure.

Enumerator

PDSCH_LTE_FRAME_STR1 LTE frame structure 1.
PDSCH_LTE_FRAME_STR2 LTE frame structure 2.

3.5.11.5.4.5 anonymous enum

BD[EXTND_CP]; Extended Cyclic Prefix.

Enumerator

PDSCH_NORMAL_CP Normal CP.
PDSCH_EXTENDED_CP Extended CP.

3.5.11.5.4.6 anonymous enum

BD[PBFIX]; PBCH Frame Index.

Enumerator

PDSCH_PBCH_FIRST_Q First quarter (bits 0-479 for normal CP and 0-431 for extended CP)
PDSCH_PBCH_SECOND_Q Second quarter (bits 480-959 for normal CP and 432-863 for extended CP)
PDSCH_PBCH_THIRD_Q Third quarter (bits 960-1439 for normal CP and 864-1295 for extended CP)
PDSCH_PBCH_FOURTH_Q Fourth quarter (bits 1440 to 1919 for normal CP and 1296-1727 for extended CP)

3.5.11.5.4.7 anonymous enum

BD[SF_7_5KHZ]; subframe subcarrier enable.

Enumerator

PDSCH_SF_SUBCARRIER_15KHZ 15khz subcarrier
PDSCH_SF_SUBCARRIER_7_5KHZ 7_5khz subcarrier

3.5.11.5.4.8 anonymous enum

BD[CFI]; Number of external PDCCH/PHICH OFDM symbols pointed by [EXT_OFDM_SYM_ADD↩ RESS].

Enumerator

PDSCH_CFI1 1 Symbols
PDSCH_CFI2 2 Symbols
PDSCH_CFI3 3 Symbols

3.5.11.5.4.9 anonymous enum

Enumerator

PDSCH_USER_TRANSMIT_SINGLE_ANT Single Antenna Transmission.
PDSCH_USER_TRANSMIT_DIVERSITY Transmit Diversity.
PDSCH_USER_SPATIAL_MULTIPLEXING Spatial Multiplexing.

3.5.11.5.4.10 anonymous enum

Enumerator

PDSCH_USER_TB0_MODULATION_QPSK QPSK Modulation.
PDSCH_USER_TB0_MODULATION_16QAM 16QAM Modulation
PDSCH_USER_TB0_MODULATION_64QAM 64QAM Modulation

3.5.11.5.4.11 anonymous enum

Enumerator

PDSCH_USER_TB1_MODULATION_QPSK QPSK Modulation.
PDSCH_USER_TB1_MODULATION_16QAM 16QAM Modulation
PDSCH_USER_TB1_MODULATION_64QAM 64QAM Modulation

3.5.11.5.4.12 anonymous enum

Enumerator

PDSCH_USER_TB0_NL_1 Number of Layers parameter = 1.
PDSCH_USER_TB0_NL_2 Number of Layers parameter = 2.
PDSCH_USER_TB0_NL_3 Number of Layers parameter = 3.
PDSCH_USER_TB0_NL_4 Number of Layers parameter = 4.

3.5.11.5.4.13 anonymous enum

Enumerator

PDSCH_USER_TB1_NL_1 Number of Layers parameter = 1.
PDSCH_USER_TB1_NL_2 Number of Layers parameter = 2.
PDSCH_USER_TB1_NL_3 Number of Layers parameter = 3.
PDSCH_USER_TB1_NL_4 Number of Layers parameter = 4.

3.5.11.5.4.14 anonymous enum

Enumerator

PDSCH_EPDCCH ePDCCH - The header generates Demodulation reference signals associated with ePDCCH. This field is valid only if NOP=0.
PDSCH_RESET_OTHER_ANTS ROA - The UE specific header writes to selected virtual antennas and all other antennas are reset. This field is valid only if NOP=0 and PDSCH_EPDCCH_EN is set.

3.5.11.5.4.15 anonymous enum

Enumerator

PDSCH_CGAIN_SAUET5 UE specific reference signal targeting antenna port 5.
PDSCH_CGAIN_SAUET7 UE specific reference signal targeting antenna port 7.
PDSCH_CGAIN_SAUET8 UE specific reference signal targeting antenna port 8.

3.5.11.5.5 Function Documentation

3.5.11.5.5.1 INLINE uint16_t PDSCH_RB_MAP_TABLE_SIZE (pdsch_mode_sys_bw_t *sys_bw*)

Returns total RB Mapping table size by System Bandwidth

Parameters

in	*sys_bw*	- System Bandwidth

Returns

RB Mapping table size by System Bandwidth

3.5.11.5.5.2 INLINE uint16_t PDSCH_RB_MAP_SLOT_SIZE (pdsch_mode_sys_bw_t *sys_bw*)

Returns each slot size by System Bandwidth

Parameters

in	*sys_bw*	- System Bandwidth

Returns

Slot's size by System Bandwidth

3.5.12 Maple TCPE API

3.5.12.1 Overview

MAPLE TCPE Initialization and Runtime API

Modules

- TCPE Initialization
- TCPE Runtime

3.5.12.2 TCPE Initialization

3.5.12.2.1 Overview

TCPE device initialization API

Data Structures

- struct tcpe_open_params_t

Typedefs

- typedef maple_pe_init_params_t maple_tcpe_init_params_t
- typedef struct maple_pe_init_params_s maple_tcpe_init_params_s
- typedef maple_pe_ch_open_params_t maple_tcpe_ch_open_params_t

Enumerations

- enum

Functions

- os_status mapleTcpeInitialize (maple_tcpe_init_params_s *init_params, unsigned int num_devices, os_status(*channel_dispatch)(void *channel, void *jobs, int *num_jobs), void(*channel_reap)(void *channel, void *maple))

MAPLE TCPE Device Names and IDs

- #define MAPLE_TCPE_NAME "TCPE"
- #define TCPE_DEV_ID 0

3.5.12.2.2 Data Structure Documentation

3.5.12.2.2.1 struct tcpe_open_params_t

MAPLE TCPE Device Open parameters.

Data Fields

- void * maple_handle
- maple_pe_bd_priority_t maple_pe_bd_priority
- maple_pe_num_bd_t maple_pe_num_bd

3.5.12.2.2.1.1 Field Documentation

3.5.12.2.2.1.2 void* tcpe_open_params_t::maple_handle

Handle returned from osCopDeviceOpen() for MAPLE controller.

3.5.12.2.2.1.3 maple_pe_bd_priority_t tcpe_open_params_t::maple_pe_bd_priority

BD rings priority scheduling - only configured by MAPLE master.

3.5.12.2.2.1.4 maple_pe_num_bd_t tcpe_open_params_t::maple_pe_num_bd

The number of BD rings for each priority - only configured by MAPLE master.

3.5.12.2.3 Macro Definition Documentation

3.5.12.2.3.1 #define MAPLE_TCPE_NAME "TCPE"

TCPE device name for Maple 0.

3.5.12.2.3.2 #define TCPE_DEV_ID 0

TCPE device id for inside one Maple.

3.5.12.2.4 Typedef Documentation

3.5.12.2.4.1 typedef maple_pe_init_params_t maple_tcpe_init_params_t

MAPLE TCPE initialization parameters type for multiple PEs.

3.5.12.2.4.2 typedef struct maple_pe_init_params_s maple_tcpe_init_params_s

MAPLE TCPE initialization parameters type for one PE.

3.5.12.2.4.3 typedef maple_pe_ch_open_params_t maple_tcpe_ch_open_params_t

MAPLE TCPE channel open parameters type.

3.5.12.2.5 Enumeration Type Documentation

3.5.12.2.5.1 anonymous enum

MAPLE TCPE steering bits pointers mapping.

Use it for accessing relevant maple_pe_ch_open_params_t.steering_bits[x]

3.5.12.2.6 Function Documentation

3.5.12.2.6.1 os_status mapleTcpeInitialize (maple_tcpe_init_params_s * *init_params,* unsigned int *num_devices,* os_status(*)(void *channel, void *jobs, int *num_jobs) *channel_dispatch,* void(*)(void *channel, void *maple) *channel_reap*)

Initializes the TCPE driver's structures

The driver can supply default MAPLE parameters for initialization. The user can override these parameters by specifying an alternative MAPLE parameters structure. Although this function is called by all cores, only the master core performs the initialization of the MAPLE registers.

Parameters

in	*init_params*	- MAPLE Initialization parameters. if NULL, default MAPLE parameters will be used.
in	*num_devices*	- Number of TCPE devices
in	*channel_dispatch*	- Pointer to channel dispatch function.
in	*channel_reap*	- Pointer to channel reap function.

Returns

OS_SUCCESS

Warning

This function is generally called by osArchInitialize() as part of the kernel and drivers

3.5.12.3 TCPE Runtime

3.5.12.3.1 Overview

TCPE Runtime API

Data Structures

- struct maple_tcpe_ucb_t
- struct maple_tcpe_uiobc_t
- struct maple_tcpe_job_t

Macros

- #define TCPE_MAX_NUM_BD_FOR_DISPACTH 16

Flags & parameters for maple_tcpe_job_t

- enum {
  TCPE_MB_PRIORITY_0 = 0, TCPE_MB_PRIORITY_1 = 1, TCPE_MB_PRIORITY_2 = 2,
  TCPE_MB_PRIORITY_3 = 3 }
- #define TCPE_BD_MANUAL_ACTIVATION 0x80000000
- #define TCPE_BD_INT_EN 0x10000000

Flags & parameters for maple_tcpe_ucb_t

- enum {
  TCPE_EXTERNAL_PILOTS = 0, TCPE_3_PILOTS = 2, TCPE_4_PILOTS = 3,
  TCPE_5_PILOTS = 4, TCPE_6_PILOTS = 5, TCPE_7_PILOTS = 6,
  TCPE_8_PILOTS = 7 }
- enum {
  TCPE_SIGNATURE_0 = 0, TCPE_SIGNATURE_1 = 1, TCPE_SIGNATURE_2 = 2,
  TCPE_SIGNATURE_3 = 3, TCPE_SIGNATURE_4 = 4, TCPE_SIGNATURE_5 = 5,
  TCPE_SIGNATURE_6 = 6, TCPE_SIGNATURE_7 = 7, TCPE_SIGNATURE_8 = 8,
  TCPE_SIGNATURE_9 = 9, TCPE_SIGNATURE_10 = 10, TCPE_SIGNATURE_11 = 11,
  TCPE_SIGNATURE_12 = 12, TCPE_SIGNATURE_13 = 13, TCPE_SIGNATURE_14 = 14,
  TCPE_SIGNATURE_15 = 15 }
- enum {
  TCPE_SEARCH_WINDOW_32_CHIPS = 0, TCPE_SEARCH_WINDOW_64_CHIPS = 1, TC↩
  PE_SEARCH_WINDOW_96_CHIPS = 2,
  TCPE_SEARCH_WINDOW_128_CHIPS = 3, TCPE_SEARCH_WINDOW_160_CHIPS = 4, T↩
  CPE_SEARCH_WINDOW_192_CHIPS = 5,
  TCPE_SEARCH_WINDOW_224_CHIPS = 6, TCPE_SEARCH_WINDOW_256_CHIPS = 7, T↩
  CPE_SEARCH_WINDOW_288_CHIPS = 8,
  TCPE_SEARCH_WINDOW_320_CHIPS = 9, TCPE_SEARCH_WINDOW_352_CHIPS = 10,
  TCPE_SEARCH_WINDOW_384_CHIPS = 11,
  TCPE_SEARCH_WINDOW_416_CHIPS = 12, TCPE_SEARCH_WINDOW_448_CHIPS = 13,
  TCPE_SEARCH_WINDOW_480_CHIPS = 14,
  TCPE_SEARCH_WINDOW_512_CHIPS = 15, TCPE_SEARCH_WINDOW_544_CHIPS = 16,
  TCPE_SEARCH_WINDOW_576_CHIPS = 17,
  TCPE_SEARCH_WINDOW_608_CHIPS = 18, TCPE_SEARCH_WINDOW_640_CHIPS = 19,
  TCPE_SEARCH_WINDOW_672_CHIPS = 20,
  TCPE_SEARCH_WINDOW_704_CHIPS = 21, TCPE_SEARCH_WINDOW_736_CHIPS = 22,
  TCPE_SEARCH_WINDOW_768_CHIPS = 23,
  TCPE_SEARCH_WINDOW_800_CHIPS = 24, TCPE_SEARCH_WINDOW_832_CHIPS = 25,
  TCPE_SEARCH_WINDOW_864_CHIPS = 26,
  TCPE_SEARCH_WINDOW_896_CHIPS = 27, TCPE_SEARCH_WINDOW_928_CHIPS = 28,
  TCPE_SEARCH_WINDOW_960_CHIPS = 29,
  TCPE_SEARCH_WINDOW_992_CHIPS = 30, TCPE_SEARCH_WINDOW_1024_CHIPS = 31,
  TCPE_SEARCH_WINDOW_1056_CHIPS = 32,
  TCPE_SEARCH_WINDOW_1088_CHIPS = 33, TCPE_SEARCH_WINDOW_1120_CHIPS =

34, TCPE_SEARCH_WINDOW_1152_CHIPS = 35,
TCPE_SEARCH_WINDOW_1184_CHIPS = 36, TCPE_SEARCH_WINDOW_1216_CHIPS = 37, TCPE_SEARCH_WINDOW_1248_CHIPS = 38,
TCPE_SEARCH_WINDOW_1280_CHIPS = 39, TCPE_SEARCH_WINDOW_1312_CHIPS = 40, TCPE_SEARCH_WINDOW_1344_CHIPS = 41,
TCPE_SEARCH_WINDOW_1376_CHIPS = 42, TCPE_SEARCH_WINDOW_1408_CHIPS = 43, TCPE_SEARCH_WINDOW_1440_CHIPS = 44,
TCPE_SEARCH_WINDOW_1472_CHIPS = 45, TCPE_SEARCH_WINDOW_1504_CHIPS = 46, TCPE_SEARCH_WINDOW_1536_CHIPS = 47,
TCPE_SEARCH_WINDOW_1568_CHIPS = 48, TCPE_SEARCH_WINDOW_1600_CHIPS = 49, TCPE_SEARCH_WINDOW_1632_CHIPS = 50,
TCPE_SEARCH_WINDOW_1664_CHIPS = 51, TCPE_SEARCH_WINDOW_1696_CHIPS = 52, TCPE_SEARCH_WINDOW_1728_CHIPS = 53,
TCPE_SEARCH_WINDOW_1760_CHIPS = 54, TCPE_SEARCH_WINDOW_1792_CHIPS = 55, TCPE_SEARCH_WINDOW_1824_CHIPS = 56,
TCPE_SEARCH_WINDOW_1856_CHIPS = 57, TCPE_SEARCH_WINDOW_1888_CHIPS = 58, TCPE_SEARCH_WINDOW_1920_CHIPS = 59,
TCPE_SEARCH_WINDOW_1952_CHIPS = 60, TCPE_SEARCH_WINDOW_1984_CHIPS = 61, TCPE_SEARCH_WINDOW_2016_CHIPS = 62,
TCPE_SEARCH_WINDOW_2048_CHIPS = 63 }

- enum {
  TCPE_CMBL_512_CHIPS = 2, TCPE_CMBL_768_CHIPS = 3, TCPE_CMBL_1024_CHIPS = 4,
  TCPE_CMBL_1280_CHIPS = 5, TCPE_CMBL_1536_CHIPS = 6, TCPE_CMBL_1792_CHIPS = 7,
  TCPE_CMBL_2048_CHIPS = 8, TCPE_CMBL_2304_CHIPS = 9, TCPE_CMBL_2560_CHIPS = 10 }
- enum { TCPE_ACCUM_NOT_INITIALIZED = 0, TCPE_COHER_ACCUM_INITIALIZED = 1, TCPE_NON_COHER_ACCUM_INITIALIZED = 2 }
- enum {
  TCPE_CS_SKIP = 0, TCPE_CS_COHER_ACCUM = 1, TCPE_CS_TO_NCA_NO_WEIGHT = 3,
  TCPE_CS_TO_NCA_WEIGHT0 = 4, TCPE_CS_TO_NCA_WEIGHT1 = 5, TCPE_CS_TO_NC↩
  A_WEIGHT2 = 6,
  TCPE_CS_TO_NCA_WEIGHT3 = 7 }
- enum { TCPE_EP_PLUS_1 = 0, TCPE_EP_MINUS_1 = 1 }
- #define TCPE_MODE_PATH_SEARCH 0
- #define TCPE_MODE_RACH 1
- #define TCPE_FC_DISABLE 0
- #define TCPE_FC_ENABLE 1
- #define TCPE_LAST_USER 1
- #define TCPE_CS(slot, strategy) ((strategy << slot*3))
- #define TCPE_SSO_DONT_ADD_4096_CG 0
- #define TCPE_SSO_ADD_4096_CG 1
- #define TCPE_SST_SHORT 0
- #define TCPE_SST_LONG 1
- #define TCPE_EP(slot, value) ((value << slot))
- #define TCPE_EP_IQ_MAP_I 0
- #define TCPE_EP_IQ_MAP_Q 0x8000
- #define TCPE_FC_STEP_SIZE(num_samples) ((num_samples / 16 - 1))

3.5.12.3.2 Data Structure Documentation

3.5.12.3.2.1 struct maple_tcpe_ucb_t

TCPE User Configuration Buffer Entry Structure.

Data Fields

- uint8_t nchip
- uint16_t pilot_id: 4
- uint16_t mode: 1
- uint16_t fc_en: 1
- uint16_t sw: 6
- uint16_t cmbl: 4
- uint32_t last: 1
- uint32_t acc_scl: 5
- uint32_t nca_scl: 5
- uint32_t ca_scl: 5
- uint32_t acc_init: 2
- uint32_t chip_offset: 13
- uint32_t combining_strategy
- uint32_t sso: 1
- uint32_t sst: 1
- uint32_t ssn: 24
- short fc_init_real
- short fc_init_imag
- short fc_step_real
- short fc_step_imag
- uint16_t ep
- uint8_t ccn
- uint8_t fc_step_size
- uint8_t nca_weight0
- uint8_t nca_weight1
- uint8_t nca_weight2
- uint8_t nca_weight3

3.5.12.3.2.1.1 Field Documentation

3.5.12.3.2.1.2 uint8_t maple_tcpe_ucb_t::nchip

The chip number of the first chip of the user scrambling code in the job.

3.5.12.3.2.1.3 uint16_t maple_tcpe_ucb_t::pilot_id

Path-Search Mode: Number of pilots, RACH Mode: Signature ID.

3.5.12.3.2.1.4 uint16_t maple_tcpe_ucb_t::mode

0 Path-Search, 1 RACH.

3.5.12.3.2.1.5 uint16_t maple_tcpe_ucb_t::fc_en

Frequency Correction Enable.

3.5.12.3.2.1.6 uint16_t maple_tcpe_ucb_t::sw

Search Window.

3.5.12.3.2.1.7 uint16_t maple_tcpe_ucb_t::cmbl

Combining length.

3.5.12.3.2.1.8 uint32_t maple_tcpe_ucb_t::last

Last User indication.

3.5.12.3.2.1.9 uint32_t maple_tcpe_ucb_t::acc_scl

Accumulator initialization Scale.

3.5.12.3.2.1.10 uint32_t maple_tcpe_ucb_t::nca_scl

Non-Coherent Accumulator Scale.

3.5.12.3.2.1.11 uint32_t maple_tcpe_ucb_t::ca_scl

Coherent Accumulator Scale.

3.5.12.3.2.1.12 uint32_t maple_tcpe_ucb_t::acc_init

Accumulator Initialization type.

3.5.12.3.2.1.13 uint32_t maple_tcpe_ucb_t::chip_offset

Chip Offset.

3.5.12.3.2.1.14 uint32_t maple_tcpe_ucb_t::combining_strategy

Combining Strategy for sub-slot.

3.5.12.3.2.1.15 uint32_t maple_tcpe_ucb_t::sso

Scrambling Sequence Offset.

3.5.12.3.2.1.16 uint32_t maple_tcpe_ucb_t::sst

Scrambling Sequence Type.

3.5.12.3.2.1.17 uint32_t maple_tcpe_ucb_t::ssn

Scrambling Sequence Number.

3.5.12.3.2.1.18 short maple_tcpe_ucb_t::fc_init_real

Frequency Correction Initialization Real part.

3.5.12.3.2.1.19 short maple_tcpe_ucb_t::fc_init_imag

Frequency Correction Initialization Imaginary part.

3.5.12.3.2.1.20 short maple_tcpe_ucb_t::fc_step_real

Frequency Correction Step Real part.

3.5.12.3.2.1.21 short maple_tcpe_ucb_t::fc_step_imag

Frequency Correction Step Imaginary part.

3.5.12.3.2.1.22 uint16_t maple_tcpe_ucb_t::ep

External Pilot.

3.5.12.3.2.1.23 uint8_t maple_tcpe_ucb_t::ccn

Channelization Code Number.

3.5.12.3.2.1.24 uint8_t maple_tcpe_ucb_t::fc_step_size

Frequency Correction Step Size (in multiples of 16).

3.5.12.3.2.1.25 uint8_t maple_tcpe_ucb_t::nca_weight0

Non-Coherent Combining Weight of Section 0.

3.5.12.3.2.1.26 uint8_t maple_tcpe_ucb_t::nca_weight1

Non-Coherent Combining Weight of Section 0.

3.5.12.3.2.1.27 uint8_t maple_tcpe_ucb_t::nca_weight2

Non-Coherent Combining Weight of Section 0.

3.5.12.3.2.1.28 uint8_t maple_tcpe_ucb_t::nca_weight3

Non-Coherent Combining Weight of Section 0.

3.5.12.3.2.2 struct maple_tcpe_uiobc_t

TCPE User Initialization/Output Buffers Configuration Structure.

Data Fields

- uint16_t u_out_size
- uint16_t u_acc_size
- uint32_t u_out_addr
- uint32_t out_steer: 4
- uint32_t u_acc_addr
- uint32_t acc_steer: 4

3.5.12.3.2.2.1 Field Documentation

3.5.12.3.2.2.2 uint32_t maple_tcpe_uiobc_t::u_out_addr

User i Output Buffer Address.

3.5.12.3.2.2.3 uint32_t maple_tcpe_uiobc_t::out_steer

Pointer Steering bits #0.

3.5.12.3.2.2.4 uint32_t maple_tcpe_uiobc_t::u_acc_addr

User i Initialization Accumulator Buffer Address.

3.5.12.3.2.2.5 uint32_t maple_tcpe_uiobc_t::acc_steer

Pointer Steering bits #0.

3.5.12.3.2.2.6 uint16_t maple_tcpe_uiobc_t::u_out_size

User i Output Buffer Size.

3.5.12.3.2.2.7 uint16_t maple_tcpe_uiobc_t::u_acc_size

User i Accumulator Buffer Size.

3.5.12.3.2.3 struct maple_tcpe_job_t

TCPE Job Descriptor.

This structure should be passed to the LLD on the cop_job_handle.device_specific field in order for the LLD to build the necessary BD

Data Fields

- uint32_t antenna_buff_size
- uint32_t antenna_fetch_off
- uint32_t antenna0_base_addr
- uint32_t antenna1_base_addr
- maple_tcpe_ucb_t * user_conf_buffer
- maple_tcpe_uiobc_t * user_init_out_buff_conf
- uint8_t bd_index
- uint32_t axi_priority:2
- uint32_t num_users:8
- uint32_t ant_fetch_size:11

3.5.12.3.2.3.1 Field Documentation

3.5.12.3.2.3.2 uint32_t maple_tcpe_job_t::axi_priority

AXI Priority.

3.5.12.3.2.3.3 uint32_t maple_tcpe_job_t::num_users

number of users/signatures that should be searched in the given antenna data

3.5.12.3.2.3.4 uint32_t maple_tcpe_job_t::ant_fetch_size

The amount of 32 bytes chunks to read from each buffer of the antenna diversity data.

3.5.12.3.2.3.5 uint32_t maple_tcpe_job_t::antenna_buff_size

Antenna Data Buffer Size.

3.5.12.3.2.3.6 uint32_t maple_tcpe_job_t::antenna_fetch_off

Antenna Data Buffer Fetch Offset.

3.5.12.3.2.3.7 uint32_t maple_tcpe_job_t::antenna0_base_addr

Antenna Diversity 0 Data Buffer Base Address.

3.5.12.3.2.3.8 uint32_t maple_tcpe_job_t::antenna1_base_addr

Antenna Diversity 1 Data Buffer Base Address.

3.5.12.3.2.3.9 maple_tcpe_ucb_t* maple_tcpe_job_t::user_conf_buffer

User Configuration Buffer Address.

3.5.12.3.2.3.10 maple_tcpe_uiobc_t* maple_tcpe_job_t::user_init_out_buff_conf

User Initialization/Output Buffers Configuration Address.

3.5.12.3.2.3.11 uint8_t maple_tcpe_job_t::bd_index

Job's index in the BD ring.

Will be assigned by the driver

3.5.12.3.3 Macro Definition Documentation

3.5.12.3.3.1 #define TCPE_MAX_NUM_BD_FOR_DISPACTH 16

Maximal number of TCPE BD per dispatch.

Defines the maximal number of BD that can be dispatched with a single call to osCopChannelDispatch(). The larger the number the higher the stack consumption is in the driver.

Warning

Users may change this value, however it requires recompiling the drivers

3.5.12.3.3.2 #define TCPE_BD_MANUAL_ACTIVATION 0x80000000

Prevent the driver from dispatching the job, MAPLE will wait until the user calls a Channel Ctrl Command to activate the job.

3.5.12.3.3.3 #define TCPE_BD_INT_EN 0x10000000

MAPLE issues an interrupt interrupt at the end of job.

3.5.12.3.3.4 #define TCPE_MODE_PATH_SEARCH 0

TCPE Reference SIgnal generation mode.

should be set in mode.Path Search.

3.5.12.3.3.5 #define TCPE_MODE_RACH 1

RACH.

3.5.12.3.3.6 #define TCPE_FC_DISABLE 0

Frequency Correction Enable.

should be set in fc_enFrequency correction is disabled.

3.5.12.3.3.7 #define TCPE_FC_ENABLE 1

Frequency correction is enabled.

3.5.12.3.3.8 #define TCPE_LAST_USER 1

Last User indication.

should be set in last

3.5.12.3.3.9 #define TCPE_CS(*slot, strategy*) ((strategy << slot∗3))

Accumulator Initialization type.

should be set in cs

3.5.12.3.3.10 #define TCPE_SSO_DONT_ADD_4096_CG 0

Scrambling Sequence Offset.

should be set in ssofor DPCCH code generation

3.5.12.3.3.11 #define TCPE_SSO_ADD_4096_CG 1

for RACH message code generation

3.5.12.3.3.12 #define TCPE_SST_SHORT 0

Scrambling Sequence Type.

For RACH mode (MODE=1), this field must be set to 1. should be set in sstShort.

3.5.12.3.3.13 #define TCPE_SST_LONG 1

Long.

3.5.12.3.3.14 #define TCPE_EP(*slot, value*) ((value << slot))

External Pilot.

External Pilots for sub-slot <x>, that is, chips x∗256: (x+1)∗256-1 (x=0,1,...,9) in the slot. This is field is valid only for Path-Search Mode (MODE=0) and PILOT_ID=0

3.5.12.3.3.15 #define TCPE_EP_IQ_MAP_I 0

Map external pilots to I branch.

3.5.12.3.3.16 #define TCPE_EP_IQ_MAP_Q 0x8000

Map external pilots to Q branch.

3.5.12.3.3.17 #define TCPE_FC_STEP_SIZE(*num_samples*) ((num_samples / 16 - 1))

Frequency Correction Step Size.

should be set in fc_step_size

3.5.12.3.4 Enumeration Type Documentation

3.5.12.3.4.1 anonymous enum

MBus Priority - used in maple_tcpe_job_t.

Valid only if the [MB_PR_SCH] of the MMC0P parameter is not 0. should be set in job.axi_priority.

Enumerator

TCPE_MB_PRIORITY_0	The MBus accesses related to that BD are initiated with priority 0.
TCPE_MB_PRIORITY_1	The MBus accesses related to that BD are initiated with priority 1.
TCPE_MB_PRIORITY_2	The MBus accesses related to that BD are initiated with priority 2.
TCPE_MB_PRIORITY_3	The MBus accesses related to that BD are initiated with priority 3.

3.5.12.3.4.2 anonymous enum

Number of pilots (IN Path-Search Mode) should be set in pilot_id.

Enumerator

TCPE_EXTERNAL_PILOTS Use External pilots.
TCPE_3_PILOTS 3 pilots in slot
TCPE_4_PILOTS 4 pilots in slot
TCPE_5_PILOTS 5 pilots in slot
TCPE_6_PILOTS 6 pilots in slot
TCPE_7_PILOTS 7 pilots in slot
TCPE_8_PILOTS 8 pilots in slot

3.5.12.3.4.3 anonymous enum

Signature ID (in RACH Mode).

should be set in pilot_id

Enumerator

TCPE_SIGNATURE_0 Use Signature #0.
TCPE_SIGNATURE_1 Use Signature #1.
TCPE_SIGNATURE_2 Use Signature #2.
TCPE_SIGNATURE_3 Use Signature #3.
TCPE_SIGNATURE_4 Use Signature #4.
TCPE_SIGNATURE_5 Use Signature #5.
TCPE_SIGNATURE_6 Use Signature #6.
TCPE_SIGNATURE_7 Use Signature #7.
TCPE_SIGNATURE_8 Use Signature #8.
TCPE_SIGNATURE_9 Use Signature #9.
TCPE_SIGNATURE_10 Use Signature #10.
TCPE_SIGNATURE_11 Use Signature #11.
TCPE_SIGNATURE_12 Use Signature #12.
TCPE_SIGNATURE_13 Use Signature #13.
TCPE_SIGNATURE_14 Use Signature #14.
TCPE_SIGNATURE_15 Use Signature #15.

3.5.12.3.4.4 anonymous enum

Search Window.

Describe the search window. should be set in sw

Enumerator

TCPE_SEARCH_WINDOW_32_CHIPS 32 chips
TCPE_SEARCH_WINDOW_64_CHIPS 64 chips
TCPE_SEARCH_WINDOW_96_CHIPS 96 chips
TCPE_SEARCH_WINDOW_128_CHIPS 128 chips
TCPE_SEARCH_WINDOW_160_CHIPS 160 chips
TCPE_SEARCH_WINDOW_192_CHIPS 192 chips

TCPE_SEARCH_WINDOW_224_CHIPS	224 chips
TCPE_SEARCH_WINDOW_256_CHIPS	256 chips
TCPE_SEARCH_WINDOW_288_CHIPS	288 chips
TCPE_SEARCH_WINDOW_320_CHIPS	320 chips
TCPE_SEARCH_WINDOW_352_CHIPS	352 chips
TCPE_SEARCH_WINDOW_384_CHIPS	384 chips
TCPE_SEARCH_WINDOW_416_CHIPS	416 chips
TCPE_SEARCH_WINDOW_448_CHIPS	448 chips
TCPE_SEARCH_WINDOW_480_CHIPS	480 chips
TCPE_SEARCH_WINDOW_512_CHIPS	512 chips
TCPE_SEARCH_WINDOW_544_CHIPS	544 chips
TCPE_SEARCH_WINDOW_576_CHIPS	576 chips
TCPE_SEARCH_WINDOW_608_CHIPS	608 chips
TCPE_SEARCH_WINDOW_640_CHIPS	640 chips
TCPE_SEARCH_WINDOW_672_CHIPS	672 chips
TCPE_SEARCH_WINDOW_704_CHIPS	704 chips
TCPE_SEARCH_WINDOW_736_CHIPS	736 chips
TCPE_SEARCH_WINDOW_768_CHIPS	768 chips
TCPE_SEARCH_WINDOW_800_CHIPS	800 chips
TCPE_SEARCH_WINDOW_832_CHIPS	832 chips
TCPE_SEARCH_WINDOW_864_CHIPS	864 chips
TCPE_SEARCH_WINDOW_896_CHIPS	896 chips
TCPE_SEARCH_WINDOW_928_CHIPS	928 chips
TCPE_SEARCH_WINDOW_960_CHIPS	960 chips
TCPE_SEARCH_WINDOW_992_CHIPS	992 chips
TCPE_SEARCH_WINDOW_1024_CHIPS	1024 chips
TCPE_SEARCH_WINDOW_1056_CHIPS	1056 chips
TCPE_SEARCH_WINDOW_1088_CHIPS	1088 chips
TCPE_SEARCH_WINDOW_1120_CHIPS	1120 chips
TCPE_SEARCH_WINDOW_1152_CHIPS	1152 chips
TCPE_SEARCH_WINDOW_1184_CHIPS	1184 chips
TCPE_SEARCH_WINDOW_1216_CHIPS	1216 chips
TCPE_SEARCH_WINDOW_1248_CHIPS	1248 chips
TCPE_SEARCH_WINDOW_1280_CHIPS	1280 chips
TCPE_SEARCH_WINDOW_1312_CHIPS	1312 chips
TCPE_SEARCH_WINDOW_1344_CHIPS	1344 chips
TCPE_SEARCH_WINDOW_1376_CHIPS	1376 chips
TCPE_SEARCH_WINDOW_1408_CHIPS	1408 chips
TCPE_SEARCH_WINDOW_1440_CHIPS	1440 chips
TCPE_SEARCH_WINDOW_1472_CHIPS	1472 chips
TCPE_SEARCH_WINDOW_1504_CHIPS	1504 chips
TCPE_SEARCH_WINDOW_1536_CHIPS	1536 chips
TCPE_SEARCH_WINDOW_1568_CHIPS	1568 chips
TCPE_SEARCH_WINDOW_1600_CHIPS	1600 chips
TCPE_SEARCH_WINDOW_1632_CHIPS	1632 chips

TCPE_SEARCH_WINDOW_1664_CHIPS 1664 chips
TCPE_SEARCH_WINDOW_1696_CHIPS 1696 chips
TCPE_SEARCH_WINDOW_1728_CHIPS 1728 chips
TCPE_SEARCH_WINDOW_1760_CHIPS 1760 chips
TCPE_SEARCH_WINDOW_1792_CHIPS 1792 chips
TCPE_SEARCH_WINDOW_1824_CHIPS 1824 chips
TCPE_SEARCH_WINDOW_1856_CHIPS 1856 chips
TCPE_SEARCH_WINDOW_1888_CHIPS 1888 chips
TCPE_SEARCH_WINDOW_1920_CHIPS 1920 chips
TCPE_SEARCH_WINDOW_1952_CHIPS 1952 chips
TCPE_SEARCH_WINDOW_1984_CHIPS 1984 chips
TCPE_SEARCH_WINDOW_2016_CHIPS 2016 chips
TCPE_SEARCH_WINDOW_2048_CHIPS 2048 chips

3.5.12.3.4.5 anonymous enum

Combining length.

should be set in cmbl

Enumerator

TCPE_CMBL_512_CHIPS 512 chips
TCPE_CMBL_768_CHIPS 768 chips
TCPE_CMBL_1024_CHIPS 1024 chips
TCPE_CMBL_1280_CHIPS 1280 chips
TCPE_CMBL_1536_CHIPS 1536 chips
TCPE_CMBL_1792_CHIPS 1792 chips
TCPE_CMBL_2048_CHIPS 2048 chips
TCPE_CMBL_2304_CHIPS 2304 chips
TCPE_CMBL_2560_CHIPS 2560 chips

3.5.12.3.4.6 anonymous enum

Accumulator Initialization type.

should be set in acc_init

Enumerator

TCPE_ACCUM_NOT_INITIALIZED Accumulators are not initialized (start from zero).
TCPE_COHER_ACCUM_INITIALIZED Coherent Accumulator initialized.
TCPE_NON_COHER_ACCUM_INITIALIZED Non-Coherent Accumulator initialized .

3.5.12.3.4.7 anonymous enum

Accumulator Initialization type.

should be set in cs

Enumerator

TCPE_CS_SKIP Skip this sub-slot.
TCPE_CS_COHER_ACCUM Coherent Accumulation.
TCPE_CS_TO_NCA_NO_WEIGHT ove Accumulator to Non-Coherent Accumulator after this sub-slot and combine without weighting.
TCPE_CS_TO_NCA_WEIGHT0 Move Accumulator to Non-Coherent Accumulator after this sub-slot and combine using NCA_WEIGHT0.
TCPE_CS_TO_NCA_WEIGHT1 Move Accumulator to Non-Coherent Accumulator after this sub-slot and combine using NCA_WEIGHT1.
TCPE_CS_TO_NCA_WEIGHT2 Move Accumulator to Non-Coherent Accumulator after this sub-slot and combine using NCA_WEIGHT2.
TCPE_CS_TO_NCA_WEIGHT3 Move Accumulator to Non-Coherent Accumulator after this sub-slot and combine using NCA_WEIGHT3.

3.5.12.3.4.8 anonymous enum

External Pilots for sub-slot.

should be set with or operation in ep

Enumerator

TCPE_EP_PLUS_1 pilot value +1.
TCPE_EP_MINUS_1 pilot value -1.

3.5.13 Maple Trace Events API

3.5.13.1 Overview

```
MAPLE Trace Events API
```

Modules

- Maple Trace Events Runtime

3.5.13.2 Maple Trace Events Runtime

3.5.13.2.1 Overview

```
MAPLE Trace Event Runtime API
```

Data Structures

- struct maple_trace_event_t
- struct maple_pdsch_trace_event_t

- struct maple_trace_read_buffers_param_t
- struct maple_pe_load_param_t

Enumerations

- enum maple_trace_event_enable_t { MAPLE_EN_TRACE_EVENT_PDSCH_STARTED_FINAL_PROCESSING_OF_AN_OFDM_SYMBOL = (0x1 << 8), MAPLE_EN_TRACE_EVENT_PDSCH_DETERMINED_NUMBER_OF_CONTROL_DATA_AND_PAD_OFDM_SYMBOLS = (0x1 << 9), MAPLE_EN_TRACE_EVENT_PDSCH_STARTED_PARSING_USER_HEADERS = (0x1 << 10), MAPLE_EN_TRACE_EVENT_PDSCH_IS_PREFETCHING_OFDM_SYMBOL_INFORMATION = (0x1 << 11), MAPLE_EN_TRACE_EVENT_PDSCH_BD_DONE = (0x1 << 13) }
- enum maple_trace_id_t { MAPLE_TRACE_ID_PDSCH_STARTED_FINAL_PROCESSING_OF_AN_OFDM_SYMBOL = 0x20, MAPLE_TRACE_ID_PDSCH_DETERMINED_NUMBER_OF_CONTROL_DATA_AND_PAD_OFDM_SYMBOLS = 0x21, MAPLE_TRACE_ID_PDSCH_STARTED_PARSING_USER_HEADERS = 0x22, MAPLE_TRACE_ID_PDSCH_IS_PREFETCHING_OFDM_SYMBOL_INFORMATION = 0x23, MAPLE_TRACE_ID_PDSCH_BD_DONE = 0x25 }
- enum maple_pe_load_type_t

3.5.13.2.2 Data Structure Documentation

3.5.13.2.2.1 struct maple_trace_event_t

MAPLE-B3 Trace event structure - use to read trace events from RISC 0/1 trace buffer.

Data Fields

- uint32_t trc_id: 8
- uint32_t __pad0__: 8
- uint32_t time: 24

3.5.13.2.2.1.1 Field Documentation

3.5.13.2.2.1.2 uint32_t maple_trace_event_t::trc_id

The content of this field is determined according to the event index.

3.5.13.2.2.1.3 uint32_t maple_trace_event_t::__pad0__

Reserved.

3.5.13.2.2.1.4 uint32_t maple_trace_event_t::time

Relative time stamp of the event.

3.5.13.2.2.2 struct maple_pdsch_trace_event_t

MAPLE-B3 PDSCH-EDF Trace event structure - This is the structure of a PDSCH trace event.

Data Fields

- uint8_t trace_id
- uint8_t task_id
- uint8_t symbol_ix
- uint8_t status
- uint32_t __pad0__: 8
- uint32_t time: 24

3.5.13.2.2.2.1 Field Documentation

3.5.13.2.2.2.2 uint8_t maple_pdsch_trace_event_t::trace_id

The content of this field is determined according to the event index.

3.5.13.2.2.2.3 uint8_t maple_pdsch_trace_event_t::task_id

The Task ID field of the BD which this job was submitted from.

3.5.13.2.2.2.4 uint8_t maple_pdsch_trace_event_t::symbol_ix

Symbol Index - This field specifies the index of the OFDM symbol that the MAPLE-B3 just started processing if applicable.

Valid values: 0...503.

3.5.13.2.2.2.5 uint8_t maple_pdsch_trace_event_t::status

This field contains the 4 LS bits of the OSFP_STATUS field when the trace is generated by the OSFP process.

It it N/A in other cases.

3.5.13.2.2.2.6 uint32_t maple_pdsch_trace_event_t::__pad0__

Reserved.

3.5.13.2.2.2.7 uint32_t maple_pdsch_trace_event_t::time

Relative time stamp of the event.

3.5.13.2.2.3 struct maple_trace_read_buffers_param_t

MAPLE-B3 PDSCH-EDF Trace Buffer structure - used to read trace events from MAPLE trace buffer.

Data Fields

- maple_trace_event_t * event_queue
- uint16_t offset

- uint16_t num_of_events

3.5.13.2.2.3.1 Field Documentation

3.5.13.2.2.3.2 maple_trace_event_t* maple_trace_read_buffers_param_t::event_queue

Pointer to where the driver will copy the trace buffer.

3.5.13.2.2.3.3 uint16_t maple_trace_read_buffers_param_t::offset

Offset from the oldest event to begin coping.

3.5.13.2.2.3.4 uint16_t maple_trace_read_buffers_param_t::num_of_events

Indicates how many events to copy from the offset in the buffer.

Valid values = (1...512) .

3.5.13.2.2.4 struct maple_pe_load_param_t

MAPLE-B3 PE Load Accumulators extractor.

should be passed when calling for a PE's Ctrl Command <XXPE>_CMD_GET_PE_ACCUMULATOR

Data Fields

- uint32_t maple_clk_counter
- uint32_t pe_load

3.5.13.2.2.4.1 Field Documentation

3.5.13.2.2.4.2 uint32_t maple_pe_load_param_t::maple_clk_counter

Stores the value of the MAPLE clock when calling the Ctrl Command.

3.5.13.2.2.4.3 uint32_t maple_pe_load_param_t::pe_load

Stores the value of the PE accumulators Ctrl Command.

3.5.13.2.3 Enumeration Type Documentation

3.5.13.2.3.1 enum maple_trace_event_enable_t

MAPLE-B3 MTRCECP[TRCx_EN] MAPLE Trace Enable Configuration Parameter.

Enumerator

MAPLE_EN_TRACE_EVENT_PDSCH_STARTED_FINAL_PROCESSING_OF_AN_OFDM_SYMBOL
MAPLE PDSCH-EDF started final processing of an OFDM symbol (Trace ID 0x20).

MAPLE_EN_TRACE_EVENT_PDSCH_DETERMINED_NUMBER_OF_CONTROL_DATA_AND_PAD_
MAPLE PDSCH-EDF has determined the number of Control data and pad OFDM symbols

(Trace ID 0x21).

MAPLE_EN_TRACE_EVENT_PDSCH_STARTED_PARSING_USER_HEADERS MAPLE PDSCH-EDF has started parsing user headers (Trace ID 0x22).

MAPLE_EN_TRACE_EVENT_PDSCH_IS_PREFETCHING_OFDM_SYMBOL_INFORMATION MAPLE PDSCH-EDF is prefetching an OFDM symbol information (Trace ID 0x23).

MAPLE_EN_TRACE_EVENT_PDSCH_BD_DONE MAPLE PDSCH-EDF finished a BD (Trace ID 0x25).

3.5.13.2.3.2 enum maple_trace_id_t

MAPLE-B3 Trace Events ID, used for extracting the trace event id.

Enumerator

MAPLE_TRACE_ID_PDSCH_STARTED_FINAL_PROCESSING_OF_AN_OFDM_SYMBOL MAPLE PDSCH-EDF started final processing of an OFDM symbol (Trace ID 0x20).

MAPLE_TRACE_ID_PDSCH_DETERMINED_NUMBER_OF_CONTROL_DATA_AND_PAD_OFDM_SY MAPLE PDSCH-EDF has determined the number of Control data and pad OFDM symbols (Trace ID 0x21).

MAPLE_TRACE_ID_PDSCH_STARTED_PARSING_USER_HEADERS MAPLE PDSCH-EDF has started parsing user headers (Trace ID 0x22).

MAPLE_TRACE_ID_PDSCH_IS_PREFETCHING_OFDM_SYMBOL_INFORMATION MAPLE PDSCH-EDF is prefetching an OFDM symbol information (Trace ID 0x23).

MAPLE_TRACE_ID_PDSCH_BD_DONE MAPLE PDSCH-EDF finished a BD (Trace ID 0x25).

3.5.13.2.3.3 enum maple_pe_load_type_t

MAPLE-B3 PE Load Accumulators index.

3.6 OCeaN DMA Module API

3.6.1 Overview

Modules

- OCeaN DMA Module API Initialization
- OCeaN DMA Module API Runtime

3.6.2 OCeaN DMA Module API Initialization

3.6.2.1 Overview

Data Structures

- struct ocn_dma_atmu_config_t
- struct ocn_dma_init_params_t

Enumerations

- enum ocn_dma_id_t { OCN_DMA_ID0 = 0, OCN_DMA_ID1 = 1 }
- enum ocn_dma_win_sz_t {
  OCN_DMA_WIN_LAST = 0x0, OCN_DMA_WIN_4K = 0xB, OCN_DMA_WIN_8K = 0xC,
  OCN_DMA_WIN_16K = 0xD, OCN_DMA_WIN_32K = 0xE, OCN_DMA_WIN_64K = 0xF,
  OCN_DMA_WIN_128K = 0x10, OCN_DMA_WIN_256K = 0x11, OCN_DMA_WIN_512K = 0x12,
  OCN_DMA_WIN_1M = 0x13, OCN_DMA_WIN_2M = 0x14, OCN_DMA_WIN_4M = 0x15,
  OCN_DMA_WIN_8M = 0x16, OCN_DMA_WIN_16M = 0x17, OCN_DMA_WIN_32M = 0x18,
  OCN_DMA_WIN_64M = 0x19, OCN_DMA_WIN_128M = 0x1A, OCN_DMA_WIN_256M = 0x1B,
  OCN_DMA_WIN_512M = 0x1C, OCN_DMA_WIN_1G = 0x1D, OCN_DMA_WIN_2G = 0x1E,
  OCN_DMA_WIN_4G = 0x1F, OCN_DMA_WIN_8G = 0x20, OCN_DMA_WIN_16G = 0x21,
  OCN_DMA_WIN_32G = 0x22, OCN_DMA_WIN_64G = 0x23 }
- enum ocn_dma_interface_t {
  OCN_DMA_INTERFACE_PEX = 0x00100000, OCN_DMA_INTERFACE_RIO0 = 0x00↩
  C00000, OCN_DMA_INTERFACE_RIO1 = 0x00D00000,
  OCN_DMA_INTERFACE_LOCAL0 = 0x00B00000, OCN_DMA_INTERFACE_LOCAL1 = 0x00F00000 }

Functions

- os_status ocnDmaInitialize (ocn_dma_init_params_t *ocn_dma_config)

3.6.2.2 Data Structure Documentation

3.6.2.2.1 struct ocn_dma_atmu_config_t

OCeaN DMA ATMU window configuration.

Data Fields

- uint64_t win_base_addr
- ocn_dma_win_sz_t win_size
- ocn_dma_interface_t win_interface

3.6.2.2.1.1 Field Documentation

3.6.2.2.1.1.1 uint64_t ocn_dma_atmu_config_t::win_base_addr

Base address for window on OCeaN space.

Only 36 bit is use

3.6.2.2.1.1.2 ocn_dma_win_sz_t ocn_dma_atmu_config_t::win_size

Window size.

3.6.2.2.1.1.3 ocn_dma_interface_t ocn_dma_atmu_config_t::win_interface

Interface towards which window generates accesses.

3.6.2.2.2 struct ocn_dma_init_params_t

OCeaN DMA initialization structure.

Warning

If intending to generate sRIO maintanence acceses, one dma_atmu_config MUST be set according to: SRIO_MAINTANENCE_ATMU_ADDR and SRIO_MAINTANENCE_ATMU_SIZE

Data Fields

- ocn_dma_id_t dma_id
- os_hwi_priority irq_priority
- uint16_t dma_channel_enable [NUMBER_OF_OCN_DMA_CH]
- void * reserved

3.6.2.2.2.1 Field Documentation

3.6.2.2.2.1.1 ocn_dma_id_t ocn_dma_init_params_t::dma_id

OCeaN DMA device index.

3.6.2.2.2.1.2 os_hwi_priority ocn_dma_init_params_t::irq_priority

Interrupt priority.

3.6.2.2.2.1.3 uint16_t ocn_dma_init_params_t::dma_channel_enable[NUMBER_OF_OCN_DMA_CH]

On which core to enable the channel.

3.6.2.2.2.1.4 void* ocn_dma_init_params_t::reserved

reserved for future uses

3.6.2.3 Enumeration Type Documentation

3.6.2.3.1 enum ocn_dma_id_t

OCeaN DMA controller device ID.

Enumerator

OCN_DMA_ID0 OCeaN DMA 0.
OCN_DMA_ID1 OCeaN DMA 1.

3.6.2.3.2 enum ocn_dma_win_sz_t

OCeaN DMA ATMU window size.

Enumerator

OCN_DMA_WIN_LAST Software indicator for end of ATMU array.
OCN_DMA_WIN_4K ATMU window, size is 4K.
OCN_DMA_WIN_8K ATMU window, size is 8K.
OCN_DMA_WIN_16K ATMU window, size is 16K.
OCN_DMA_WIN_32K ATMU window, size is 32K.
OCN_DMA_WIN_64K ATMU window, size is 64K.
OCN_DMA_WIN_128K ATMU window, size is 128K.
OCN_DMA_WIN_256K ATMU window, size is 256K.
OCN_DMA_WIN_512K ATMU window, size is 512K.
OCN_DMA_WIN_1M ATMU window, size is 1M.
OCN_DMA_WIN_2M ATMU window, size is 2M.
OCN_DMA_WIN_4M ATMU window, size is 4M.
OCN_DMA_WIN_8M ATMU window, size is 8M.
OCN_DMA_WIN_16M ATMU window, size is 16M.
OCN_DMA_WIN_32M ATMU window, size is 32M.
OCN_DMA_WIN_64M ATMU window, size is 64M.
OCN_DMA_WIN_128M ATMU window, size is 128M.
OCN_DMA_WIN_256M ATMU window, size is 256M.
OCN_DMA_WIN_512M ATMU window, size is 512M.
OCN_DMA_WIN_1G ATMU window, size is 1G.
OCN_DMA_WIN_2G ATMU window, size is 2G.
OCN_DMA_WIN_4G ATMU window, size is 4G.
OCN_DMA_WIN_8G ATMU window, size is 8G.
OCN_DMA_WIN_16G ATMU window, size is 16G.
OCN_DMA_WIN_32G ATMU window, size is 32G.
OCN_DMA_WIN_64G ATMU window, size is 64G.

3.6.2.3.3 enum ocn_dma_interface_t

OCeaN DMA ATMU interface.

Enumerator

OCN_DMA_INTERFACE_PEX PCI Express.

OCN_DMA_INTERFACE_RIO0 RapidIO port 0.
OCN_DMA_INTERFACE_RIO1 RapidIO port 1.
OCN_DMA_INTERFACE_LOCAL0 Local address space OCeaN Port 1.
OCN_DMA_INTERFACE_LOCAL1 Local address space OCeaN Port 6.

3.6.2.4 Function Documentation

3.6.2.4.1 os_status ocnDmaInitialize (ocn_dma_init_params_t * *ocn_dma_config*)

OCeaN DMA initialization function.

Parameters

in	*ocn_dma_config*	- OCeaN DMA initialization parameters structure

Return values

OS_SUCCESS	if OCeaN DMA was initialized successfully

Returns

Error status, encoded in os_error.h, for other errors

3.6.3 OCeaN DMA Module API Runtime

3.6.3.1 Overview

Data Structures

- struct ocn_dma_stride_t
- struct ocn_dma_chain_config_t
- struct ocn_dma_channel_config_t
- union ocn_dma_desc_t
- struct ocn_dma_buff_t
- struct ocn_dma_t

Macros

- #define OCN_DMA_WRITE 0
- #define OCN_DMA_READ 1
- #define DMA_LIST_DESC_ADDR_MASK 0xFFFFFFE0
- #define DMA_LINK_DESC_LAST 0x00000001
- #define OCN_ATMU_DEFAULT_WIN 0
- #define OCN_DMA_MR_CC 0x00000002
- #define OCN_DMA_MR_CS 0x00000001

- #define ocn_buff_addr addr.full_addr
- #define ocnDmaChannelBound(ocn_dma, dma_channel) (bool)(((ocn_dma_t *)ocn_dma)->bound_chain[dma_channel] != NULL? TRUE: FALSE)

Enumerations

- enum ocn_dma_ch_state_t { CHANNEL_UNAVAILABLE, CHANNEL_FREE, CHANNEL_OCCUPIED, CHANNEL_PUBLIC_FREE, CHANNEL_PUBLIC_OCCUPIED }
- enum ocn_dma_bwc_t { DMA_MR_BWC_0001_BYTE = 0x00000000, DMA_MR_BWC_0002_BYTE = 0x01000000, DMA_MR_BWC_0004_BYTE = 0x02000000, DMA_MR_BWC_0008_BYTE = 0x03000000, DMA_MR_BWC_0016_BYTE = 0x04000000, DMA_MR_BWC_0032_BYTE = 0x05000000, DMA_MR_BWC_0064_BYTE = 0x06000000, DMA_MR_BWC_0128_BYTE = 0x07000000, DMA_MR_BWC_0256_BYTE = 0x08000000, DMA_MR_BWC_0512_BYTE = 0x09000000, DMA_MR_BWC_1024_BYTE = 0x0A000000, DMA_MR_BWC_EXCLUSIVE = 0x0F000000 }
- enum ocn_dma_response_t { TRGT_RESPONSE = 0x00000000, NO_TRGT_RESPOSNSE = 0x40000000 }

Functions

- os_dma_handle ocnDmaControllerOpen (ocn_dma_id_t device_id)
- os_status ocnDmaChainCreate (ocn_dma_chain_t *ocn_dma_chain, ocn_dma_chain_config_t *ocn_dma_chain_config)
- os_status ocnDmaTransfer (os_dma_handle ocn_dma, ocn_dma_transfer_config_t *ocn_dma_transfer_config, uint32_t ch)
- os_status ocnDmaChainTransferAdd (ocn_dma_chain_t *ocn_dma_chain, ocn_dma_transfer_config_t *ocn_dma_transfer_config, os_dma_xfer *xfer)
- os_status ocnDmaChannelStart (os_dma_handle ocn_dma, uint32_t ocn_dma_channel)
- bool ocnDmaChannelIsActive (os_dma_handle ocn_dma, uint32_t ocn_dma_channel)
- os_status ocnDmaChannelOpen (os_dma_handle ocn_dma, uint32_t *dma_channel, ocn_dma_channel_config_t *dma_channel_config)
- os_status ocnDmaChannelClose (os_dma_handle ocn_dma, uint32_t ch)
- os_status ocnDmaChannelBind (os_dma_handle ocn_dma, uint32_t ocn_dma_channel, ocn_dma_chain_t *ocn_dma_chain)
- os_status ocnDmaChannelUnbind (os_dma_handle ocn_dma, uint32_t dma_channel)
- INLINE void ocnDmaTransferSizeSet (os_dma_xfer xfer, uint32_t size)
- INLINE void ocnDmaTransferLowSourceSet (os_dma_xfer xfer, uint32_t source)
- INLINE void ocnDmaTransferLowDestinationSet (os_dma_xfer xfer, uint32_t destination)
- INLINE void ocnDmaTransferHighDestinationSet (os_dma_xfer xfer, uint32_t destination)
- INLINE void ocnDmaChainNextSet (ocn_dma_chain_t *ocn_dma_cur_chain, ocn_dma_chain_t *ocn_dma_next_chain)
- INLINE void ocnDmaChainLastSet (ocn_dma_chain_t *ocn_dma_chain)
- INLINE void ocnDmaChainReset (ocn_dma_chain_t *ocn_dma_chain)
- os_status ocnDmaTransferWait (os_dma_handle ocn_dma, uint32_t dma_channel)
- void ocnDmaChannelContinue (os_dma_handle ocn_dma, uint32_t dma_channel)

OCeaN DMA status. Flags are passed in interrupt callback.

- #define OCN_DMA_XFER_ERROR 0x00000080
- #define OCN_DMA_HALTED 0x00000020
- #define OCN_DMA_PROG_ERROR 0x00000010
- #define OCN_DMA_END_LINK 0x00000008
- #define OCN_DMA_BUSY 0x00000004
- #define OCN_DMA_END_SEGMENT 0x00000002
- #define OCN_DMA_END_LIST 0x00000001
- #define OCN_DMA_COMPLETED (OCN_DMA_END_LINK | OCN_DMA_END_SEGMENT | OCN_DMA_END_LIST)
- #define OCN_DMA_HAS_ERROR (OCN_DMA_XFER_ERROR |OCN_DMA_PROG_ERROR)

Interrupt enabling flags.

```
Should be combined in interrupt_en field of ocn_dma_chain_config_t
```

- #define DMA_END_SEG_INT_EN 0x00000200
- #define DMA_END_LINK_INT_EN 0x00000100
- #define DMA_END_LIST_INT_EN 0x00000080
- #define DMA_ERROR_INT_EN 0x00000040

3.6.3.2 Data Structure Documentation

3.6.3.2.1 struct ocn_dma_stride_t

Configuration parameters for the OCeaN DMA chain striding.

Data Fields

- uint32_t stride_size
- uint32_t stride_distance

3.6.3.2.1.1 Field Documentation

3.6.3.2.1.1.1 uint32_t ocn_dma_stride_t::stride_size

Size of the stride.

3.6.3.2.1.1.2 uint32_t ocn_dma_stride_t::stride_distance

During stride enabled transfers, DMA copies stride_size and the skips stride_distance.

3.6.3.2.2 struct ocn_dma_chain_config_t

Configuration parameters for the OCeaN DMA chain.

OCeaN DMA chain a sequence of OCeaN DMA transfers

Data Fields

- uint32_t max_num_of_transfers
- ocn_dma_stride_t * source_stride
- ocn_dma_stride_t * destination_stride
- os_mem_type heap

3.6.3.2.2.1 Field Documentation

3.6.3.2.2.1.1 uint32_t ocn_dma_chain_config_t::max_num_of_transfers

Number of transfers that can be added to this chain by ocnDmaChainTransferAdd().

3.6.3.2.2.1.2 ocn_dma_stride_t* ocn_dma_chain_config_t::source_stride

Stride configuration for the source of DMA transfers.

3.6.3.2.2.1.3 ocn_dma_stride_t* ocn_dma_chain_config_t::destination_stride

Stride configuration for the destination of DMA transfers.

3.6.3.2.2.1.4 os_mem_type ocn_dma_chain_config_t::heap

Which heap to use for buffer descriptors.

3.6.3.2.3 struct ocn_dma_channel_config_t

Configuration parameters for the OCeaN DMA channel.

Data Fields

- ocn_dma_bwc_t priority
- os_hwi_function ocn_dma_callback
- uint32_t interrupt_en

3.6.3.2.3.1 Field Documentation

3.6.3.2.3.1.1 ocn_dma_bwc_t ocn_dma_channel_config_t::priority

Channel bandwidth control (relative priority)

3.6.3.2.3.1.2 os_hwi_function ocn_dma_channel_config_t::ocn_dma_callback

Application callback function to call when channel finishes transferring.

3.6.3.2.3.1.3 uint32_t ocn_dma_channel_config_t::interrupt_en

A combination of the interrupt enable flags: DMA_END_LIST_INT_EN, DMA_END_LINK_INT_EN, DMA_END_SEG_INT_EN and DMA_ERROR_INT_EN.

3.6.3.2.4 union ocn_dma_desc_t

Configuration parameters for a OCeaN DMA transfer/chain.

OCeaN DMA transfer describes a specific data move that can be stored in an OCeaN DMA chain and then executed.

3.6.3.2.4.1 Field Documentation

3.6.3.2.4.1.1 uint32_t ocn_dma_desc_t::satr

Source Attributes Register.

3.6.3.2.4.1.2 uint32_t ocn_dma_desc_t::sar

Source Address Register.

3.6.3.2.4.1.3 uint32_t ocn_dma_desc_t::datr

Destination Attributes Register.

3.6.3.2.4.1.4 uint32_t ocn_dma_desc_t::dar

Destination Address Register.

3.6.3.2.4.1.5 uint32_t ocn_dma_desc_t::enlndar

Next link descriptor extended address.

3.6.3.2.4.1.6 uint32_t ocn_dma_desc_t::nlndar

Next Link Descriptor Address Registers.

3.6.3.2.4.1.7 uint32_t ocn_dma_desc_t::bcr

Byte Count register.

3.6.3.2.4.1.8 uint32_t ocn_dma_desc_t::ext_next_list_desc

Next list descriptor extended address.

3.6.3.2.4.1.9 uint32_t ocn_dma_desc_t::next_list_desc

Next Link Descriptor.

3.6.3.2.4.1.10 uint32_t ocn_dma_desc_t::ext_first_link_descriptor

First Link extended Descriptor.

3.6.3.2.4.1.11 uint32_t ocn_dma_desc_t::first_link_descriptor

First Link Descriptor.

3.6.3.2.4.1.12 uint32_t ocn_dma_desc_t::source_stride

Source Stride - copied from the chain configuration.

3.6.3.2.4.1.13 uint32_t ocn_dma_desc_t::destination_stride

Destination Stride - copied from the chain configuration.

3.6.3.2.5 struct ocn_dma_buff_t

Description of either source or destination of a DMA transfer.

Data Fields

- union {
  uint64_t full_addr
  } addr

- bool stride_enable
- bool snoop_enable
- bool enhanced_enable
- uint32_t high_addr
- uint32_t low_addr

3.6.3.2.5.1 Field Documentation

3.6.3.2.5.1.1 uint64_t ocn_dma_buff_t::full_addr

Full 36 bit (OCeaN address space size) address.

3.6.3.2.5.1.2 uint32_t ocn_dma_buff_t::high_addr

4 msb of the 36 bit address

3.6.3.2.5.1.3 uint32_t ocn_dma_buff_t::low_addr

32 lsb of the 36 bit address

3.6.3.2.5.1.4 union { ... } ocn_dma_buff_t::addr

Actual address.

If accessing local addresses high_addr must be set to 0. If accessing HSSI addresses (sRIO/PCI-e) the 4 lower bits of high_addr are used as the 4 high bits of OCeaN address. Other bits must be 0.

3.6.3.2.5.1.5 bool ocn_dma_buff_t::stride_enable

Enable striding according to chain.

3.6.3.2.5.1.6 bool ocn_dma_buff_t::snoop_enable

Enable snooping.

3.6.3.2.5.1.7 bool ocn_dma_buff_t::enhanced_enable

Enable enhanced mode.

3.6.3.2.6 struct ocn_dma_t

OCeaN DMA controller structure.

Data Fields

- unsigned int device_id
- void ∗ ocn_dma_ch_map [NUMBER_OF_OCN_DMA_CH]
- ocn_dma_ch_state_t channel_state [NUMBER_OF_OCN_DMA_CH]
- ocn_dma_chain_t ∗ bound_chain [NUMBER_OF_OCN_DMA_CH]
- os_hwi_function ocn_dma_callback [NUMBER_OF_OCN_DMA_CH]
- uint32_t channel_mr [NUMBER_OF_OCN_DMA_CH]

3.6.3.2.6.1 Field Documentation

3.6.3.2.6.1.1 unsigned int ocn_dma_t::device_id

Number of OCeaN DMA.

3.6.3.2.6.1.2 void∗ ocn_dma_t::ocn_dma_ch_map[NUMBER_OF_OCN_DMA_CH]

OCean DMA channels map.

3.6.3.2.6.1.3 ocn_dma_ch_state_t ocn_dma_t::channel_state[NUMBER_OF_OCN_DMA_CH]

OCean DMA channels state.

3.6.3.2.6.1.4 ocn_dma_chain_t∗ ocn_dma_t::bound_chain[NUMBER_OF_OCN_DMA_CH]

OCean DMA chains bound to channels.

3.6.3.2.6.1.5 os_hwi_function ocn_dma_t::ocn_dma_callback[NUMBER_OF_OCN_DMA_CH]

Channel callback function for completion.

3.6.3.2.6.1.6 uint32_t ocn_dma_t::channel_mr[NUMBER_OF_OCN_DMA_CH]

Channel Mode Register.

3.6.3.3 Macro Definition Documentation

3.6.3.3.1 #define OCN_DMA_WRITE 0

OCeaN DMA write data.

3.6.3.3.2 #define OCN_DMA_READ 1

OCeaN DMA read data.

3.6.3.3.3 #define DMA_LIST_DESC_ADDR_MASK 0xFFFFFFE0

List descending address mask.

3.6.3.3.4 #define DMA_LINK_DESC_LAST 0x00000001

Last descending address.

3.6.3.3.5 #define OCN_ATMU_DEFAULT_WIN 0

Default ATMU window.

3.6.3.3.6 #define OCN_DMA_MR_CC 0x00000002

Channel continue.

3.6.3.3.7 #define OCN_DMA_MR_CS 0x00000001

Channel start.

3.6.3.3.8 #define OCN_DMA_XFER_ERROR 0x00000080

Indicates whether an error occurred during the DMA transfer.

3.6.3.3.9 #define OCN_DMA_HALTED 0x00000020

Indicates whether the transfer is halted; If the bit is set, the channel was successfully halted by software and can be restarted.

3.6.3.3.10 #define OCN_DMA_PROG_ERROR 0x00000010

Indicates whether a programming error was detected that prevented the DMA transfer.

3.6.3.3.11 #define OCN_DMA_END_LINK 0x00000008

This bit is set after transferring the last block of data in the last link if MR[EOLSIE] is set.

3.6.3.3.12 #define OCN_DMA_BUSY 0x00000004

Indicates the current status of the channel.

3.6.3.3.13 #define OCN_DMA_END_SEGMENT 0x00000002

This bit is set:

- In chaining mode, after finishing a data transfer, if MR[EOLSIE] is set or if CLNDAR[EOSIE} is set;
- In direct mode, if MR[EOSIE] is set

3.6.3.3.14 #define OCN_DMA_END_LIST 0x00000001

This bit is set after transferring the last block of data in the last list descriptor, if MR[EOLSIE] is set.

3.6.3.3.15 #define OCN_DMA_COMPLETED (OCN_DMA_END_LINK | OCN_DMA_END_SEGMENT | OCN_DMA_END_LIST)

Collects all types of successful end-of-transfer indications.

3.6.3.3.16 #define OCN_DMA_HAS_ERROR (OCN_DMA_XFER_ERROR |OCN_DMA_PROG_ERROR)

Collects all types of error indications.

3.6.3.3.17 #define DMA_END_SEG_INT_EN 0x00000200

End-of-segments interrupt enable.

3.6.3.3.18 #define DMA_END_LINK_INT_EN 0x00000100

End-of-links interrupt enable.

3.6.3.3.19 #define DMA_END_LIST_INT_EN 0x00000080

End-of-list interrupt enable.

3.6.3.3.20 #define DMA_ERROR_INT_EN 0x00000040

Error interrupt enable.

3.6.3.3.21 #define ocn_buff_addr addr.full_addr

OCeaN buffer address.

3.6.3.3.22 #define ocnDmaChannelBound(*ocn_dma, dma_channel*) (bool)(((ocn_dma_t *)ocn_dma)->bound_chain[dma_channel] != NULL? TRUE: FALSE)

Checks if the channel is bound to a chain

Parameters

in	*ocn_dma*	- A handle to a Ocean DMA controller, returned by ocnDmaControllerOpen()
in	*dma_channel*	- A handle to a Ocean DMA channel, returned by ocnDmaChannelOpen()

Return values

TRUE	if the channel is bound to a chain
FALSE	if the channel is not bound to a chain

3.6.3.4 Enumeration Type Documentation

3.6.3.4.1 enum ocn_dma_ch_state_t

State of OCeaN DMA channel.

Enumerator

CHANNEL_UNAVAILABLE Channel not allocated to this core.
CHANNEL_FREE Channel private to this core and free.
CHANNEL_OCCUPIED Channel private to this core and accupied.
CHANNEL_PUBLIC_FREE Channel shared among cores, not in use by this core.
CHANNEL_PUBLIC_OCCUPIED Channel shared among cores, in use by this core.

3.6.3.4.2 enum ocn_dma_bwc_t

DMA Bandwidth/pause control options.

If multiple channels are executing transfers concurrently, this value determines how many bytes a channel can transfer before the DMA controller shifts to the next channel

Enumerator

DMA_MR_BWC_0001_BYTE Shift control to next channel after 1 byte.
DMA_MR_BWC_0002_BYTE Shift control to next channel after 2 bytes.
DMA_MR_BWC_0004_BYTE Shift control to next channel after 4 bytes.
DMA_MR_BWC_0008_BYTE Shift control to next channel after 8 bytes.
DMA_MR_BWC_0016_BYTE Shift control to next channel after 16 bytes.
DMA_MR_BWC_0032_BYTE Shift control to next channel after 32 bytes.
DMA_MR_BWC_0064_BYTE Shift control to next channel after 64 bytes.
DMA_MR_BWC_0128_BYTE Shift control to next channel after 128 bytes.
DMA_MR_BWC_0256_BYTE Shift control to next channel after 256 bytes.
DMA_MR_BWC_0512_BYTE Shift control to next channel after 512 bytes.
DMA_MR_BWC_1024_BYTE Shift control to next channel after 1024 bytes.
DMA_MR_BWC_EXCLUSIVE Disable bandwidth sharing to allow uninterrupted transfers from each channel.

3.6.3.4.3 enum ocn_dma_response_t

Type of response required by the OCeaN DMA, prior to its indication that a transfer is finished.

Enumerator

TRGT_RESPONSE Response is required from the target.
NO_TRGT_RESPOSNSE No response required.

3.6.3.5 Function Documentation

3.6.3.5.1 os_dma_handle ocnDmaControllerOpen (ocn_dma_id_t *device_id*)

Opens an OCeaN DMA Device.

The function opens an OCeaN DMA Device and returns a handle. This handle

Parameters

in	*device_id*	- OCeaN DMA device ID.

Return values

Device	handle
NULL	if failed to open the controller

3.6.3.5.2 os_status ocnDmaChainCreate (ocn_dma_chain_t * *ocn_dma_chain,* ocn_dma_chain_config_t * *ocn_dma_chain_config*)

Creates an empty OCeaN DMA chain.

The function also initializes the chain structure according to the configuration parameters.

Parameters

out	*ocn_dma_chain*	- gets the chain handle
in	*ocn_dma_chain_config*	- OCeaN DMA chain configuration parameters.

Return values

OS_SUCCESS	if chain was opened successfully
OS_ERR_NO_MEMORY	if there is not enough meory to create the chain

Returns

Error status, encoded in os_error.h, for other errors

3.6.3.5.3 os_status ocnDmaTransfer (os_dma_handle *ocn_dma,* ocn_dma_transfer_config_t * *ocn_dma_transfer_config,* uint32_t *ch*)

OCeaN DMA Transfer execution.

Parameters

in	*ocn_dma*	- A handle to a Ocean DMA controller, returned by ocnDmaController↩ Open()
in	*ocn_dma_transfer_config*	- OCeaN DMA transfer configuration parameters
in	*ch*	- OCeaN DMA channel index

Return values

OS_SUCCESS	if transfer finished successfully
OS_FAIL	if any of the possible DMA errors were reported

Warning

If user calls this function consecutively, with the first transaction triggerring an interrupt; it is the user's responsibility to wait for the previous interrupt to trigger. This allows for optimization of the function in terms of runtime and interrupt latency.

3.6.3.5.4 os_status ocnDmaChainTransferAdd (ocn_dma_chain_t * *ocn_dma_chain,* ocn_dma_transfer_config_t * *ocn_dma_transfer_config,* os_dma_xfer * *xfer*)

Adds a transfer to a OCeaN DMA chain.

Parameters

in	*ocn_dma_chain*	- A handle to a OCeaN DMA chain created by ocnDmaChainCreate()
in	*ocn_dma_transfer_config*	- OCeaN DMA transfer configuration parameters.
out	*xfer*	- OCeaN DMA transfer handle

Returns

OS_SUCCESS (function always succeeds, although it may assert)

3.6.3.5.5 os_status ocnDmaChannelStart (os_dma_handle *ocn_dma,* uint32_t *ocn_dma_channel*)

Trigger a channel.

Parameters

in	*ocn_dma*	- A handle to a Ocean DMA controller, returned by ocnDmaControllerOpen()
in	*ocn_dma_channel*	- A handle to a Ocean DMA channel, returned by ocnDmaChannelOpen()

Returns

OS_SUCCESS (function always succeeds, although it may assert)

3.6.3.5.6 bool ocnDmaChannelIsActive (os_dma_handle *ocn_dma,* uint32_t *ocn_dma_channel*)

Used to poll the channel to see when it is executing a transfer.

Parameters

in	*ocn_dma*	- A handle to a Ocean DMA controller, returned by ocnDmaController↩ Open()
in	*ocn_dma_↩ channel*	- A handle to a Ocean DMA channel, returned by ocnDmaChannel↩ Open()

Return values

TRUE	if the channel is active
FALSE	if the channel is not active

3.6.3.5.7 os_status ocnDmaChannelOpen (os_dma_handle *ocn_dma,* uint32_t * *dma_channel,* ocn_dma_channel_config_t * *dma_channel_config*)

Open and configure an OCeaN DMA channel.

Parameters

in	*ocn_dma*	- A handle to a Ocean DMA controller, returned by ocnDmaController↩ Open()
out	*dma_channel*	- Opened channel number.
in	*dma_channel↩ _config*	- Configuration parameters

Return values

OS_SUCCESS	if found and configured an available channel
OS_ERR_DMA_RESO↩ URCE_UNAVAILABLE	if no channel is available

Warning

Channel allocation will first try to allocate a private channel. Only if there is no private channels will the driver allocate (if available) a shared channel.

3.6.3.5.8 os_status ocnDmaChannelClose (os_dma_handle *ocn_dma,* uint32_t *ch*)

Free OCeaN DMA channel.

Parameters

in	*ocn_dma*	- A handle to a Ocean DMA controller, returned by ocnDmaController↩ Open()
in	*ch*	- A handle to a Ocean DMA channel, returned by ocnDmaChannel↩ Open()

Return values

OS_SUCCESS	if freed the channel
OS_ERR_DMA_RESO↩ URCE_ALREADY_FREE	the channel is already free

3.6.3.5.9 os_status ocnDmaChannelBind (os_dma_handle *ocn_dma*, uint32_t *ocn_dma_channel*, ocn_dma_chain_t * *ocn_dma_chain*)

Bind a chain to a channel.

Parameters

in	*ocn_dma*	- A handle to a Ocean DMA controller, returned by ocnDmaController↩ Open()
in	*ocn_dma_↩ channel*	- A handle to a Ocean DMA channel, returned by ocnDmaChannel↩ Open()
in	*ocn_dma_↩ chain*	- A handle to a OCeaN DMA chain created by ocnDmaChainCreate()

Returns

OS_SUCCESS (function always succeeds, although it may assert)

3.6.3.5.10 os_status ocnDmaChannelUnbind (os_dma_handle *ocn_dma*, uint32_t *dma_channel*)

Unbind the bound chain from a channel.

Parameters

in	*ocn_dma*	- A handle to a Ocean DMA controller, returned by ocnDmaController↩ Open()
in	*dma_channel*	- A handle to a Ocean DMA channel, returned by ocnDmaChannel↩ Open()

Returns

OS_SUCCESS (function always succeeds, although it may assert)

3.6.3.5.11 INLINE void ocnDmaTransferSizeSet (os_dma_xfer *xfer,* uint32_t *size*)

Set size of a specific transfer.

Parameters

in	*xfer*	- Transfer handle, populated by ocnDmaChainTransferAdd()
in	*size*	- New size to configure the transfer with

3.6.3.5.12 INLINE void ocnDmaTransferLowSourceSet (os_dma_xfer *xfer,* uint32_t *source*)

Set low bits in source address of a specific transfer.

Parameters

in	*xfer*	- Transfer handle, populated by ocnDmaChainTransferAdd()
in	*source*	- New source address (32 lsb) to configure the transfer with

3.6.3.5.13 INLINE void ocnDmaTransferLowDestinationSet (os_dma_xfer *xfer,* uint32_t *destination*)

Set low bits in destination address of a specific transfer.

Parameters

in	*xfer*	- Transfer handle, populated by ocnDmaChainTransferAdd()
in	*destination*	- New destination address (32 lsb) to configure the transfer with

3.6.3.5.14 INLINE void ocnDmaTransferHighDestinationSet (os_dma_xfer *xfer,* uint32_t *destination*)

Set high bits in destination address of a specific transfer.

Parameters

in	*xfer*	- Transfer handle, populated by ocnDmaChainTransferAdd()
in	*destination*	- New destination address (4 msb) to configure the transfer with

3.6.3.5.15 INLINE void ocnDmaChainNextSet (ocn_dma_chain_t * *ocn_dma_cur_chain*, ocn_dma_chain_t * *ocn_dma_next_chain*)

Concatenate two chains.

Parameters

in	*ocn_dma_cur_chain*	- A handle to the first OCeaN DMA chain in the concatenation, created by ocnDmaChainCreate()
in	*ocn_dma_next_chain*	- A handle to the second OCeaN DMA chain in the concatenation, created by ocnDmaChainCreate()

3.6.3.5.16 INLINE void ocnDmaChainLastSet (ocn_dma_chain_t * *ocn_dma_chain*)

Make channel terminate at the end of this chain.

Parameters

in	*ocn_dma_chain*	- A handle to the OCeaN DMA chain (created by ocnDmaChainCreate()), after which to terminate the channel

3.6.3.5.17 INLINE void ocnDmaChainReset (ocn_dma_chain_t * *ocn_dma_chain*)

Reset the chain.

Parameters

in	*ocn_dma_chain*	- A handle to the OCeaN DMA chain (created by ocnDmaChainCreate()), to reset

3.6.3.5.18 os_status ocnDmaTransferWait (os_dma_handle *ocn_dma*, uint32_t *dma_channel*)

Waits for channel to finish pending transactions

Parameters

in	*ocn_dma*	- A handle to a Ocean DMA controller, returned by ocnDmaController↩ Open()
in	*dma_channel*	- A handle to a Ocean DMA channel, returned by ocnDmaChannel↩ Open()

Returns

OS_SUCCESS (function always succeeds, although it may assert)

3.6.3.5.19 void ocnDmaChannelContinue (os_dma_handle *ocn_dma,* uint32_t *dma_channel*)

Continues execution on an existing channel.

This function is generally called after calling ocnDmaChainNextSet() to add an additional chain to a channel. If the channel finished prior to concatenating the new chain, this function will restart the DMA engine; if not, it'll have no effect

Parameters

in	*ocn_dma*	- A handle to a Ocean DMA controller, returned by ocnDmaController↩ Open()
in	*dma_channel*	- A handle to a Ocean DMA channel, returned by ocnDmaChannel↩ Open()

3.7 DPAA Module API

3.7.1 Overview

DPAA module

Modules

- Buffer Manager Module API
- Queue Manager Module API

Data Structures

- struct dpaa_fd_t

Macros

- #define QM_MAX_NUM_OF_POOL_CHANNELS 15
- #define QM_MAX_NUM_OF_WQ 8
- #define QM_MAX_NUM_OF_CGS 256
- #define QM_MAX_NUM_OF_FQIDS (16 ∗ MEGABYTE)

Enumerations

- enum dpaa_sw_portal_t
- enum rx_store_response_t { E_RX_STORE_RESPONSE_PAUSE, E_RX_STORE_RESPONSE↩ _CONTINUE }
- enum dpaa_dc_portal_t

Variables

- volatile uint32_t dpaa_fd_t::id
- volatile uint32_t dpaa_fd_t::addrl
- volatile uint32_t dpaa_fd_t::length
- volatile uint32_t dpaa_fd_t::status
- uint8_t qm_revision_info_t::major_rev
- uint8_t qm_revision_info_t::minor_rev
- qm_rejection_code_t qm_rejected_frame_info_t::rejection_code
- uint8_t qm_rejected_frame_info_t::cg_id
- struct {
  uint8_t qm_rejected_frame_info_t::cg_id
  } qm_rejected_frame_info_t::cg

- uint32_t qm_context_a_t::res [2]
- uint8_t qm_param_t::guest_id
- uintptr_t qm_param_t::sw_portals_base_address
- uint16_t qm_param_t::liodn
- uint32_t qm_param_t::total_num_of_fqids
- os_mem_type qm_param_t::fqd_mem_partition_id
- os_mem_type qm_param_t::pfdr_mem_partition_id
- qm_exceptions_callback_t ∗ qm_param_t::f_exception
- void ∗ qm_param_t::h_app
- os_hwi_handle qm_param_t::err_irq
- uint32_t qm_param_t::part_fqid_base
- uint32_t qm_param_t::part_num_of_fqids
- uint16_t qm_param_t::part_cgs_base
- uint16_t qm_param_t::part_num_of_cgs
- uint8_t qm_portal_stash_param_t::stash_dest_queue
- uint8_t qm_portal_stash_param_t::eqcr
- bool qm_portal_stash_param_t::eqcr_high_pri
- bool qm_portal_stash_param_t::dqrr
- uint16_t qm_portal_stash_param_t::dqrr_liodn
- bool qm_portal_stash_param_t::dqrr_high_pri
- bool qm_portal_stash_param_t::fd_fq
- uint16_t qm_portal_stash_param_t::fd_fq_liodn
- bool qm_portal_stash_param_t::fd_fq_high_pri

- bool qm_portal_stash_param_t::fd_fq_drop
- uintptr_t qm_portal_param_t::ce_base_address
- uintptr_t qm_portal_param_t::ci_base_address
- void * qm_portal_param_t::h_qm
- dpaa_sw_portal_t qm_portal_param_t::sw_portal_id
- os_hwi_handle qm_portal_param_t::irq
- uint16_t qm_portal_param_t::fd_liodn_offset
- qm_received_frame_callback_t * qm_portal_param_t::f_dflt_frame
- qm_rejected_frame_callback_t * qm_portal_param_t::f_rejected_frame
- void * qm_portal_param_t::h_app
- void * qm_fqr_congestion_avoidance_params_t::h_qm_cg
- int8_t qm_fqr_congestion_avoidance_params_t::overhead_accounting_length
- uint32_t qm_fqr_congestion_avoidance_params_t::fq_tail_drop_threshold
- void * qm_fqr_params_t::h_qm
- void * qm_fqr_params_t::h_qm_portal
- bool qm_fqr_params_t::init_parked
- bool qm_fqr_params_t::hold_active
- bool qm_fqr_params_t::avoid_blocking
- bool qm_fqr_params_t::prefer_in_cache
- bool qm_fqr_params_t::use_context_a_for_stash
- uint8_t qm_fqr_params_t::frame_annotation_size
- uint8_t qm_fqr_params_t::frame_data_size
- uint8_t qm_fqr_params_t::fq_context_size
- uint64_t qm_fqr_params_t::fq_context_addr
- qm_context_a_t * qm_fqr_params_t::context_a
- qm_context_b_t * qm_fqr_params_t::context_b
- qm_fq_channel_t qm_fqr_params_t::channel
- uint8_t qm_fqr_params_t::wq
- bool qm_fqr_params_t::shadow_mode
- uint32_t qm_fqr_params_t::num_of_fqids
- bool qm_fqr_params_t::use_force
- uint32_t qm_fqr_params_t::align
- uint32_t qm_fqr_params_t::fqid
- bool qm_fqr_params_t::congestion_avoidance_enable
- qm_fqr_congestion_avoidance_params_t qm_fqr_params_t::congestion_avoidance_params
- qm_fqr_mod_opt_t qm_fqr_modify_params_t::mod_options
- bool qm_fqr_modify_params_t::hold_active
- bool qm_fqr_modify_params_t::avoid_blocking
- bool qm_fqr_modify_params_t::prefer_in_cache
- bool qm_fqr_modify_params_t::use_context_a_for_stash
- uint8_t qm_fqr_modify_params_t::frame_annotation_size
- uint8_t qm_fqr_modify_params_t::frame_data_size
- uint8_t qm_fqr_modify_params_t::fq_context_size
- uint64_t qm_fqr_modify_params_t::fq_context_addr
- qm_context_a_t qm_fqr_modify_params_t::context_a
- qm_context_b_t qm_fqr_modify_params_t::context_b
- qm_fq_channel_t qm_fqr_modify_params_t::channel
- uint8_t qm_fqr_modify_params_t::wq
- bool qm_fqr_modify_params_t::congestion_avoidance_enable
- qm_fqr_congestion_avoidance_params_t qm_fqr_modify_params_t::congestion_avoidance_↩ params
- qm_fqr_drained_completion_cb_t * qm_fqr_modify_params_t::f_completion_cb
- bool qm_fqr_modify_params_t::deliver_frame
- qm_received_frame_callback_t * qm_fqr_modify_params_t::f_callback
- void * qm_fqr_modify_params_t::h_app
- uint32_t qm_cg_wred_curve_t::max_th

- uint32_t qm_cg_wred_curve_t::min_th
- uint8_t qm_cg_wred_curve_t::probability_denominator
- void * qm_cg_params_t::h_qm
- void * qm_cg_params_t::h_qm_portal
- bool qm_cg_params_t::frame_count
- bool qm_cg_params_t::wred_enable
- qm_cg_wred_params_t qm_cg_params_t::wred_params
- bool qm_cg_params_t::tail_drop_enable
- uint32_t qm_cg_params_t::threshold
- bool qm_cg_params_t::notify_dc_portal
- dpaa_dc_portal_t qm_cg_params_t::dc_portal_id
- qm_exceptions_callback_t * qm_cg_params_t::f_exception
- void * qm_cg_params_t::h_app

Frame descriptor macros

- #define DPAA_FD_DD_MASK 0xc0000000
- #define DPAA_FD_PID_MASK 0x3f000000
- #define DPAA_FD_ELIODN_MASK 0x0000f000
- #define DPAA_FD_BPID_MASK 0x00ff0000
- #define DPAA_FD_ADDRH_MASK 0x000000ff
- #define DPAA_FD_ADDRL_MASK 0xffffffff
- #define DPAA_FD_FORMAT_MASK 0xe0000000
- #define DPAA_FD_OFFSET_MASK 0x1ff00000
- #define DPAA_FD_LENGTH_MASK 0x000fffff
- #define DPAA_FD_GET_DD(fd) ((((dpaa_fd_t *)fd)->id & DPAA_FD_DD_MASK) >> (31-1))
- #define DPAA_FD_GET_PID(fd)
- #define DPAA_FD_GET_BPID(fd) ((((dpaa_fd_t *)fd)->id & DPAA_FD_BPID_MASK) >> (31-15))
- #define DPAA_FD_GET_ADDRH(fd) (((dpaa_fd_t *)fd)->id & DPAA_FD_ADDRH_MASK)
- #define DPAA_FD_GET_ADDRL(fd) ((dpaa_fd_t *)fd)->addrl
- #define DPAA_FD_GET_PHYS_ADDR(fd) ((os_phys_ptr)(((uint64_t)DPAA_FD_GET_ADDRH(fd) << 32) | (uint64_t)DPAA_FD_GET_ADDRL(fd)))
- #define DPAA_FD_GET_FORMAT(fd) ((((dpaa_fd_t *)fd)->length & DPAA_FD_FORMAT_MASK) >> (31-2))
- #define DPAA_FD_GET_OFFSET(fd) ((((dpaa_fd_t *)fd)->length & DPAA_FD_OFFSET_MASK) >> (31-11))
- #define DPAA_FD_GET_LENGTH(fd) (((dpaa_fd_t *)fd)->length & DPAA_FD_LENGTH_MASK)
- #define DPAA_FD_GET_STATUS(fd) ((dpaa_fd_t *)fd)->status
- #define DPAA_FD_GET_ADDR(fd, virt) if (osMmuDataPhysToVirtManual(DPAA_FD_GET_PHYS_ADDR(fd),&virt) != OS_SUCCESS) OS_ASSERT;
- #define DPAA_FD_SET_DD(fd, val) (((dpaa_fd_t *)fd)->id = ((((dpaa_fd_t *)fd)->id & ~DPAA_FD_DD_MASK) | (((val) << (31-1)) & DPAA_FD_DD_MASK)))
- #define **DPAA_FD_SET_PID**(fd, val) (((dpaa_fd_t *)fd)->id = ((((dpaa_fd_t *)fd)->id & ~(DPAA_FD_PID_MASK|DPAA_FD_ELIODN_MASK)) | ((((val) << (31-7)) & DPAA_FD_PID_MASK) | ((((val)>>6) << (31-19)) & DPAA_FD_ELIODN_MASK))))
- #define DPAA_FD_SET_BPID(fd, val) (((dpaa_fd_t *)fd)->id = ((((dpaa_fd_t *)fd)->id & ~DPAA_FD_BPID_MASK) | (((val) << (31-15)) & DPAA_FD_BPID_MASK)))
- #define DPAA_FD_SET_ADDRH(fd, val) (((dpaa_fd_t *)fd)->id = ((((dpaa_fd_t *)fd)->id & ~DPAA_FD_ADDRH_MASK) | ((val) & DPAA_FD_ADDRH_MASK)))

- #define DPAA_FD_SET_ADDRL(fd, val) ((dpaa_fd_t *)fd)->addrl = (val)
- #define DPAA_FD_SET_ADDR(fd, val)
- #define DPAA_FD_SET_FORMAT(fd, val) (((dpaa_fd_t *)fd)->length = ((((dpaa_fd_t *)fd)->length & ~DPAA_FD_FORMAT_MASK) | (((val) << (31-2))& DPAA_FD_FORMAT_MASK)))
- #define DPAA_FD_SET_OFFSET(fd, val) (((dpaa_fd_t *)fd)->length = ((((dpaa_fd_t *)fd)->length & ~DPAA_FD_OFFSET_MASK) | (((val) << (31-11))& DPAA_FD_OFFSET_MASK)))
- #define DPAA_FD_SET_LENGTH(fd, val) (((dpaa_fd_t *)fd)->length = (((dpaa_fd_t *)fd)->length & ~DPAA_FD_LENGTH_MASK) | ((val) & DPAA_FD_LENGTH_MASK))
- #define DPAA_FD_SET_STATUS(fd, val) ((dpaa_fd_t *)fd)->status = (val)

3.7.2 Data Structure Documentation

3.7.2.1 struct dpaa_fd_t

Frame descriptor.

Data Fields

- volatile uint32_t id
- volatile uint32_t addrl
- volatile uint32_t length
- volatile uint32_t status

3.7.3 Macro Definition Documentation

3.7.3.1 #define QM_MAX_NUM_OF_POOL_CHANNELS 15

Total number of channels, dedicated and pool.

3.7.3.2 #define QM_MAX_NUM_OF_WQ 8

Number of work queues per channel.

3.7.3.3 #define QM_MAX_NUM_OF_CGS 256

Congestion groups number.

3.7.3.4 #define QM_MAX_NUM_OF_FQIDS (16 * MEGABYTE)

FQIDs range - 24 bits.

3.7.3.5 #define DPAA_FD_DD_MASK 0xc0000000

FD DD field mask.

3.7.3.6 #define DPAA_FD_PID_MASK 0x3f000000

FD PID field mask.

3.7.3.7 #define DPAA_FD_ELIODN_MASK 0x0000f000

FD ELIODN field mask.

3.7.3.8 #define DPAA_FD_BPID_MASK 0x00ff0000

FD BPID field mask.

3.7.3.9 #define DPAA_FD_ADDRH_MASK 0x000000ff

FD ADDRH field mask.

3.7.3.10 #define DPAA_FD_ADDRL_MASK 0xffffffff

FD ADDRL field mask.

3.7.3.11 #define DPAA_FD_FORMAT_MASK 0xe0000000

FD FORMAT field mask.

3.7.3.12 #define DPAA_FD_OFFSET_MASK 0x1ff00000

FD OFFSET field mask.

3.7.3.13 #define DPAA_FD_LENGTH_MASK 0x000fffff

FD LENGTH field mask.

3.7.3.14 #define DPAA_FD_GET_DD(*fd*) ((((dpaa_fd_t *)fd)->id & DPAA_FD_DD_MASK) >> (31-1))

Macro to get FD DD field.

3.7.3.15 #define DPAA_FD_GET_PID(*fd*)

Value:

```
(((((dpaa_fd_t *)fd)->id & DPAA_FD_PID_MASK) >> (31-7)) | \
                                    ((((dpaa_fd_t *)fd)->id &
      DPAA_FD_ELIODN_MASK) >> (31-19-6)))
```

Macro to get FD PID field.

3.7.3.16 #define DPAA_FD_GET_BPID(*fd*) ((((dpaa_fd_t *)fd)->id & DPAA_FD_BPID_MASK) >> (31-15))

Macro to get FD BPID field.

3.7.3.17 #define DPAA_FD_GET_ADDRH(*fd*) (((dpaa_fd_t *)fd)->id & DPAA_FD_ADDRH_MASK)

Macro to get FD ADDRH field.

3.7.3.18 #define DPAA_FD_GET_ADDRL(*fd*) ((dpaa_fd_t *)fd)->addrl

Macro to get FD ADDRL field.

3.7.3.19 #define DPAA_FD_GET_PHYS_ADDR(*fd*) ((os_phys_ptr)(((uint64_t)DPAA_FD_GET_ADDRH(fd) << 32) | (uint64_t)DPAA_FD_GET_ADDRL(fd)))

Macro to get FD ADDR field.

3.7.3.20 #define DPAA_FD_GET_FORMAT(*fd*) ((((dpaa_fd_t *)fd)->length & DPAA_FD_FORMAT_MASK) >> (31-2))

Macro to get FD FORMAT field.

3.7.3.21 #define DPAA_FD_GET_OFFSET(*fd*) ((((dpaa_fd_t *)fd)->length & DPAA_FD_OFFSET_MASK) >> (31-11))

Macro to get FD OFFSET field.

3.7.3.22 #define DPAA_FD_GET_LENGTH(*fd*) (((dpaa_fd_t *)fd)->length & DPAA_FD_LENGTH_MASK)

Macro to get FD LENGTH field.

3.7.3.23 #define DPAA_FD_GET_STATUS(*fd*) ((dpaa_fd_t *)fd)->status

Macro to get FD STATUS field.

3.7.3.24 #define DPAA_FD_GET_ADDR(*fd, virt*) if (osMmuDataPhysToVirtManual(DPAA_FD_GET_PHYS_ADDR(fd),&virt) != OS_SUCCESS) OS_ASSERT;

Macro to get FD ADDR (virtual)

3.7.3.25 #define DPAA_FD_SET_DD(*fd, val*) (((dpaa_fd_t *)fd)->id = ((((dpaa_fd_t *)fd)->id & ~DPAA_FD_DD_MASK) | (((val) << (31-1)) & DPAA_FD_DD_MASK)))

Macro to set FD DD field.

Macro to set FD PID field or LIODN offset

3.7.3.26 #define DPAA_FD_SET_BPID(*fd, val*) (((dpaa_fd_t *)fd)->id = ((((dpaa_fd_t *)fd)->id & ~DPAA_FD_BPID_MASK) | (((val) << (31-15)) & DPAA_FD_BPID_MASK)))

Macro to set FD BPID field.

3.7.3.27 #define DPAA_FD_SET_ADDRH(*fd, val*) (((dpaa_fd_t *)fd)->id = ((((dpaa_fd_t *)fd)->id & ~DPAA_FD_ADDRH_MASK) | ((val) & DPAA_FD_ADDRH_MASK)))

Macro to set FD ADDRH field.

3.7.3.28 #define DPAA_FD_SET_ADDRL(*fd, val*) ((dpaa_fd_t *)fd)->addrl = (val)

Macro to set FD ADDRL field.

3.7.3.29 #define DPAA_FD_SET_ADDR(*fd, val*)

Macro to set FD ADDR field.

3.7.3.30 #define DPAA_FD_SET_FORMAT(*fd, val*) (((dpaa_fd_t *)fd)->length = ((((dpaa_fd_t *)fd)->length & ~DPAA_FD_FORMAT_MASK) | (((val) << (31-2))& DPAA_FD_FORMAT_MASK)))

Macro to set FD FORMAT field.

3.7.3.31 #define DPAA_FD_SET_OFFSET(*fd, val*) (((dpaa_fd_t *)fd)->length = ((((dpaa_fd_t *)fd)->length & ~DPAA_FD_OFFSET_MASK) | (((val) << (31-11))& DPAA_FD_OFFSET_MASK)))

Macro to set FD OFFSET field.

3.7.3.32 #define DPAA_FD_SET_LENGTH(*fd, val*) (((dpaa_fd_t *)fd)->length = (((dpaa_fd_t *)fd)->length & ~DPAA_FD_LENGTH_MASK) | ((val) & DPAA_FD_LENGTH_MASK))

Macro to set FD LENGTH field.

3.7.3.33 #define DPAA_FD_SET_STATUS(*fd, val*) ((dpaa_fd_t *)fd)->status = (val)

Macro to set FD STATUS field.

3.7.4 Enumeration Type Documentation

3.7.4.1 enum dpaa_sw_portal_t

DPAA SW Portals Enumeration.

3.7.4.2 enum rx_store_response_t

Possible RxStore callback responses.

Enumerator

E_RX_STORE_RESPONSE_PAUSE Pause invoking callback with received data; in polling mode, start again invoking callback only next time user invokes the receive routine; in interrupt mode, start again invoking callback only next time a receive event triggers an interrupt; in all cases,

received data that are pending are not lost, rather, their processing is temporarily deferred; in all cases, received data are processed in the order in which they were received.

E_RX_STORE_RESPONSE_CONTINUE Continue invoking callback with received data.

3.7.4.3 enum dpaa_dc_portal_t

DPAA Direct Connect Portals Enumeration.

3.7.5 Variable Documentation

3.7.5.1 volatile uint32_t dpaa_fd_t::id

FD id.

3.7.5.2 volatile uint32_t dpaa_fd_t::addrl

Data Address.

3.7.5.3 volatile uint32_t dpaa_fd_t::length

Frame length.

3.7.5.4 volatile uint32_t dpaa_fd_t::status

FD status.

3.7.5.5 uint8_t qm_revision_info_t::major_rev

Major revision.

3.7.5.6 uint8_t qm_revision_info_t::minor_rev

Minor revision.

3.7.5.7 qm_rejection_code_t qm_rejected_frame_info_t::rejection_code

Rejection code.

3.7.5.8 uint8_t qm_rejected_frame_info_t::cg_id

congestion group id

3.7.5.9 uint8_t { ... } ::cg_id

congestion group id

3.7.5.10 struct { ... } qm_rejected_frame_info_t::cg

rejection parameters when rejectionCode = e_QM_RC_CG_TAILDROP or e_QM_RC_CG_WRED.

3.7.5.11 struct { ... } ::cg

rejection parameters when rejectionCode = e_QM_RC_CG_TAILDROP or e_QM_RC_CG_WRED.

3.7.5.12 uint32_t qm_context_a_t::res[2]

reserved size for context-a

3.7.5.13 uint8_t qm_param_t::guest_id

QM Partition Id.

3.7.5.14 uintptr_t qm_param_t::sw_portals_base_address

QM Software Portals Base Address (virtual)

3.7.5.15 uint16_t qm_param_t::liodn

This value is attached to every transaction initiated by QMan when accessing its private data structures.

3.7.5.16 uint32_t qm_param_t::total_num_of_fqids

Total number of frame-queue-ids in the system.

3.7.5.17 os_mem_type qm_param_t::fqd_mem_partition_id

FQD’s mem partition id; NOTE: The memory partition must be non-cacheable and no-coherent area.

3.7.5.18 os_mem_type qm_param_t::pfdr_mem_partition_id

PFDR's mem partition id; NOTE: The memory partition must be non-cacheable and no-coherent area.

3.7.5.19 qm_exceptions_callback_t* qm_param_t::f_exception

An application callback routine to handle exceptions.

3.7.5.20 void* qm_param_t::h_app

A handle to an application layer object; This handle will be passed by the driver upon calling the above callbacks.

3.7.5.21 os_hwi_handle qm_param_t::err_irq

error interrupt line; NO_IRQ if interrupts not used

3.7.5.22 uint32_t qm_param_t::part_fqid_base

The first frame-queue-id dedicated to this partition.

NOTE: this parameter relevant only when working with multiple partitions.

3.7.5.23 uint32_t qm_param_t::part_num_of_fqids

Number of frame-queue-ids dedicated to this partition.

NOTE: this parameter relevant only when working with multiple partitions.

3.7.5.24 uint16_t qm_param_t::part_cgs_base

The first cgr dedicated to this partition.

NOTE: this parameter relevant only when working with multiple partitions.

3.7.5.25 uint16_t qm_param_t::part_num_of_cgs

Number of cgr's dedicated to this partition.

NOTE: this parameter relevant only when working with multiple partitions.

3.7.5.26 uint8_t qm_portal_stash_param_t::stash_dest_queue

This value is used to direct all stashing transactions initiated on behalf of this software portal to the specific Stashing Request Queues (SRQ)

3.7.5.27 uint8_t qm_portal_stash_param_t::eqcr

If 0, disabled.

If 1, for every EQCR entry consumed by QMan a new stash transaction is performed. If 2-7, after 2-7 EQCR entries being consumed by QMAN a new stash transaction is performed.

3.7.5.28 bool qm_portal_stash_param_t::eqcr_high_pri

EQCR entry stash transactions for this software portal will be signaled with higher priority.

3.7.5.29 bool qm_portal_stash_param_t::dqrr

DQRR entry stash enable/disable.

3.7.5.30 uint16_t qm_portal_stash_param_t::dqrr_liodn

This value is attached to every transaction initiated by QMan when performing DQRR entry or EQCR_CI stashing on behalf of this software portal.

3.7.5.31 bool qm_portal_stash_param_t::dqrr_high_pri

DQRR entry stash transactions for this software portal will be signaled with higher priority.

3.7.5.32 bool qm_portal_stash_param_t::fd_fq

Dequeued Frame Data, Annotation, and FQ Context Stashing enable/disable.

3.7.5.33 uint16_t qm_portal_stash_param_t::fd_fq_liodn

This value is attached to every transaction initiated by QMan when performing dequeued frame data and annotation stashing, or FQ context stashing on behalf of this software portal.

3.7.5.34 bool qm_portal_stash_param_t::fd_fq_high_pri

Dequeued frame data, annotation, and FQ context stash transactions for this software portal will be signaled with higher priority.

3.7.5.35 bool qm_portal_stash_param_t::fd_fq_drop

If True, Dequeued frame data, annotation, and FQ context stash transactions for this software portal will be dropped by QMan if the target SRQ is almost full, to prevent QMan sequencer stalling.

Stash transactions that are dropped will result in a fetch from main memory when a core reads the addressed coherency granule. If FALSE, Dequeued frame data, annotation, and FQ context stash transactions for this software portal will never be dropped by QMan. If the target SRQ is full a sequencer will stall until each stash transaction can be completed.

3.7.5.36 uintptr_t qm_portal_param_t::ce_base_address

Cache-enabled base address (virtual)

3.7.5.37 uintptr_t qm_portal_param_t::ci_base_address

Cache-inhibited base address (virtual)

3.7.5.38 void* qm_portal_param_t::h_qm

Qm Handle.

3.7.5.39 dpaa_sw_portal_t qm_portal_param_t::sw_portal_id

Portal id.

3.7.5.40 os_hwi_handle qm_portal_param_t::irq

portal interrupt line; used only if useIrq set to TRUE

3.7.5.41 uint16_t qm_portal_param_t::fd_liodn_offset

liodn to be used for all frames enqueued via this software portal

3.7.5.42 qm_received_frame_callback_t* qm_portal_param_t::f_dflt_frame

this callback will be called unless specific callback assigned to the FQ

3.7.5.43 qm_rejected_frame_callback_t* qm_portal_param_t::f_rejected_frame

this callback will be called for rejected frames.

3.7.5.44 void* qm_portal_param_t::h_app

a handle to the upper layer; It will be passed by the driver upon calling the CB

3.7.5.45 void* qm_fqr_congestion_avoidance_params_t::h_qm_cg

A handle to the congestion group.

3.7.5.46 int8_t qm_fqr_congestion_avoidance_params_t::overhead_accounting_length

For each frame add this number for CG calculation (may be negative), if 0 - disable feature.

3.7.5.47 uint32_t qm_fqr_congestion_avoidance_params_t::fq_tail_drop_threshold

if not "0" - enable tail drop on this FQR

3.7.5.48 void* qm_fqr_params_t::h_qm

A handle to a QM module.

3.7.5.49 void* qm_fqr_params_t::h_qm_portal

A handle to a QM Portal Module; will be used only for Create/Free/Modify routines; NOTE : if NULL, assuming affinity.

3.7.5.50 bool qm_fqr_params_t::init_parked

This FQ-Range will be initialize in park state (un-schedule)

3.7.5.51 bool qm_fqr_params_t::hold_active

This FQ-Range can be parked (un-schedule); This affects only on queues destined to software portals.

3.7.5.52 bool qm_fqr_params_t::avoid_blocking

This FQ-Range will be allow to be suspended from a software portal, and re-scheduled to another software portal servicing the same pool channel, when the DQRR of the first portal becomes full; This affects only on queues destined to software portals via pool channel; NOTE: 'holdActive' must be set to FALSE.

3.7.5.53 bool qm_fqr_params_t::prefer_in_cache

Prefer this FQ-Range to be in QMAN's internal cache for all states.

3.7.5.54 bool qm_fqr_params_t::use_context_a_for_stash

This FQ-Range will use context A for stash.

3.7.5.55 uint8_t qm_fqr_params_t::frame_annotation_size

Size of Frame Annotation to be stashed.

3.7.5.56 uint8_t { ... } ::frame_annotation_size

Size of Frame Annotation to be stashed.

3.7.5.57 uint8_t qm_fqr_params_t::frame_data_size

Size of Frame Data to be stashed.

3.7.5.58 uint8_t { ... } ::frame_data_size

Size of Frame Data to be stashed.

3.7.5.59 uint8_t qm_fqr_params_t::fq_context_size

Size of FQ context to be stashed.

3.7.5.60 uint8_t { ... } ::fq_context_size

Size of FQ context to be stashed.

3.7.5.61 uint64_t qm_fqr_params_t::fq_context_addr

40 bit memory address containing the FQ context information to be stashed; Must be cacheline-aligned

3.7.5.62 uint64_t { ... } ::fq_context_addr

40 bit memory address containing the FQ context information to be stashed; Must be cacheline-aligned

3.7.5.63 qm_context_a_t* qm_fqr_params_t::context_a

context-A field to be written in the FQ structure

3.7.5.64 qm_context_a_t* { ... } ::context_a

context-A field to be written in the FQ structure

3.7.5.65 qm_context_b_t* qm_fqr_params_t::context_b

context-B field to be written in the FQ structure; Note that this field may be used for Tx queues only!

3.7.5.66 qm_fq_channel_t qm_fqr_params_t::channel

Qm Channel.

3.7.5.67 uint8_t qm_fqr_params_t::wq

Work queue within the channel.

3.7.5.68 bool qm_fqr_params_t::shadow_mode

If TRUE, useForce MUST set to TRUE and numOfFqids MUST set to '1'.

3.7.5.69 uint32_t qm_fqr_params_t::num_of_fqids

number of fqids to be allocated

3.7.5.70 bool qm_fqr_params_t::use_force

TRUE - force allocation of specific fqids; FALSE - allocate several fqids.

3.7.5.71 uint32_t qm_fqr_params_t::align

The first FQID alignment desired.

will be used if useForce=FALSE

3.7.5.72 uint32_t { ... } ::align

The first FQID alignment desired.

will be used if useForce=FALSE

3.7.5.73 uint32_t qm_fqr_params_t::fqid

the fqid base of the forced fqids.

will be used if useForce=TRUE

3.7.5.74 uint32_t { ... } ::fqid

the fqid base of the forced fqids.

will be used if useForce=TRUE

3.7.5.75 bool qm_fqr_params_t::congestion_avoidance_enable

TRUE to enable congestion avoidance mechanism.

3.7.5.76 qm_fqr_congestion_avoidance_params_t qm_fqr_params_t::congestion_↩ avoidance_params

Parameters for congestion avoidance.

3.7.5.77 qm_fqr_mod_opt_t qm_fqr_modify_params_t::mod_options

the flags that represent the modification that are required to be done for the FQR

3.7.5.78 bool qm_fqr_modify_params_t::hold_active

This option maybe used when 'modOptions' set with QM_FQR_MOD_OPT_HOLD_ACTIVE; This F↩ Q-Range can be parked (un-schedule); This affects only on queues destined to software portals.

3.7.5.79 bool qm_fqr_modify_params_t::avoid_blocking

This option maybe used when 'modOptions' set with QM_FQR_MOD_OPT_AVOID_BLOCKING; This FQ-Range will be allow to be suspended from a software portal, and re-scheduled to another software portal servicing the same pool channel, when the DQRR of the first portal becomes full; This affects only on queues destined to software portals via pool channel; NOTE: 'holdActive' must be FALSE.

3.7.5.80 bool qm_fqr_modify_params_t::prefer_in_cache

Prefer this FQ-Range to be in QMAN's internal cache for all states.

3.7.5.81 bool qm_fqr_modify_params_t::use_context_a_for_stash

This FQ-Range will use context A for stash.

3.7.5.82 uint8_t qm_fqr_modify_params_t::frame_annotation_size

Size of Frame Annotation to be stashed.

3.7.5.83 uint8_t { ... } ::frame_annotation_size

Size of Frame Annotation to be stashed.

3.7.5.84 uint8_t qm_fqr_modify_params_t::frame_data_size

Size of Frame Data to be stashed.

3.7.5.85 uint8_t { ... } ::frame_data_size

Size of Frame Data to be stashed.

3.7.5.86 uint8_t qm_fqr_modify_params_t::fq_context_size

Size of FQ context to be stashed.

3.7.5.87 uint8_t { ... } ::fq_context_size

Size of FQ context to be stashed.

3.7.5.88 uint64_t qm_fqr_modify_params_t::fq_context_addr

40 bit memory address containing the FQ context information to be stashed; Must be cacheline-aligned

3.7.5.89 uint64_t { ... } ::fq_context_addr

40 bit memory address containing the FQ context information to be stashed; Must be cacheline-aligned

3.7.5.90 qm_context_a_t qm_fqr_modify_params_t::context_a

context-A field to be written in the FQ structure

3.7.5.91 qm_context_a_t { ... } ::context_a

context-A field to be written in the FQ structure

3.7.5.92 qm_context_b_t qm_fqr_modify_params_t::context_b

context-B field to be written in the FQ structure; Note that this field may be used for Tx queues only!

3.7.5.93 qm_fq_channel_t qm_fqr_modify_params_t::channel

Qm Channel.

3.7.5.94 uint8_t qm_fqr_modify_params_t::wq

Work queue within the channel.

3.7.5.95 bool qm_fqr_modify_params_t::congestion_avoidance_enable

This option maybe used when 'modOptions' set with DPAPORT_Q_MOD_OPT_QM_CG; TRUE to enable congestion avoidance mechanism.

3.7.5.96 qm_fqr_congestion_avoidance_params_t qm_fqr_modify_params_t↩ ::congestion_avoidance_params

Parameters for congestion avoidance.

3.7.5.97 qm_fqr_drained_completion_cb_t* qm_fqr_modify_params_t::f_completion_cb

Pointer to a completion callback to be used in non-blocking mode; NOTE: If interrupts are disabled (i.e. polling mode) than f_completion_cb should be passed and user should call the portal poll function. If interrupts are enabled than f_completion_cb should be NULL.

3.7.5.98 bool qm_fqr_modify_params_t::deliver_frame

TRUE for deliver the drained frames to the user; FALSE for not deliver the frames.

3.7.5.99 qm_received_frame_callback_t* qm_fqr_modify_params_t::f_callback

Pointer to a callback to handle the delivered frames.

3.7.5.100 void* qm_fqr_modify_params_t::h_app

User's application descriptor.

3.7.5.101 uint32_t qm_cg_wred_curve_t::max_th

minimum threshold - below this level all packets are rejected (approximated to be expressed as x*2^y due to HW implementation)

3.7.5.102 uint32_t qm_cg_wred_curve_t::min_th

minimum threshold - below this level all packets are accepted (approximated due to HW implementation)

3.7.5.103 uint8_t qm_cg_wred_curve_t::probability_denominator

1-64, the fraction of packets dropped when the average queue depth is at the maximum threshold. (approximated due to HW implementation).

3.7.5.104 void* qm_cg_params_t::h_qm

A handle to a QM module.

3.7.5.105 void* qm_cg_params_t::h_qm_portal

A handle to a QM Portal Module; will be used for Init, Free and as an interrupt destination for cg state change (if CgStateChangeEnable = TRUE)

3.7.5.106 bool qm_cg_params_t::frame_count

TRUE for frame count, FALSE - byte count.

3.7.5.107 bool qm_cg_params_t::wred_enable

if TRUE - WRED enabled.

Each color is enabled independently so that some colors may use WRED, but others may use Tail drop - if enabled, or none.

3.7.5.108 qm_cg_wred_params_t qm_cg_params_t::wred_params

WRED parameters, relevant if wredEnable = TRUE.

3.7.5.109 bool qm_cg_params_t::tail_drop_enable

if TRUE - Tail drop enabled

3.7.5.110 uint32_t qm_cg_params_t::threshold

If Tail drop - used as Tail drop threshold, otherwise 'threshold' may still be used to receive notifications when threshold is passed.

If threshold and f_Exception are set, interrupts are automatically set by driver.

3.7.5.111 bool qm_cg_params_t::notify_dc_portal

Relevant if this CG receives enqueues from a DC portal (NOTE only FMan portals!).

TRUE to notify the DC portal, FALSE to notify this SW portal.

3.7.5.112 dpaa_dc_portal_t qm_cg_params_t::dc_portal_id

relevant if notifyDcPortal=TRUE - DC Portal id

3.7.5.113 qm_exceptions_callback_t* qm_cg_params_t::f_exception

relevant and mandatory if threshold is configured and notifyDcPortal = FALSE.

If threshold and f_Exception are set, interrupts are automatically set by driver

3.7.5.114 void* qm_cg_params_t::h_app

A handle to the application layer, will be passed as argument to f_Exception.

3.7.6 Buffer Manager Module API

3.7.6.1 Overview

BMAN API file.

BMAN initialize.

Modules

- Buffer Manager General
- BM-Pool API
- BMAN portal Unit

3.7.6.2 Buffer Manager General

3.7.6.2.1 Overview

BM (common) Runtime control unit API functions, definitions and enums.

BM General Initialization Unit

Modules

- Buffer Manager General Runtime
- Buffer Manager General Initialization

3.7.6.2.2 Buffer Manager General Runtime

3.7.6.2.2.1 Overview

/* /**

BM (common) Runtime control unit API functions, definitions and enums.

Enumerations

- enum bm_counters_t { BM_COUNTERS_FBPR = 0 }

Functions

- uint32_t bmCounterGet (void *h_bm, bm_counters_t counter)

3.7.6.2.2.2 Enumeration Type Documentation

3.7.6.2.2.2.1 enum bm_counters_t

enum for defining BM counters

Enumerator

BM_COUNTERS_FBPR Total Free Buffer Proxy Record (FBPR) Free Pool Count in external memory.

3.7.6.2.2.3 Function Documentation

3.7.6.2.2.3.1 uint32_t bmCounterGet (void * *h_bm,* bm_counters_t *counter*)

Reads one of the BM counters.

Parameters

in	*h_Bm*	- A handle to the BM Module.
in	*counter*	- The requested counter.

Returns

Counter's current value.

3.7.6.2.3 Buffer Manager General Initialization

3.7.6.2.3.1 Overview

BM General Initialization Unit

Data Structures

- struct bm_param_t
- struct bm_revision_info_t

Typedefs

- typedef void(bmExceptionsCallback)(void *h_app, bm_exceptions_t exception)

Enumerations

- enum bm_exceptions_t {
  E_BM_EX_MEMORY_ACCESS_INT = 0, E_BM_EX_MEMORY_CORRUPTION, E_BM_E↩
  X_INVALID_COMMAND,
  E_BM_EX_FBPR_THRESHOLD, E_BM_EX_SINGLE_ECC, E_BM_EX_MULTI_ECC,
  E_BM_EX_POOLS_AVAIL_STATE }

Functions

- void ∗ bmConfig (bm_param_t ∗bm_param)
- os_status bmInit (void ∗h_bm)
- os_status bmFree (void ∗h_bm)
- os_status bmRevisionGet (void ∗h_bm, bm_revision_info_t ∗bm_revision_info)
- void bmErrorIsr (void ∗h_bm)
- os_status bmFbprThresholdConfig (void ∗h_bm, uint32_t threshold)

3.7.6.2.3.2 Data Structure Documentation

3.7.6.2.3.2.1 struct bm_param_t

structure representing BM initialization parameters

Data Fields

- uint8_t guest_id
- uint16_t liodn
- uint32_t total_num_of_buffers
- uint32_t fbpr_mem_partition_id
- bmExceptionsCallback ∗ f_exception
- void ∗ h_app
- os_hwi_handle err_irq
- uint8_t part_bpid_base
- uint8_t part_num_of_pools

3.7.6.2.3.2.2 Field Documentation

3.7.6.2.3.2.3 uint8_t bm_param_t::guest_id

BM Partition Id.

3.7.6.2.3.2.4 uint16_t bm_param_t::liodn

This value is attached to every transaction initiated by BMan when accessing its private data structures LIODN_EXT should be used if LIODN is configured externally to DSP NOTE: this parameter relevant only for BM in master mode ('guest_id'=DSP_DPAA_MASTER_ID).

3.7.6.2.3.2.5 uint32_t bm_param_t::total_num_of_buffers

Total number of buffers NOTE: this parameter relevant only for BM in master mode ('guest_id'=DSP_↩
DPAA_MASTER_ID).

3.7.6.2.3.2.6 uint32_t bm_param_t::fbpr_mem_partition_id

FBPR's mem partition id; NOTE: The memory partition must be non-cacheable and no-coherent area.

NOTE: this parameter relevant only for BM in master mode ('guest_id'=DSP_DPAA_MASTER_ID).

3.7.6.2.3.2.7 bmExceptionsCallback* bm_param_t::f_exception

An application callback routine to handle exceptions.

NOTE: this parameter relevant only for BM in master mode ('guest_id'=DSP_DPAA_MASTER_ID).

3.7.6.2.3.2.8 void* bm_param_t::h_app

A handle to an application layer object; This handle will be passed by the driver upon calling the above callbacks.

NOTE: this parameter relevant only for BM in master mode ('guest_id'=DSP_DPAA_MASTER_ID).

3.7.6.2.3.2.9 os_hwi_handle bm_param_t::err_irq

BM error interrupt line; NO_IRQ if interrupts not used.

NOTE: this parameter relevant only for BM in master mode ('guest_id'=DSP_DPAA_MASTER_ID).

3.7.6.2.3.2.10 uint8_t bm_param_t::part_bpid_base

The first buffer-pool-id dedicated to this partition.

NOTE: this parameter relevant only when working with multiple partitions.

3.7.6.2.3.2.11 uint8_t bm_param_t::part_num_of_pools

Number of Pools dedicated to this partition.

NOTE: this parameter relevant only when working with multiple partitions.

3.7.6.2.3.2.12 struct bm_revision_info_t

structure for returning revision information

Data Fields

- uint8_t major_rev
- uint8_t minor_rev

3.7.6.2.3.2.13 Field Documentation

3.7.6.2.3.2.14 uint8_t bm_revision_info_t::major_rev

Major revision.

3.7.6.2.3.2.15 uint8_t bm_revision_info_t::minor_rev

Minor revision.

3.7.6.2.3.3 Typedef Documentation

3.7.6.2.3.3.1 typedef void(bmExceptionsCallback)(void *h_app, bm_exceptions_t exception)

Exceptions user callback routine, will be called upon an exception passing the exception identification.

Parameters

in	*h_app*	- User's application descriptor.
in	*exception*	- The exception.

3.7.6.2.3.4 Enumeration Type Documentation

3.7.6.2.3.4.1 enum bm_exceptions_t

BM Exceptions.

Enumerator

E_BM_EX_MEMORY_ACCESS_INT External memory access interrupt.
E_BM_EX_MEMORY_CORRUPTION External memory corruption interrupt.
E_BM_EX_INVALID_COMMAND Invalid Command Verb Interrupt.
E_BM_EX_FBPR_THRESHOLD FBPR Low Watermark Interrupt.
E_BM_EX_SINGLE_ECC Single Bit ECC Error Interrupt.
E_BM_EX_MULTI_ECC Multi Bit ECC Error Interrupt.
E_BM_EX_POOLS_AVAIL_STATE Buffer pool availability state change interrupt.

3.7.6.2.3.5 Function Documentation

3.7.6.2.3.5.1 void* bmConfig (bm_param_t * *bm_param*)

Creates descriptor for the BM module and initializes the BM module.

The routine returns a handle (descriptor) to the BM object. This descriptor must be passed as first parameter to all other BM function calls.

Parameters

in	*p_BmParam*	- A pointer to data structure of parameters

Returns

Handle to BM object, or NULL for Failure.

3.7.6.2.3.5.2 os_status bmInit (void * *h_bm*)

Initializes the BM module

Parameters

in	*h_Bm*	- A handle to the BM module

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following BM_Config().

3.7.6.2.3.5.3 os_status bmFree (void * *h_bm*)

Frees all resources that were assigned to BM module.

Calling this routine invalidates the descriptor.

Parameters

in	*h_Bm*	- A handle to the BM module

Returns

OS_SUCCESS on success; Error code otherwise.

3.7.6.2.3.5.4 os_status bmRevisionGet (void * *h_bm,* bm_revision_info_t * *bm_revision_info*)

Returns the BM revision

Parameters

in	*h_bm*	A handle to a BM Module.
out	*p_Bm↩ RevisionInfo*	A structure of revision information parameters.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following QM_Init().

3.7.6.2.3.5.5 void bmErrorIsr (void * *h_bm*)

BM interrupt-service-routine for errors.

Parameters

in	*h_Bm*	- A handle to the BM Module.

Warning

Allowed only following BM_Init(). NOTE: this parameter relevant only for BM in master mode (i.e. 'guest_id'=DSP_DPAA_MASTER_ID).

3.7.6.2.3.5.6 os_status bmFbprThresholdConfig (void * *h_bm,* uint32_t *threshold*)

Change the fbpr threshold from its default configuration [DEFAULT_FBPR_THRESHOLD]. An interrupt if enables is asserted when the number of FBPRs is below this threshold. NOTE: this parameter relevant only for BM in master mode ('guest_id'=DSP_DPAA_MASTER_ID).

Parameters

in	*h_Bm*	- A handle to the BM module
in	*threshold*	- threshold value.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following BM_Config() and before BM_Init().

3.7.6.3 BM-Pool API

3.7.6.3.1 Overview

BM-Pool API functions, definitions and enums.

BM-Pool Unit

Modules

- BM-Pool Runtime Control Unit
- BM-Pool Initialization Unit

3.7.6.3.2 BM-Pool Runtime Control Unit

3.7.6.3.2.1 Overview

BM-Pool Runtime unit API functions, definitions and enums.

Enumerations

- enum bm_pool_counters_t { BM_POOL_COUNTERS_CONTENT = 0, BM_POOL_COUNTERS_SW_DEPLETION, BM_POOL_COUNTERS_HW_DEPLETION }

Functions

- uint8_t bmPoolIdGet (void *bm_pool)
- void * bmPoolBufferContextGet (void *bm_pool, void *buff)
- void * bmPoolPhysToVirt (void *bm_pool, os_phys_ptr addr)
- os_phys_ptr bmPoolVirtToPhys (void *bm_pool, void *buff)
- uint32_t bmPOOLCounterGet (void *bm_pool, bm_pool_counters_t counter)
- void * bmPoolBufGet (void *bm_pool, void *bm_portal)
- os_status bmPoolBufPut (void *bm_pool, void *bm_portal, void *buff)

3.7.6.3.2.2 Enumeration Type Documentation

3.7.6.3.2.2.1 enum bm_pool_counters_t

enum for defining BM Pool counters

Enumerator

BM_POOL_COUNTERS_CONTENT number of free buffers for a particular pool
BM_POOL_COUNTERS_SW_DEPLETION number of times pool entered sw depletion
BM_POOL_COUNTERS_HW_DEPLETION number of times pool entered hw depletion

3.7.6.3.2.3 Function Documentation

3.7.6.3.2.3.1 uint8_t bmPoolIdGet (void * *bm_pool*)

return a buffer pool id.

Parameters

in	*h_BmPool*	- A handle to a BM-pool

Returns

Pool ID.

3.7.6.3.2.3.2 void* bmPoolBufferContextGet (void * *bm_pool,* void * *buff*)

Returns the user's private context that should be associated with the buffer.

Parameters

in	*h_BmPool*	- A handle to a BM-pool
in	*p_Buff*	- A Pointer to the buffer

Returns

user's private context.

3.7.6.3.2.3.3 void* bmPoolPhysToVirt (void * *bm_pool,* os_phys_ptr *addr*)

Translates a physical address to the matching virtual address.

Parameters

in	*h_BmPool*	- A handle to a BM-pool
in	*addr*	- The physical address to translate

Returns

Virtual address.

3.7.6.3.2.3.4 os_phys_ptr bmPoolVirtToPhys (void * *bm_pool,* void * *buff*)

Translates a virtual address to the matching physical address.

Parameters

in	*h_BmPool*	- A handle to a BM-pool
in	*addr*	- The virtual address to translate

Returns

Physical address.

3.7.6.3.2.3.5 uint32_t bmPOOLCounterGet (void * *bm_pool,* bm_pool_counters_t *counter*)

Reads one of the BM Pool counters.

Parameters

in	*h_BmPool*	- A handle to a BM-pool
in	*counter*	- The requested counter.

Returns

Counter's current value.

3.7.6.3.2.3.6 void* bmPoolBufGet (void * *bm_pool,* void * *bm_portal*)

Allocate buffer from a buffer pool.

Parameters

in	*h_BmPool*	- A handle to a BM-pool
in	*h_BmPortal*	- A handle to a BM Portal Module; NOTE : if NULL, assuming affinity.

Returns

A Pointer to the allocated buffer.

3.7.6.3.2.3.7 os_status bmPoolBufPut (void * *bm_pool,* void * *bm_portal,* void * *buff*)

Deallocate buffer to a buffer pool.

Parameters

in	*h_BmPool*	- A handle to a BM-pool
in	*h_BmPortal*	- A handle to a BM Portal Module; NOTE : if NULL, assuming affinity.
in	*p_Buff*	- A Pointer to the buffer.

Returns

OS_SUCCESS on success; Error code otherwise.

3.7.6.3.3 BM-Pool Initialization Unit

3.7.6.3.3.1 Overview

```
BM-Pool Initialization Unit
```

Data Structures

- struct buffer_pool_info_t
- struct bm_pool_param_t

Typedefs

- typedef void ∗ phys_to_virt_t (os_phys_ptr addr)
- typedef os_phys_ptr virt_to_phys_t (void ∗addr)
- typedef void(bm_depletion_callback_t)(void ∗h_app, bool in)

Functions

- void ∗ bmPoolConfig (bm_pool_param_t ∗bm_pool_param)
- os_status bmPoolInit (void ∗bm_pool)
- os_status bmPoolBpidConfig (void ∗bm_pool, uint8_t bpid)
- os_status bmPoolDepletionConfig (void ∗bm_pool, bm_depletion_callback_t ∗depletion, uint32_t thresholds[MAX_DEPLETION_THRESHOLDS])
- os_status bmPoolStockpileConfig (void ∗bm_pool, uint16_t max_buffers, uint16_t min_buffers)
- os_status bmPoolBuffContextModeConfig (void ∗bm_pool, bool en)
- os_status bmPoolFree (void ∗bm_pool)

BM Pool Depletion Thresholds macros

```
The thresholds are represent by an array of size MAX_DEPLETION_THRESHOLDS
Use the following macros to access the appropriate location in the array.
```

- #define BM_POOL_DEP_THRESH_SW_ENTRY 0
- #define BM_POOL_DEP_THRESH_SW_EXIT 1
- #define BM_POOL_DEP_THRESH_HW_ENTRY 2
- #define BM_POOL_DEP_THRESH_HW_EXIT 3
- #define MAX_DEPLETION_THRESHOLDS 4

3.7.6.3.3.2 Data Structure Documentation

3.7.6.3.3.2.1 struct buffer_pool_info_t

Buffer Pool Information Structure.

Data Fields

- uint16_t buffer_size
- phys_to_virt_t * phys_to_virt
- virt_to_phys_t * virt_to_phys

3.7.6.3.3.2.2 Field Documentation

3.7.6.3.3.2.3 uint16_t buffer_pool_info_t::buffer_size

Buffer size (in bytes)

3.7.6.3.3.2.4 phys_to_virt_t* buffer_pool_info_t::phys_to_virt

User callback to translate pool buffers physical addresses to virtual addresses.

3.7.6.3.3.2.5 virt_to_phys_t* buffer_pool_info_t::virt_to_phys

User callback to translate pool buffers virtual addresses to physical addresses.

3.7.6.3.3.2.6 struct bm_pool_param_t

structure representing BM Pool initialization parameters

Data Fields

- void * bm
- void * bm_portal
- void * h_app
- bool shadow_mode
- uint8_t bpid

3.7.6.3.3.2.7 Field Documentation

3.7.6.3.3.2.8 void* bm_pool_param_t::bm

A handle to a BM Module.

3.7.6.3.3.2.9 void* bm_pool_param_t::bm_portal

A handle to a BM Portal Module.

will be used only for Init and Free routines. NOTE: if NULL, assuming affinity

3.7.6.3.3.2.10 void* bm_pool_param_t::h_app

opaque user value passed as a parameter to callbacks

3.7.6.3.3.2.11 bool bm_pool_param_t::shadow_mode

If TRUE, num_of_buffers will be set to '0'.

3.7.6.3.3.2.12 uint8_t bm_pool_param_t::bpid

index of the shadow buffer pool (0-BM_MAX_NUM_OF_POOLS).

valid only if shadow_mode='TRUE'.

3.7.6.3.3.3 Macro Definition Documentation

3.7.6.3.3.3.1 #define BM_POOL_DEP_THRESH_SW_ENTRY 0

BMAN pool software depletion entry.

3.7.6.3.3.3.2 #define BM_POOL_DEP_THRESH_SW_EXIT 1

BMAN pool software depletion exit.

3.7.6.3.3.3.3 #define BM_POOL_DEP_THRESH_HW_ENTRY 2

BMAN pool hardware depletion entry.

3.7.6.3.3.3.4 #define BM_POOL_DEP_THRESH_HW_EXIT 3

BMAN pool hardware depletion exit.

3.7.6.3.3.3.5 #define MAX_DEPLETION_THRESHOLDS 4

depletion array size

3.7.6.3.3.4 Typedef Documentation

3.7.6.3.3.4.1 typedef void* phys_to_virt_t(os_phys_ptr addr)

Translates a physical address to the matching virtual address.

Parameters

in	*addr*	- The physical address to translate.

Returns

Virtual address.

3.7.6.3.3.4.2 typedef os_phys_ptr virt_to_phys_t(void *addr)

Translates a virtual address to the matching physical address.

Parameters

in	*addr*	- The virtual address to translate.

Returns

Physical address.

3.7.6.3.3.4.3 typedef void(bm_depletion_callback_t)(void *h_app, bool in)

This callback type is used when handling pool depletion entry/exit.

User provides this function. Driver invokes it.

Parameters

in	*h_app*	- User's application descriptor.
in	*in*	- TRUE when entered depletion state FALSE when exit the depletion state.

3.7.6.3.3.5 Function Documentation

3.7.6.3.3.5.1 void* bmPoolConfig (bm_pool_param_t * *bm_pool_param*)

Creates descriptor for the BM Pool;

The routine returns a handle (descriptor) to the BM Pool object.

Parameters

in	*p_BmPool↩ Param*	- A pointer to data structure of parameters

Returns

Handle to BM Portal object, or NULL for Failure.

3.7.6.3.3.5.2 os_status bmPoolInit (void * *bm_pool*)

Initializes a BM-Pool module

Parameters

in	*h_BmPool*	- A handle to a BM-Pool module

Returns

OS_SUCCESS on success; Error code otherwise.

3.7.6.3.3.5.3 os_status bmPoolBpidConfig (void * *bm_pool,* uint8_t *bpid*)

Config a specific pool id rather than dynamic pool id.

Parameters

in	*h_BmPool*	- A handle to a BM-Pool module
in	*bpid*	- index of the buffer pool (0-BM_MAX_NUM_OF_POOLS).

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following BM_POOL_Config() and before BM_POOL_Init().

3.7.6.3.3.5.4 os_status bmPoolDepletionConfig (void * *bm_pool,* bm_depletion_callback_t * *depletion,* uint32_t *thresholds[MAX_DEPLETION_THRESHOLDS]*)

Config depletion-entry/exit thresholds and callback.

Parameters

in	*h_BmPool*	- A handle to a BM-Pool module
in	*f_depletion*	- depletion-entry/exit callback.
in	*thresholds*	- depletion-entry/exit thresholds.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following BM_POOL_Config() and before BM_POOL_Init(); Allowed only if shadow_mode='FALSE'. Allowed only if BM in master mode ('guest_id'=DSP_DPAA_MAS↩ TER_ID), or the BM is in guest mode BUT than this routine will invoke IPC call to the master.

3.7.6.3.3.5.5 os_status bmPoolStockpileConfig (void * *bm_pool,* uint16_t *max_buffers,* uint16_t *min_buffers*)

Config software stockpile.

Parameters

in	*h_BmPool*	- A handle to a BM-Pool module
in	*maxBuffers*	- the software data structure size saved for stockpile; when reached this value, release to hw command performed.
in	*minBuffers*	- if current capacity is equal or lower then this value, acquire from hw command performed.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following BM_POOL_Config() and before BM_POOL_Init().

3.7.6.3.3.5.6 os_status bmPoolBuffContextModeConfig (void * *bm_pool,* bool *en*)

Configure the BM pool to enable/disable buffer-context; If buffer context is enabled than for each buffer the driver gets from the f_Get function it will put the context as the prefix of the pointer data. the prefix room should be larger than sizeof(void*); Calling this routine will change the driver from its default [D↩ EFAULT_BUFFER_CONTEXT].

Parameters

in	*h_BmPool*	- A handle to a BM-Pool module
in	*en*	- enable/disable buffer context mode

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following BM_POOL_Config() and before BM_POOL_Init().

3.7.6.3.3.5.7 os_status bmPoolFree (void * *bm_pool*)

Frees all resources that were assigned to BM Pool module.

Calling this routine invalidates the descriptor.

Parameters

in	*h_BmPool*	- BM Pool module descriptor

Returns

OS_SUCCESS on success; Error code otherwise.

3.7.6.4 BMAN portal Unit

3.7.6.4.1 Overview

BMAN portal API.

Modules

- BM-Portal Initialization Unit

3.7.6.4.2 BM-Portal Initialization Unit

3.7.6.4.2.1 Overview

BM-Portal Initialization Unit

Data Structures

- struct bm_portal_params_t

Functions

- void ∗ bmPortalConfig (bm_portal_params_t ∗bm_portal_param)
- os_status bmPortalInit (void ∗bm_portal)
- os_status bmPortalFree (void ∗bm_portal)
- os_status bmPortalConfigMemAttr (void ∗bm_portal, uint32_t hw_ext_structs_mem_attr)

3.7.6.4.2.2 Data Structure Documentation

3.7.6.4.2.2.1 struct bm_portal_params_t

BMAN global control flags.

structure representing BM Portal initialization parameters

Data Fields

- uintptr_t ce_base_address
- uintptr_t ci_base_address
- void ∗ h_bm
- dpaa_sw_portal_t sw_portal_id
- os_hwi_handle irq

3.7.6.4.2.2.2 Field Documentation

3.7.6.4.2.2.3 uintptr_t bm_portal_params_t::ce_base_address

Cache-enabled base address (virtual)

3.7.6.4.2.2.4 uintptr_t bm_portal_params_t::ci_base_address

Cache-inhibited base address (virtual)

3.7.6.4.2.2.5 void* bm_portal_params_t::h_bm

Bm Handle.

3.7.6.4.2.2.6 dpaa_sw_portal_t bm_portal_params_t::sw_portal_id

Portal id.

3.7.6.4.2.2.7 os_hwi_handle bm_portal_params_t::irq

portal interrupt line; NO_IRQ if interrupts not used

3.7.6.4.2.3 Function Documentation

3.7.6.4.2.3.1 void* bmPortalConfig (bm_portal_params_t * *bm_portal_param*)

Creates descriptor for the BM Portal;

The routine returns a handle (descriptor) to a BM-Portal object; This descriptor must be passed as first parameter to all other BM-Portal function calls. No actual initialization or configuration of BM-Portal hardware is done by this routine.

Parameters

in	*p_BmPortal↩ Param*	- Pointer to data structure of parameters

Return values

Handle	to a BM-Portal object, or NULL for Failure.

3.7.6.4.2.3.2 os_status bmPortalInit (void * *bm_portal*)

Initializes a BM-Portal module

Parameters

in	*bm_portal*	- A handle to a BM-Portal module

Returns

OS_SUCCESS on success; Error code otherwise.

3.7.6.4.2.3.3 os_status bmPortalFree (void * *bm_portal*)

Frees all resources that were assigned to BM Portal module.

Calling this routine invalidates the descriptor.

Parameters

in	*bm_portal*	- BM Portal module descriptor

Returns

OS_SUCCESS on success; Error code otherwise.

3.7.6.4.2.3.4 os_status bmPortalConfigMemAttr (void * *bm_portal*, uint32_t *hw_ext_structs_mem_attr*)

Change the memory attributes from its default configuration [DEFAULT_MEM_ATTR].

Parameters

in	*bm_portal*	- A handle to a BM-Portal module
in	*hw_ext_↩ structs_mem_↩ attr*	- memory attributes (cache/non-cache, etc.)

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following bmPortalConfig() and before bmPortalInit().

3.7.7 Queue Manager Module API

3.7.7.1 Overview

QM API functions, definitions and enums.

Modules

- QM General API
- QM-Portal API
- QM Frame-Queue-Range API
- QM Congestion Group API

3.7.7.2 QM General API

3.7.7.2.1 Overview

QM common API functions, definitions and enums.

QM general API functions, definitions and enums.

Modules

- QM General Initialization
- QM General Runtime

3.7.7.2.2 QM General Initialization

3.7.7.2.2.1 Overview

QM General Initialization

Data Structures

- struct qm_dc_portal_params_t
- struct qm_revision_info_t
- struct qm_rejected_frame_info_t
- struct qm_context_a_t
- struct qm_param_t

Typedefs

- typedef void(qm_exceptions_callback_t)(void *h_app, qm_exceptions_t exception)
- typedef rx_store_response_t(qm_received_frame_callback_t)(void *h_app, void *qm_fqr, void *qm_portal, uint32_t fqid_offset, dpaa_fd_t *frame)
- typedef os_status(qm_fqr_drained_completion_cb_t)(void *h_app, void *qm_fqr)

- typedef rx_store_response_t(qm_rejected_frame_callback_t)(void *h_app, void *qm_fqr, void *qm_portal, uint32_t fqid_offset, dpaa_fd_t *frame, qm_rejected_frame_info_t *qm_rejected_frame_info)
- typedef uint32_t qm_context_b_t

Enumerations

- enum qm_rejection_code_t { , E_QM_RC_CG_TAILDROP, E_QM_RC_CG_WRED, E_QM_RC_FQ_TAILDROP }
- enum qm_exceptions_t { E_QM_EX_CORENET_INITIATOR_DATA = 0, E_QM_EX_CORENET_TARGET_DATA, E_QM_EX_CORENET_INVALID_TARGET_TRANSACTION, E_QM_EX_PFDR_THRESHOLD, E_QM_EX_PFDR_ENQUEUE_BLOCKED, E_QM_EX_SINGLE_ECC, E_QM_EX_MULTI_ECC, E_QM_EX_INVALID_COMMAND, E_QM_EX_DEQUEUE_DCP, E_QM_EX_DEQUEUE_FQ, E_QM_EX_DEQUEUE_SOURCE, E_QM_EX_DEQUEUE_QUEUE, E_QM_EX_ENQUEUE_OVERFLOW, E_QM_EX_ENQUEUE_STATE, E_QM_EX_ENQUEUE_CHANNEL, E_QM_EX_ENQUEUE_QUEUE, E_QM_EX_CG_STATE_CHANGE }

Functions

- void * qmConfig (qm_param_t *qm_param)
- os_status qmInit (void *h_qm)
- os_status qmFree (void *h_qm)
- os_status qmConfigRTFramesDepth (void *h_qm, uint32_t rt_frames_depth)
- os_status qmConfigPfdrThreshold (void *h_qm, uint32_t threshold)
- os_status qmConfigSfdrReservationThreshold (void *h_qm, uint32_t threshold)

3.7.7.2.2.2 Data Structure Documentation

3.7.7.2.2.2.1 struct qm_dc_portal_params_t

structure for defining DC portal ERN destination

3.7.7.2.2.2.2 struct qm_revision_info_t

structure for returning revision information

Data Fields

- uint8_t major_rev
- uint8_t minor_rev

3.7.7.2.2.2.3 struct qm_rejected_frame_info_t

QM Rejected frame information.

Data Fields

- qm_rejection_code_t rejection_code

3.7.7.2.2.2.4 struct qm_context_a_t

structure representing QM contextA of FQ initialization parameters Note that this is only "space-holder" for the Context-A.

The "real" Context-A is described in each specific driver (E.g. FM driver has its own Context-A API).

Data Fields

- uint32_t res [2]

3.7.7.2.2.2.5 struct qm_param_t

structure representing QM initialization parameters

Data Fields

- uint8_t guest_id
- uintptr_t sw_portals_base_address
- uint16_t liodn
- uint32_t total_num_of_fqids
- os_mem_type fqd_mem_partition_id
- os_mem_type pfdr_mem_partition_id
- qm_exceptions_callback_t * f_exception
- void * h_app
- os_hwi_handle err_irq
- uint32_t part_fqid_base
- uint32_t part_num_of_fqids
- uint16_t part_cgs_base
- uint16_t part_num_of_cgs

3.7.7.2.2.3 Typedef Documentation

3.7.7.2.2.3.1 typedef void(qm_exceptions_callback_t)(void *h_app, qm_exceptions_t exception)

Exceptions user callback routine, will be called upon an exception passing the exception identification.

Parameters

in	*h_app*	- User's application descriptor.
in	*exception*	- The exception.

3.7.7.2.2.3.2 typedef rx_store_response_t(qm_received_frame_callback_t)(void *h_app, void *qm_fqr, void *qm_portal, uint32_t fqid_offset, dpaa_fd_t *frame)

This callback type is used when receiving frame.

User provides this function. Driver invokes it.

Parameters

in	*h_app*	A user argument to the callback
in	*h_qm_fqr*	A handle to an QM-FQR Module.
in	*fqid_offset*	fqid offset from the FQR's fqid base.
in	*frame*	The Received Frame

Return values

e_RX_STORE_RESPONSE_CONTINUE	- order the driver to continue Rx operation for all ready data.
e_RX_STORE_RESPONSE_PAUSE	- order the driver to stop Rx operation.

Warning

frame is local parameter; i.e. users must NOT access or use this parameter in any means outside this callback context.

3.7.7.2.2.3.3 typedef os_status(qm_fqr_drained_completion_cb_t)(void *h_app, void *qm_fqr)

This callback type is used when the FQR is completely was drained.

User provides this function. Driver invokes it.

Parameters

in	*h_app*	A user argument to the callback
in	*h_qm_fqr*	A handle to an QM-FQR Module.

Return values

OS_SUCCESS	on success; Error code otherwise.

3.7.7.2.2.3.4 typedef rx_store_response_t(qm_rejected_frame_callback_t)(void *h_app, void *qm_fqr, void *qm_portal, uint32_t fqid_offset, dpaa_fd_t *frame, qm_rejected_frame_info_t *qm_rejected_frame_info)

This callback type is used when receiving rejected frames.

User provides this function. Driver invokes it.

Parameters

in	*h_app*	A user argument to the callback

in	*h_qm_fqr*	A handle to an QM-FQR Module.
in	*fqid_offset*	fqid offset from the FQR's fqid base.
in	*frame*	The Rejected Frame
in	*qm_rejected_frame_info*	Rejected Frame information

Return values

e_RX_STORE_RESPONSE_CONTINUE	- order the driver to continue Rx operation for all ready data.
e_RX_STORE_RESPONSE_PAUSE	- order the driver to stop Rx operation.

Warning

frame is local parameter; i.e. users must NOT access or use this parameter in any means outside this callback context.

3.7.7.2.2.3.5 typedef uint32_t qm_context_b_t

structure representing QM contextB of FQ initialization parameters Note that this is only "space-holder" for the Context-B.

The "real" Context-B is described in each specific driver (E.g. FM driver has its own Context-B API).

3.7.7.2.2.4 Enumeration Type Documentation

3.7.7.2.2.4.1 enum qm_rejection_code_t

QM Rejection code enum.

Enumerator

E_QM_RC_CG_TAILDROP This frames was rejected due to congestion group tail-drop situation.
E_QM_RC_CG_WRED This frames was rejected due to congestion group WRED situation.
E_QM_RC_FQ_TAILDROP This frames was rejected due to FQID TD situation.

3.7.7.2.2.4.2 enum qm_exceptions_t

QM Exceptions.

Enumerator

E_QM_EX_CORENET_INITIATOR_DATA Initiator Data Error.
E_QM_EX_CORENET_TARGET_DATA CoreNet Target Data Error.
E_QM_EX_CORENET_INVALID_TARGET_TRANSACTION Invalid Target Transaction.
E_QM_EX_PFDR_THRESHOLD PFDR Low Watermark Interrupt.

E_QM_EX_PFDR_ENQUEUE_BLOCKED PFDR Enqueues Blocked Interrupt.
E_QM_EX_SINGLE_ECC Single Bit ECC Error Interrupt.
E_QM_EX_MULTI_ECC Multi Bit ECC Error Interrupt.
E_QM_EX_INVALID_COMMAND Invalid Command Verb Interrupt.
E_QM_EX_DEQUEUE_DCP Invalid Dequeue Direct Connect Portal Interrupt.
E_QM_EX_DEQUEUE_FQ Invalid Dequeue FQ Interrupt.
E_QM_EX_DEQUEUE_SOURCE Invalid Dequeue Source Interrupt.
E_QM_EX_DEQUEUE_QUEUE Invalid Dequeue Queue Interrupt.
E_QM_EX_ENQUEUE_OVERFLOW Invalid Enqueue Overflow Interrupt.
E_QM_EX_ENQUEUE_STATE Invalid Enqueue State Interrupt.
E_QM_EX_ENQUEUE_CHANNEL Invalid Enqueue Channel Interrupt.
E_QM_EX_ENQUEUE_QUEUE Invalid Enqueue Queue Interrupt.
E_QM_EX_CG_STATE_CHANGE CG change state notification.

3.7.7.2.2.5 Function Documentation

3.7.7.2.2.5.1 void* qmConfig (qm_param_t * *qm_param*)

Creates descriptor for the QM module.

The routine returns a handle (descriptor) to the QM object. This descriptor must be passed as first parameter to all other QM function calls. No actual initialization or configuration of QM hardware is done by this routine.

Parameters

in	*qm_param*	- Pointer to data structure of parameters

Return values

Handle	to the QM object, or NULL for Failure.

3.7.7.2.2.5.2 os_status qmInit (void * *h_qm*)

Initializes the QM module

Parameters

in	*h_qm*	- A handle to the QM module

Returns

OS_SUCCESS on success; Error code otherwise.

3.7.7.2.2.5.3 os_status qmFree (void * *h_qm*)

Frees all resources that were assigned to the QM module.

```
Calling this routine invalidates the descriptor.
```

Parameters

in	*h_qm*	- A handle to the QM module

Returns

OS_SUCCESS on success; Error code otherwise.

3.7.7.2.2.5.4 os_status qmConfigRTFramesDepth (void * *h_qm*, uint32_t *rt_frames_depth*)

Change the run-time frames depth (i.e. the maximum total number of frames that may be inside QM at a certain time) from its default configuration [DEFAULT_rtFramesDepth].

Parameters

in	*h_qm*	- A handle to the QM module
in	*rt_frames_depth*	- run-time max num of frames.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following qmConfig() and before qmInit().

3.7.7.2.2.5.5 os_status qmConfigPfdrThreshold (void * *h_qm*, uint32_t *threshold*)

Change the pfdr threshold from its default configuration [DEFAULT_pfdrThreshold]. An interrupt if enables is asserted when the number of PFDRs is below this threshold.

Parameters

in	*h_qm*	- A handle to the QM module
in	*threshold*	- threshold value.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following qmConfig() and before qmInit().

3.7.7.2.2.5.6 os_status qmConfigSfdrReservationThreshold (void * *h_qm,* uint32_t *threshold*)

Change the sfdr threshold from its default configuration [DEFAULT_sfdrThreshold].

Parameters

in	*h_qm*	- A handle to the QM module
in	*threshold*	- threshold value.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following qmConfig() and before qmInit().

3.7.7.2.3 QM General Runtime

3.7.7.2.3.1 Overview

QM General Runtime API functions, definitions and enums.

Data Structures

- struct qm_rsrv_fqr_params_t
- struct qm_error_info_t
- struct qm_wq_class_scheduler_params_t

Enumerations

- enum qm_counters_t { E_QM_COUNTERS_SFDR_IN_USE = 0, E_QM_COUNTERS_PFDR_IN_USE, E_QM_COUNTERS_PFDR_FREE_POOL }
- enum qm_channel_type_t { E_QM_SW_CHANNEL = 0, E_QM_POOL_CHANNEL, E_QM_DC_CHANNEL }

Functions

- os_status qmReserveQueues (void *h_qm, qm_rsrv_fqr_params_t *qm_fqr_params, uint32_t *base_fqid)
- os_status qmUnreserveQueues (void *h_qm, uint32_t base_fqid)
- os_status qmSetException (void *h_qm, qm_exceptions_t exception, bool enable)
- void qmErrorIsr (void *h_qm)
- os_status qmGetErrorInformation (void *h_qm, qm_error_info_t *err_info)
- uint32_t qmGetCounter (void *h_qm, qm_counters_t counter)
- os_status qmGetRevision (void *h_qm, qm_revision_info_t *qm_revision_info)
- os_status qmSetWqClassScheduler (void *h_qm, qm_wq_class_scheduler_params_t *qm_wq_class_scheduler_params)

3.7.7.2.3.2 Data Structure Documentation

3.7.7.2.3.2.1 struct qm_rsrv_fqr_params_t

structure representing QM FQ-Range reservation parameters

Data Fields

- bool use_force
- uint32_t num_of_fqids
- union {
  - struct {
    - uint32_t align
  - } non_frc_qs
  - struct {
    - uint32_t fqid
  - } frc_q

} qs

3.7.7.2.3.2.2 Field Documentation

3.7.7.2.3.2.3 bool qm_rsrv_fqr_params_t::use_force

TRUE - force reservation of specific fqids; FALSE - reserve several fqids.

3.7.7.2.3.2.4 uint32_t qm_rsrv_fqr_params_t::num_of_fqids

Number of fqids to be reserved.

3.7.7.2.3.2.5 uint32_t qm_rsrv_fqr_params_t::align

Alignment.

will be used if useForce=FALSE

3.7.7.2.3.2.6 struct { ... } qm_rsrv_fqr_params_t::non_frc_qs

used when useForce = FALSE

3.7.7.2.3.2.7 uint32_t qm_rsrv_fqr_params_t::fqid

The fqid base of the forced fqids.

will be used if useForce=TRUE

3.7.7.2.3.2.8 struct { ... } qm_rsrv_fqr_params_t::frc_q

used when useForce = TRUE

3.7.7.2.3.2.9 union { ... } qm_rsrv_fqr_params_t::qs

force/non-force parameters

3.7.7.2.3.2.10 struct qm_error_info_t

structure representing QM Error information

Data Fields

- bool portal_valid
- bool hw_portal
- dpaa_sw_portal_t sw_portal_id
- dpaa_dc_portal_t dcp_id
- bool fqid_valid
- uint32_t fqid

3.7.7.2.3.2.11 Field Documentation

3.7.7.2.3.2.12 bool qm_error_info_t::portal_valid

TRUE - portal information is valid.

3.7.7.2.3.2.13 bool qm_error_info_t::hw_portal

TRUE - hardware portal id is valid.

3.7.7.2.3.2.14 dpaa_sw_portal_t qm_error_info_t::sw_portal_id

SW Portal ID.

3.7.7.2.3.2.15 dpaa_dc_portal_t qm_error_info_t::dcp_id

DCP (HW Portal) ID.

3.7.7.2.3.2.16 bool qm_error_info_t::fqid_valid

TRUE - fqid is valid.

3.7.7.2.3.2.17 uint32_t qm_error_info_t::fqid

the fqid that belongs to the error

3.7.7.2.3.2.18 struct qm_wq_class_scheduler_params_t

structure representing QM WQ Class-Scheduler parameters

Data Fields

- qm_channel_type_t channel_type
- dpaa_dc_portal_t dcp_id
- uint8_t low_tier_over_medium_tier_ratio
- uint8_t wq_2_weight
- uint8_t wq_3_weight
- uint8_t wq_4_weight
- uint8_t wq_5_weight
- uint8_t wq_6_weight
- uint8_t wq_7_weight

3.7.7.2.3.2.19 Field Documentation

3.7.7.2.3.2.20 qm_channel_type_t qm_wq_class_scheduler_params_t::channel_type

the channel type to be used for this WQ class

3.7.7.2.3.2.21 dpaa_dc_portal_t qm_wq_class_scheduler_params_t::dcp_id

Only valid if channelType == E_QM_DC_CHANNEL.

3.7.7.2.3.2.22 uint8_t qm_wq_class_scheduler_params_t::low_tier_over_medium_tier_ratio

Allows the low priority tier to be elevated above the medium priority tier one in lowTierOverMedium↩ TierRatio+1 times; valid range is 1-255; a value of 0 disables the elevation of the low tier.

3.7.7.2.3.2.23 uint8_t qm_wq_class_scheduler_params_t::wq_2_weight

Set weight for WQ 2; valid range is 1-8.

3.7.7.2.3.2.24 uint8_t qm_wq_class_scheduler_params_t::wq_3_weight

Set weight for WQ 3; valid range is 1-8.

3.7.7.2.3.2.25 uint8_t qm_wq_class_scheduler_params_t::wq_4_weight

Set weight for WQ 4; valid range is 1-8.

3.7.7.2.3.2.26 uint8_t qm_wq_class_scheduler_params_t::wq_5_weight

Set weight for WQ 5; valid range is 1-8.

3.7.7.2.3.2.27 uint8_t qm_wq_class_scheduler_params_t::wq_6_weight

Set weight for WQ 6; valid range is 1-8.

3.7.7.2.3.2.28 uint8_t qm_wq_class_scheduler_params_t::wq_7_weight

Set weight for WQ 7; valid range is 1-8.

3.7.7.2.3.3 Enumeration Type Documentation

3.7.7.2.3.3.1 enum qm_counters_t

enum for defining QM counters

Enumerator

E_QM_COUNTERS_SFDR_IN_USE Total Single Frame Descriptor Record (SFDR) currently in use.

E_QM_COUNTERS_PFDR_IN_USE Total Packed Frame Descriptor Record (PFDR) currently in use.

E_QM_COUNTERS_PFDR_FREE_POOL Total Packed Frame Descriptor Record (PFDR) Free Pool Count in external memory.

3.7.7.2.3.3.2 enum qm_channel_type_t

enum for defining QM WQ-CS channel type

Enumerator

E_QM_SW_CHANNEL SW-portal channel.

E_QM_POOL_CHANNEL pool channel

E_QM_DC_CHANNEL DC-portal (HW) channel.

3.7.7.2.3.4 Function Documentation

3.7.7.2.3.4.1 os_status qmReserveQueues (void * *h_qm*, qm_rsrv_fqr_params_t * *qm_fqr_params*, uint32_t * *base_fqid*)

Request to Reserved queues for future use.

Parameters

in	*h_qm*	- A handle to the QM Module.
in	*qm_fqr_params*	- A structure of parameters for defining the desired queues parameters.
out	*base_fqid*	- base-fqid on success; '0' code otherwise.

Returns

E_OK on success;

Warning

Allowed only after QM_Init().

3.7.7.2.3.4.2 os_status qmUnreserveQueues (void * *h_qm*, uint32_t *base_fqid*)

Release previously reserved queues.

Parameters

in	*h_qm*	- A handle to the QM Module.
in	*base_fqid*	- base-fqid of a group to release, the same value returned in base_fqid when called QM_ReserveQueues.

Returns

E_OK on success;

Warning

Allowed only after QM_Init().

3.7.7.2.3.4.3 os_status qmSetException (void * *h_qm*, qm_exceptions_t *exception*, bool *enable*)

Calling this routine enables/disables the specified exception.

Parameters

in	*h_qm*	- A handle to the QM Module.
in	*exception*	- The exception to be selected.
in	*enable*	- TRUE to enable interrupt, FALSE to mask it.

Returns

E_OK on success; Error code otherwise.

Warning

Allowed only following QM_Init(). This routine should NOT be called from guest-partition (i.e. guestId != NCSW_MASTER_ID)

3.7.7.2.3.4.4 void qmErrorIsr (void * *h_qm*)

QM interrupt-service-routine for errors.

Parameters

in	*h_qm*	- A handle to the QM module

Warning

Allowed only following QM_Init(). This routine should NOT be called from guest-partition (i.e. guestId != NCSW_MASTER_ID)

3.7.7.2.3.4.5 os_status qmGetErrorInformation (void * *h_qm*, qm_error_info_t * *err_info*)

Reads the last error information.

Parameters

in	*h_qm*	- A handle to the QM Module.
out	*err_info*	- Error information will be loaded to this structure.

Returns

E_OK on success; Error code otherwise.

Warning

Allowed only following QM_Init(). This routine should NOT be called from guest-partition (i.e. guestId != NCSW_MASTER_ID)

3.7.7.2.3.4.6 uint32_t qmGetCounter (void * *h_qm*, qm_counters_t *counter*)

Reads one of the QM counters.

Parameters

in	*h_qm*	- A handle to the QM Module.
in	*counter*	- The requested counter.

Returns

Counter's current value.

Warning

Allowed only following QM_Init().

3.7.7.2.3.4.7 os_status qmGetRevision (void * *h_qm*, qm_revision_info_t * *qm_revision_info*)

s

Returns the QM revision

Parameters

in	*h_qm*	A handle to a QM Module.
out	*qm_revision_info*	A structure of revision information parameters.

Returns

None.

Warning

Allowed only following QM_Init().

3.7.7.2.3.4.8 os_status qmSetWqClassScheduler (void * *h_qm*, qm_wq_class_scheduler_params_t * *qm_wq_class_scheduler_params*)

Calling this routine allow to change the weights of the WQ in a channel.

You may call this function for changing the weight of the WQs and the evaluation time of the low-priority tier over the medium-priority tier; This can be done on the fly, but because of the scheduling algorithms, it make take a few frame dequeues before the effects of the new settings are observed.

Parameters

in	*h_qm*	A handle to a QM Module.
in	*qm_wq_class↩ _scheduler_↩ params*	A structure of wq class-scheduler parameters.

Returns

E_OK on success; Error code otherwise.

Warning

Allowed only following QM_Init(). This routine should NOT be called from guest-partition (i.e. guestId != NCSW_MASTER_ID)

3.7.7.3 QM-Portal API

3.7.7.3.1 Overview

QM common API functions, definitions and enums.

Modules

- QM-Portal Initialization
- QM-Portal Runtime

3.7.7.3.2 QM-Portal Initialization

3.7.7.3.2.1 Overview

QM-Portal Initialization Unit

Data Structures

- struct qm_portal_stash_param_t
- struct qm_portal_param_t

Enumerations

- enum qm_portal_poll_source_t { E_QM_PORTAL_POLL_SOURCE_DATA_FRAMES = 0, E_↩ QM_PORTAL_POLL_SOURCE_CONTROL_FRAMES, E_QM_PORTAL_POLL_SOURCE_↩ BOTH }

Functions

- void ∗ qmPortalConfig (qm_portal_param_t ∗qm_portal_param)
- os_status qmPortalInit (void ∗h_qm_portal)
- os_status qmPortalFree (void ∗h_qm_portal)
- os_status qmPortalConfigDcaMode (void ∗h_qm_portal, bool enable)
- os_status qmPortalConfigStash (void ∗h_qm_portal, qm_portal_stash_param_t ∗stash_params)
- os_status qmPortalConfigDequeueRingSize (void ∗h_qm_portal, uint8_t dqrr_size)
- os_status qmPortalConfigPullMode (void ∗h_qm_portal, bool pull_mode)

3.7.7.3.2.2 Data Structure Documentation

3.7.7.3.2.2.1 struct qm_portal_stash_param_t

structure representing QM-Portal Stash parameters

Data Fields

- uint8_t stash_dest_queue
- uint8_t eqcr
- bool eqcr_high_pri
- bool dqrr
- uint16_t dqrr_liodn
- bool dqrr_high_pri
- bool fd_fq
- uint16_t fd_fq_liodn
- bool fd_fq_high_pri
- bool fd_fq_drop

3.7.7.3.2.2.2 struct qm_portal_param_t

structure representing QM-Portal initialization parameters

Data Fields

- uintptr_t ce_base_address
- uintptr_t ci_base_address
- void ∗ h_qm
- dpaa_sw_portal_t sw_portal_id
- os_hwi_handle irq
- uint16_t fd_liodn_offset
- qm_received_frame_callback_t ∗ f_dflt_frame
- qm_rejected_frame_callback_t ∗ f_rejected_frame
- void ∗ h_app

3.7.7.3.2.3 Enumeration Type Documentation

3.7.7.3.2.3.1 enum qm_portal_poll_source_t

Frame's Type to poll.

Enumerator

E_QM_PORTAL_POLL_SOURCE_DATA_FRAMES Poll only data frames.
E_QM_PORTAL_POLL_SOURCE_CONTROL_FRAMES Poll only control frames.

E_QM_PORTAL_POLL_SOURCE_BOTH Poll both.

3.7.7.3.2.4 Function Documentation

3.7.7.3.2.4.1 void* qmPortalConfig (qm_portal_param_t * *qm_portal_param*)

Creates descriptor for a QM-Portal module.

The routine returns a handle (descriptor) to a QM-Portal object. This descriptor must be passed as first parameter to all other QM-Portal function calls. No actual initialization or configuration of QM-Portal hardware is done by this routine.

Parameters

in	*qm_portal_param*	- Pointer to data structure of parameters

Return values

Handle	to a QM-Portal object, or NULL for Failure.

3.7.7.3.2.4.2 os_status qmPortalInit (void * *h_qm_portal*)

Initializes a QM-Portal module

Parameters

in	*h_qm_portal*	- A handle to a QM-Portal module

Returns

OS_SUCCESS on success; Error code otherwise.

3.7.7.3.2.4.3 os_status qmPortalFree (void * *h_qm_portal*)

Frees all resources that were assigned to a QM-Portal module.

Calling this routine invalidates the descriptor.

Parameters

in	*h_qm_portal*	- A handle to a QM-Portal module

Returns

OS_SUCCESS on success; Error code otherwise.

3.7.7.3.2.4.4 os_status qmPortalConfigDcaMode (void * *h_qm_portal,* bool *enable*)

Change the Discrete Consumption Acknowledge mode from its default configuration [DEFAULT_↩ dequeueDcaMode].

Parameters

in	*h_qm_portal*	- A handle to a QM-Portal module
in	*enable*	- Enable/Disable DCA mode

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following qmPortalConfig() and before qmPortalInit().

3.7.7.3.2.4.5 os_status qmPortalConfigStash (void * *h_qm_portal,* qm_portal_stash_param_t * *stash_params*)

Configures the portal to active stash mode.

Parameters

in	*h_qm_portal*	- A handle to a QM-Portal module
in	*stash_params*	- Pointer to data structure of parameters

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following qmPortalConfig() and before qmPortalInit().

3.7.7.3.2.4.6 os_status qmPortalConfigDequeueRingSize (void * *h_qm_portal,* uint8_t *dqrr_size*)

Change the Dequeue Ring Size from its default configuration [DEFAULT_dqrrSize].

Parameters

in	*h_qm_portal*	- A handle to a QM-Portal module
in	*dqrr_size*	- if 0 disable the dequeue, else 1-15 are valid values.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following qmPortalConfig() and before qmPortalInit().

3.7.7.3.2.4.7 os_status qmPortalConfigPullMode (void * *h_qm_portal,* bool *pull_mode*)

Change the Pull Mode from its default configuration [DEFAULT_pullMode].

Parameters

in	*h_qm_portal*	- A handle to a QM-Portal module
in	*pull_mode*	- When TRUE, the Portal will work in pull mode.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following qmPortalConfig() and before qmPortalInit().

3.7.7.3.3 QM-Portal Runtime

3.7.7.3.3.1 Overview

QM-Portal Runtime API functions, definitions and enums.

Data Structures

- struct qm_portal_frame_info_t

Functions

- os_status qmPortalSetDequeueInterruptCoalescing (void *h_qm_portal, uint8_t dqrr_thresh, uint8_t mr_thresh, uint32_t timeout_period)
- os_status qmPortalPoll (void *h_qm_portal, qm_portal_poll_source_t source)
- os_status qmPortalPollFrame (void *h_qm_portal, qm_portal_frame_info_t *frame_info)

3.7.7.3.3.2 Data Structure Documentation

3.7.7.3.3.2.1 struct qm_portal_frame_info_t

structure representing QM Portal Frame Info

3.7.7.3.3.3 Function Documentation

3.7.7.3.3.3.1 os_status qmPortalSetDequeueInterruptCoalescing (void * *h_qm_portal,* uint8_t *dqrr_thresh,* uint8_t *mr_thresh,* uint32_t *timeout_period*)

Sets the threshold for DQRR and MR interrupt generation, and also the timeout period in case the threshold wasn't reached.

The default values are as followed: Dqrr - [DEFAULT_dqrrInterruptThresh] Mr - [DEFAULT_mr↩ InterruptThresh] Timeout - [DEFAULT_dequeueInterrupTimoutPeriod]

Parameters

in	*h_qm_portal*	- A handle to a QM-Portal module
in	*dqrr_thresh*	- The DQRR interrupt asserts when the ring contains greater than 'dqrr↩ _thresh' valid entries.
in	*mr_thresh*	- The MR interrupt asserts when the ring contains greater than 'mr_↩ thresh' valid entries.
in	*timeout_period*	- If the DQRR/MR interrupt wasn't asserted due to the threshold, it will be asserted after the time specified in 'timeout_period'; Note: This value is in Qman clock units, permit values are 0 - (256M-1).

Returns

E_OK on success; Error code otherwise.

Warning

Allowed only following qmPortalInit().

3.7.7.3.3.3.2 os_status qmPortalPoll (void * *h_qm_portal,* qm_portal_poll_source_t *source*)

Poll frames from the specified SW portal.

Parameters

in	*h_qm_portal*	- A handle to a QM-Portal module
in	*source*	- The selected frames type to poll

Returns

E_OK on success; Error code otherwise.

Warning

Allowed only following qmPortalInit().

3.7.7.3.3.3.3 os_status qmPortalPollFrame (void * *h_qm_portal,* qm_portal_frame_info_t * *frame_info*)

Poll frames from the specified SW portal; will poll only data frames.

Parameters

in	*h_qm_portal*	- A handle to a QM-Portal module
out	*frame_info*	- A structure to hold the dequeued frame information

Returns

E_OK on success; Error code otherwise.

Warning

Allowed only following qmPortalInit().

3.7.7.4 QM Frame-Queue-Range API

3.7.7.4.1 Overview

QM-FQR API functions, definitions and enums.

Modules

- QM-FQR Initialization
- QM-FQR Runtime
- QM Congestion Group API

General QM FQR defines

- #define QM_FQR_ALL_QUEUES (uint32_t)(-1)
- enum qm_fq_channel_t {
  E_QM_FQ_CHANNEL_SWPORTAL0 = 0x0 , E_QM_FQ_CHANNEL_POOL1 = 0x401 , E_Q↩
  M_FQ_CHANNEL_FMAN0_SP0 = 0x800 ,
  E_QM_FQ_CHANNEL_FMAN1_SP0 = 0x820 , E_QM_FQ_CHANNEL_CAAM = 0x840, E_↩
  QM_FQ_CHANNEL_PME_SP0 = 0x860,
  E_QM_FQ_CHANNEL_RMAN_SP0 = 0x880 , E_QM_FQ_CHANNEL_DCE_SP0 = 0x8a0 }

3.7.7.4.2 Macro Definition Documentation

3.7.7.4.2.1 #define QM_FQR_ALL_QUEUES (uint32_t)(-1)

General QM FQR defines.

3.7.7.4.3 Enumeration Type Documentation

3.7.7.4.3.1 enum qm_fq_channel_t

Work Queue Channel assignments in QMan.

Enumerator

E_QM_FQ_CHANNEL_SWPORTAL0 Dedicated channels serviced by software portals 0 to 24.

E_QM_FQ_CHANNEL_POOL1 Pool channels that can be serviced by any of the software portals.

E_QM_FQ_CHANNEL_FMAN0_SP0 Dedicated channels serviced by Direct Connect Portal 0↩ : connected to FMan 0; assigned in incrementing order to each sub-portal (SP) in the portal.

E_QM_FQ_CHANNEL_FMAN1_SP0 Dedicated channels serviced by Direct Connect Portal 1↩ : connected to FMan 1; assigned in incrementing order to each sub-portal (SP) in the portal.

E_QM_FQ_CHANNEL_CAAM Dedicated channel serviced by Direct Connect Portal 2: connected to SEC.

E_QM_FQ_CHANNEL_PME_SP0 Dedicated channels serviced by Direct Connect Portal 3: connected to PME.

E_QM_FQ_CHANNEL_RMAN_SP0 Dedicated channels serviced by Direct Connect Portal 4↩ : connected to RMan.

E_QM_FQ_CHANNEL_DCE_SP0 Dedicated channels serviced by Direct Connect Portal 5: connected to DCE.

3.7.7.4.4 QM-FQR Initialization

3.7.7.4.4.1 Overview

QMan Frame-Queue-Range Initialization Unit

Data Structures

- struct qm_fqr_congestion_avoidance_params_t
- struct qm_fqr_params_t
- struct qm_fqr_modify_params_t

General QM FQR defines

- enum qm_fq_channel_t {
  E_QM_FQ_CHANNEL_SWPORTAL0 = 0x0 , E_QM_FQ_CHANNEL_POOL1 = 0x401 , E_Q↩
  M_FQ_CHANNEL_FMAN0_SP0 = 0x800 ,
  E_QM_FQ_CHANNEL_FMAN1_SP0 = 0x820 , E_QM_FQ_CHANNEL_CAAM = 0x840, E_↩
  QM_FQ_CHANNEL_PME_SP0 = 0x860,
  E_QM_FQ_CHANNEL_RMAN_SP0 = 0x880 , E_QM_FQ_CHANNEL_DCE_SP0 = 0x8a0 }

QM-FQR modification options

- typedef uint32_t qm_fqr_mod_opt_t
- void ∗ qmFqrCreate (qm_fqr_params_t ∗qm_fqr_params)
- os_status qmFqrFree (void ∗h_qm_fqr)
- os_status qmFqrFreeWDrain (void ∗h_qm_fqr, qm_fqr_drained_completion_cb_t ∗f_completion↩
  _cb, bool deliver_frame, qm_received_frame_callback_t ∗f_callback, void ∗h_app)
- os_status qmFqrModify (void ∗h_qm_fqr, uint32_t fqid_offset, qm_fqr_modify_params_t ∗params)
- #define QM_FQR_MOD_OPT_HOLD_ACTIVE 0x00000001
- #define QM_FQR_MOD_OPT_AVOID_BLOCKING 0x00000002
- #define QM_FQR_MOD_OPT_PREFER_IN_CACHE 0x00000004
- #define QM_FQR_MOD_OPT_STASH_PARAMS 0x00000008
- #define QM_FQR_MOD_OPT_CTXA 0x00000010
- #define QM_FQR_MOD_OPT_CTXB 0x00000020
- #define QM_FQR_MOD_OPT_CH 0x00000040
- #define QM_FQR_MOD_OPT_WQ 0x00000080
- #define QM_FQR_MOD_OPT_CG 0x00000100

3.7.7.4.4.2 Data Structure Documentation

3.7.7.4.4.2.1 struct qm_fqr_congestion_avoidance_params_t

structure representing QM FQ-Range congestion group parameters

Data Fields

- void ∗ h_qm_cg
- int8_t overhead_accounting_length
- uint32_t fq_tail_drop_threshold

3.7.7.4.4.2.2 struct qm_fqr_params_t

structure representing QM FQ-Range initialization parameters

Data Fields

- void * h_qm
- void * h_qm_portal
- bool init_parked
- bool hold_active
- bool avoid_blocking
- bool prefer_in_cache
- bool use_context_a_for_stash
- qm_context_b_t * context_b
- qm_fq_channel_t channel
- uint8_t wq
- bool shadow_mode
- uint32_t num_of_fqids
- bool use_force
- bool congestion_avoidance_enable
- qm_fqr_congestion_avoidance_params_t congestion_avoidance_params

3.7.7.4.4.2.3 struct qm_fqr_modify_params_t

structure representing QM FQ-Range modification parameters

Data Fields

- qm_fqr_mod_opt_t mod_options
- bool hold_active
- bool avoid_blocking
- bool prefer_in_cache
- bool use_context_a_for_stash
- qm_context_b_t context_b
- qm_fq_channel_t channel
- uint8_t wq
- bool congestion_avoidance_enable
- qm_fqr_congestion_avoidance_params_t congestion_avoidance_params
- qm_fqr_drained_completion_cb_t * f_completion_cb
- bool deliver_frame
- qm_received_frame_callback_t * f_callback
- void * h_app

3.7.7.4.4.3 Macro Definition Documentation

3.7.7.4.4.3.1 #define QM_FQR_MOD_OPT_HOLD_ACTIVE 0x00000001

Modify the "hold-active" attribute.

3.7.7.4.4.3.2 #define QM_FQR_MOD_OPT_AVOID_BLOCKING 0x00000002

Modify the "avoid-blocking" attribute.

3.7.7.4.4.3.3 #define QM_FQR_MOD_OPT_PREFER_IN_CACHE 0x00000004

Modify the "prefer-in-cache" attribute.

3.7.7.4.4.3.4 #define QM_FQR_MOD_OPT_STASH_PARAMS 0x00000008

Modify the stashing attribute.

3.7.7.4.4.3.5 #define QM_FQR_MOD_OPT_CTXA 0x00000010

Modify the contextA attribute.

3.7.7.4.4.3.6 #define QM_FQR_MOD_OPT_CTXB 0x00000020

Modify the contextB attribute.

3.7.7.4.4.3.7 #define QM_FQR_MOD_OPT_CH 0x00000040

Modify the channel attribute.

3.7.7.4.4.3.8 #define QM_FQR_MOD_OPT_WQ 0x00000080

Modify the wq attribute.

3.7.7.4.4.3.9 #define QM_FQR_MOD_OPT_CG 0x00000100

Modify the CG attribute.

3.7.7.4.4.4 Typedef Documentation

3.7.7.4.4.4.1 typedef uint32_t qm_fqr_mod_opt_t

a type to represent all queue modification options

3.7.7.4.4.5 Enumeration Type Documentation

3.7.7.4.4.5.1 enum qm_fq_channel_t

Work Queue Channel assignments in QMan.

Enumerator

E_QM_FQ_CHANNEL_SWPORTAL0 Dedicated channels serviced by software portals 0 to 24.
E_QM_FQ_CHANNEL_POOL1 Pool channels that can be serviced by any of the software portals.

E_QM_FQ_CHANNEL_FMAN0_SP0 Dedicated channels serviced by Direct Connect Portal 0↩ : connected to FMan 0; assigned in incrementing order to each sub-portal (SP) in the portal.

E_QM_FQ_CHANNEL_FMAN1_SP0 Dedicated channels serviced by Direct Connect Portal 1↩ : connected to FMan 1; assigned in incrementing order to each sub-portal (SP) in the portal.

E_QM_FQ_CHANNEL_CAAM Dedicated channel serviced by Direct Connect Portal 2: connected to SEC.

E_QM_FQ_CHANNEL_PME_SP0 Dedicated channels serviced by Direct Connect Portal 3: connected to PME.

E_QM_FQ_CHANNEL_RMAN_SP0 Dedicated channels serviced by Direct Connect Portal 4↩ : connected to RMan.

E_QM_FQ_CHANNEL_DCE_SP0 Dedicated channels serviced by Direct Connect Portal 5: connected to DCE.

3.7.7.4.4.6 Function Documentation

3.7.7.4.4.6.1 void* qmFqrCreate (qm_fqr_params_t * *qm_fqr_params*)

Initializing and enabling a Frame-Queue-Range; This routine should be called for adding an FQR.

Parameters

in	*qm_fqr_↩ params*	- A structure of parameters for defining the desired queues parameters.

Returns

A handle to the initialized FQR on success; NULL code otherwise.

Warning

Allowed only following qmInit().

3.7.7.4.4.6.2 os_status qmFqrFree (void * *h_qm_fqr*)

Deleting and free all resources of an initialized FQR.

Parameters

in	*h_qm_fqr*	- A handle to a QM-FQR Module.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following qmFqrCreate() for this FQR.

3.7.7.4.4.6.3 os_status qmFqrFreeWDrain (void * *h_qm_fqr,* qm_fqr_drained_completion_cb_t * *f_completion_cb,* bool *deliver_frame,* qm_received_frame_callback_t * *f_callback,* void * *h_app*)

Deleting and free all resources of an initialized FQR with the option of draining.

If interrupts are disabled (i.e. polling mode) than f_completion_cb should be passed and user should call the portal poll function. If interrupts are enabled than f_completion_cb should be NULL.

Parameters

in	*h_qm_fqr*	- A handle to a QM-FQR Module.
in	*f_completion_cb*	- Pointer to a completion callback to be used in non-blocking mode.
in	*deliver_frame*	- TRUE for deliver the drained frames to the user; FALSE for not deliver the frames.
in	*f_callback*	- Pointer to a callback to handle the delivered frames.
in	*h_app*	- User's application descriptor.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following qmFqrCreate() for this FQR.

3.7.7.4.4.6.4 os_status qmFqrModify (void * *h_qm_fqr,* uint32_t *fqid_offset,* qm_fqr_modify_params_t * *params*)

Modify attributes for the given fqid_offset.

Note that if 'fqid_offset==QM_FQR_ALL_QUEUES' than it will be applied on all fqids in this FQR; Before the modification the function will perform draining.

Parameters

in	*h_qm_fqr*	- A handle to a QM-FQR Module.
in	*fqid_offset*	- Fqid offset within the FQ-Range.
in	*params*	- A pointer to the modification parameters structure.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following qmInit() and qmFqrCreate() for this FQR.

3.7.7.4.5 QM-FQR Runtime

3.7.7.4.5.1 Overview

QM-FQR Runtime API functions, definitions and enums.

Enumerations

- enum qm_fqr_counters_t { E_QM_FQR_COUNTERS_FRAME = 0, E_QM_FQR_COUNTERS_BYTE }

Functions

- os_status qmFqrRegisterCB (void *h_qm_fqr, qm_received_frame_callback_t *f_callback, void *h_app)
- uint32_t qmFqrGetFqid (void *h_qm_fqr)
- uint32_t qmFqrGetCounter (void *h_qm_fqr, void *h_qm_portal, uint32_t fqid_offset, qm_fqr_counters_t counter)
- os_status qmFqrEnqueue (void *h_qm_fqr, void *h_qm_portal, uint32_t fqid_offset, dpaa_fd_t *frame)

3.7.7.4.5.2 Enumeration Type Documentation

3.7.7.4.5.2.1 enum qm_fqr_counters_t

enum for defining QM-FQR counters

Enumerator

E_QM_FQR_COUNTERS_FRAME Total number of frames on this frame queue.
E_QM_FQR_COUNTERS_BYTE Total number of bytes in all frames on this frame queue.

3.7.7.4.5.3 Function Documentation

3.7.7.4.5.3.1 os_status qmFqrRegisterCB (void * *h_qm_fqr*, qm_received_frame_callback_t * *f_callback*, void * *h_app*)

Register a callback routine to be called when a frame comes from this FQ-Range

Parameters

in	*h_qm_fqr*	- A handle to a QM-FQR Module.
in	*f_callback*	- An application callback
in	*h_app*	- User's application descriptor

Returns

E_OK on success; Error code otherwise.

Warning

Allowed only following QM_FQR_Create().

3.7.7.4.5.3.2 uint32_t qmFqrGetFqid (void * *h_qm_fqr*)

Returned the Fqid base of the FQ-Range

Parameters

in	*h_qm_fqr*	- A handle to a QM-FQR Module.

Returns

Fqid base.

Warning

Allowed only following QM_FQR_Create().

3.7.7.4.5.3.3 uint32_t qmFqrGetCounter (void * *h_qm_fqr,* void * *h_qm_portal,* uint32_t *fqid_offset,* qm_fqr_counters_t *counter*)

Reads one of the QM-FQR counters.

Parameters

in	*h_qm_fqr*	- A handle to a QM-FQR Module.
in	*h_qm_portal*	- A handle to a QM Portal Module; NOTE : if NULL, assuming affinity.
in	*fqid_offset*	- Fqid offset within the FQ-Range.
in	*counter*	- The requested counter.

Returns

Counter's current value.

Warning

Allowed only following QM_FQR_Create(); fqid_offset cannot be equal to QM_FQR_ALL_QU↩ EUES.

3.7.7.4.5.3.4 os_status qmFqrEnqueue (void * *h_qm_fqr,* void * *h_qm_portal,* uint32_t *fqid_offset,* dpaa_fd_t * *frame*)

Enqueue the frame into the FQ to be transmitted.

Parameters

in	*h_qm_fqr*	- A handle to a QM-FQR Module.
in	*h_qm_portal*	- A handle to a QM Portal Module; NOTE : if NULL, assuming affinity.
in	*fqid_offset*	- Fqid offset within the FQ-Range.
in	*frame*	- Pointer to the frame to be enqueued.

Returns

E_OK on success; Error code otherwise.

Warning

Allowed only following QM_FQR_Create(); fqid_offset cannot be equal to QM_FQR_ALL_QU↩ EUES.

3.7.7.4.5.4 QM Congestion Group API

3.7.7.4.5.4.1 Overview

```
QM-CG API functions, definitions and enums.
```

Modules

- QM-Congestion Group Initialization
- QM-CG Runtime

3.7.7.4.5.4.2 QM-Congestion Group Initialization

3.7.7.4.5.4.3 Overview

```
QM-CG Initialization Unit
```

Data Structures

- struct qm_cg_wred_curve_t
- struct qm_cg_wred_params_t
- struct qm_cg_params_t

Functions

- void * qmCgCreate (qm_cg_params_t *cg_params)
- os_status qmCgFree (void *h_qm_cg)

3.7.7.4.5.4.4 Data Structure Documentation

3.7.7.4.5.4.5 struct qm_cg_wred_curve_t

structure representing QM CG WRED curve

Data Fields

- uint32_t max_th
- uint32_t min_th
- uint8_t probability_denominator

3.7.7.4.5.4.6 struct qm_cg_wred_params_t

structure representing QM CG WRED parameters

3.7.7.4.5.4.7 struct qm_cg_params_t

structure representing QM CG configuration parameters

Data Fields

- void * h_qm
- void * h_qm_portal
- bool frame_count
- bool wred_enable
- qm_cg_wred_params_t wred_params
- bool tail_drop_enable
- uint32_t threshold
- bool notify_dc_portal
- dpaa_dc_portal_t dc_portal_id
- qm_exceptions_callback_t * f_exception
- void * h_app

3.7.7.4.5.4.8 Function Documentation

3.7.7.4.5.4.9 void* qmCgCreate (qm_cg_params_t * *cg_params*)

Create and configure a congestion Group.

Parameters

in	*cg_params*	- CG parameters

Returns

A handle to the CG module

Warning

Allowed only following qmInit().

3.7.7.4.5.4.10 os_status qmCgFree (void * *h_qm_cg*)

Deleting and free all resources of an initialized CG.

Parameters

in	*h_qm_cg*	- A handle to a QM-CG Module.

Returns

OS_SUCCESS on success; Error code otherwise.

Warning

Allowed only following qmInit() and qmCgCreate() for this CG.

3.7.7.4.5.4.11 QM-CG Runtime

3.7.7.4.5.4.12 Overview

QM-CG Runtime API functions, definitions and enums.

Data Structures

- struct qm_cg_modify_wred_params_t

Enumerations

- enum qm_cg_color_t

Functions

- os_status qmCgSetException (void *h_qm_cg, qm_exceptions_t exception, bool enable)
- os_status qmCgModifyWredCurve (void *h_qm_cg, qm_cg_modify_wred_params_t *qm_cg_↩ modify_params)
- os_status qmCgModifyTailDropThreshold (void *h_qm_cg, uint32_t threshold)

3.7.7.4.5.4.13 Data Structure Documentation

3.7.7.4.5.4.14 struct qm_cg_modify_wred_params_t

structure representing QM CG modification parameters

3.7.7.4.5.4.15 Enumeration Type Documentation

3.7.7.4.5.4.16 enum qm_cg_color_t

structure representing QM CG WRED colors

3.7.7.4.5.4.17 Function Documentation

3.7.7.4.5.4.18 os_status qmCgSetException (void * *h_qm_cg*, qm_exceptions_t *exception*, bool *enable*)

Set CG exception.

Parameters

in	*h_qm_cg*	- A handle to a QM-CG Module.
in	*exception*	- exception enum
in	*enable*	- TRUE to enable, FALSE to disable.

Returns

E_OK on success; Error code otherwise.

Warning

Allowed only following QM_Init() and QM_CG_Create() for this CG.

3.7.7.4.5.4.19 os_status qmCgModifyWredCurve (void * *h_qm_cg*, qm_cg_modify_wred_↩ params_t * *qm_cg_modify_params*)

Change WRED curve parameters for a selected color.

Note that this routine may be called only for valid CG's that already have been configured for WRED, and only need a change in the WRED parameters.

Parameters

in	*h_qm_cg*	- A handle to a QM-CG Module.
in	*qm_cg_↩ modify_params*	- A structure of new WRED parameters.

Returns

E_OK on success; Error code otherwise.

Warning

Allowed only following QM_Init() and QM_CG_Create() for this CG.

3.7.7.4.5.4.20 os_status qmCgModifyTailDropThreshold (void * *h_qm_cg,* uint32_t *threshold*)

Change WRED curve parameters for a selected color.

Note that this routine may be called only for valid CG's that already have been configured for tail drop, and only need a change in the threshold value.

Parameters

in	*h_qm_cg*	- A handle to a QM-CG Module.
in	*threshold*	- New threshold.

Returns

E_OK on success; Error code otherwise.

Warning

Allowed only following QM_Init() and QM_CG_Create() for this CG.

3.7.7.5 PDSCH Initialization

3.7.7.5.1 Overview

PDSCH device initialization API

Data Structures

- struct pdsch_config_param_t
- struct pdsch_open_params_t
- struct maple_pdsch_ch_add_int_params_t
- struct maple_pdsch_ch_open_params_t

Macros

- #define PDSCH_MAX_USER_PER_SLOT 200
- #define PDSCH_CW_PER_USER 2
- #define PDSCH_VANT_LAYERS 8

Typedefs

- typedef void(∗ maple_pdsch_int_cb_t)(uint32_t status, os_virt_ptr bd_addr)
- typedef maple_pe_init_params_t maple_pdsch_init_params_t
- typedef struct maple_pe_init_params_s maple_pdsch_init_params_s

Enumerations

- enum { MAPLE_PDSCH_BF_CONFIG_CSRS = 0, MAPLE_PDSCH_BF_CONFIG_MBSFNRS = 1, MAPLE_PDSCH_BF_CONFIG_POSRS = 2, MAPLE_PDSCH_BF_CONFIG_PSS = 3, MAPLE_PDSCH_BF_CONFIG_SSS = 4, MAPLE_PDSCH_BF_CONFIG_CSIRS = 5, MAPLE_PDSCH_BF_CONFIG_PBCH = 6, MAPLE_PDSCH_BF_CONFIG_UE = 7 }
- enum { MAPLE_PDSCH_1_BF_MATRIX_1_RB_RS = 0x00000000, MAPLE_PDSCH_1_BF_MATRIX_1_2_RB_RS = 0x00010000, MAPLE_PDSCH_1_BF_MATRIX_1_3_RB_RS = 0x00020000, MAPLE_PDSCH_1_BF_MATRIX_1_4_RB_RS = 0x00030000, MAPLE_PDSCH_1_BF_MATRIX_1_6_RB_RS = 0x00040000, MAPLE_PDSCH_1_BF_MATRIX_1_12_RB_RS = 0x00050000, MAPLE_PDSCH_1_BF_MATRIX_1_24_RB_RS = 0x00060000, MAPLE_PDSCH_1_BF_MATRIX_ALL_RB_RS = 0x00070000, MAPLE_PDSCH_1_BF_MATRIX_25_RB_RS = 0x00080000, MAPLE_PDSCH_1_BF_MATRIX_20_RB_RS = 0x00090000, MAPLE_PDSCH_1_BF_MATRIX_15_RB_RS = 0x000A0000, MAPLE_PDSCH_1_BF_MATRIX_10_RB_RS = 0x000B0000, MAPLE_PDSCH_1_BF_MATRIX_5_RB_RS = 0x000C0000, MAPLE_PDSCH_1_BF_MATRIX_4_RB_RS = 0x000D0000, MAPLE_PDSCH_1_BF_MATRIX_3_RB_RS = 0x000E0000, MAPLE_PDSCH_1_BF_MATRIX_2_RB_RS = 0x000F0000 }
- enum { MAPLE_PDSCH_STEERING_USER_HEADER_PTR = 0, MAPLE_PDSCH_STEERING_CW_HEADER_PTR = 1, MAPLE_PDSCH_STEERING_RB_MAP_TABLE_PTR = 2, MAPLE_PDSCH_STEERING_UE_RS_HEADER_PTR = 3, MAPLE_PDSCH_STEERING_ANT0_DATA_PTR = 4, MAPLE_PDSCH_STEERING_ANT1_DATA_PTR = 5, MAPLE_PDSCH_STEERING_ANT2_DATA_PTR = 6, MAPLE_PDSCH_STEERING_ANT3_DATA_PTR = 7, MAPLE_PDSCH_STEERING_ANT4_DATA_PTR = 8, MAPLE_PDSCH_STEERING_ANT5_DATA_PTR = 9, MAPLE_PDSCH_STEERING_ANT6_DATA_PTR = 10, MAPLE_PDSCH_STEERING_ANT7_DATA_PTR = 11, MAPLE_PDSCH_STEERING_EXT_OFDM_INPUT_PTR = 12, MAPLE_PDSCH_STEERING

_BF_PTR = 13, MAPLE_PDSCH_STEERING_SM_PTR = 14, MAPLE_PDSCH_STEERING_PSS_SSS_INPUT_PTR = 15, MAPLE_PDSCH_STEERING_TB_INPUT_PTR = 16, MAPLE_PDSCH_STEERING_DNOSIBA_PTR = 17, MAPLE_PDSCH_STEERING_BF_COEFF_PTR = 18, MAPLE_PDSCH_STEERING_POD_PTR = 19, MAPLE_PDSCH_STEERING_PID_PTR = 20, MAPLE_PDSCH_STEERING_PM_PTR = 21, MAPLE_PDSCH_STEERING_PAD_PTR = 22, MAPLE_PDSCH_STEERING_IMPORT_PTR = 23, MAPLE_PDSCH_STEERING_CSI_HEADER_PTR = 24, MAPLE_PDSCH_STEERING_CSRS_HEADER_PTR = 25 }

- enum { MAPLE_PDSCH_SYMBOL_INT, MAPLE_PDSCH_ACK_INT, MAPLE_PDSCH_NUM_ADD_INT }

Functions

- os_status maplePdschInitialize (maple_pdsch_init_params_s *init_params, unsigned int num_devices, os_status(*channel_dispatch)(void *channel, void *jobs, int *num_jobs), void(*channel_reap)(void *channel, void *maple))

MAPLE PDSCH FLAGS for maple_pe_ch_open_params_t.bf_config[x][sector]

- #define MAPLE_PDSCH_CONFIG_BF_EN 0x80000000
- #define MAPLE_PDSCH_CONFIG_BF_OFFSET(BF_OFFSET) ((uint16_t)BF_OFFSET << 2)

MAPLE PDSCH Device Names and IDs

- #define **PDSCH_DEV_ID_0** 0
- #define **PDSCH_DEV_ID_1** 1
- #define **MAPLE_0_PDSCH_NAME** "PDSCH0"
- #define **MAPLE_1_PDSCH_NAME** "PDSCH1"

PDSCH EDF General Configuration Parameter flags; Used for maple_pdsch_edf_init_params_t.MPDSCHCP

- enum
- #define PDSCH_SYNC_ENALE 0x00008000
- #define PDSCH_IFFT_DISABLE 0x00004000
- #define PDSCH_DNOSD 0x00002000
- #define PDSCH_MNOSEM_EN 0x00001000

PDSCH EDF Target Antenna Configuration Structure parameters;

```
Used for maple_pdsch_edf_init_params_t.MPTACSs[NUM_SECTOR][x]
```

- enum { PDSCH_SSS_TGT_ANT = 0, PDSCH_PSS_TGT_ANT = 1, PDSCH_ALT_DBW_PARAM = 3, PDSCH_POS_TGT_ANT = 12, PDSCH_MBSFN_TGT_ANT = 13 }

- enum pdsch_mode_alt_dbw_t {
  PDSCH_MODE_ALT_DBW_72_SAMPLES = 0x00000000, PDSCH_MODE_ALT_DBW_180↩
  _SAMPLES = 0x00000001, PDSCH_MODE_ALT_DBW_512_SAMPLES = 0x00000002,
  PDSCH_MODE_ALT_DBW_1024_SAMPLES = 0x00000003, PDSCH_MODE_ALT_DBW_↩
  1536_SAMPLES = 0x00000004, PDSCH_MODE_ALT_DBW_2048_SAMPLES = 0x00000005
  }
- #define PDSCH_MODE_ALT_DBW_EN 0x00000008

PDSCH EDF Sector Configuration Parameter flags;

```
Used for maple_pdsch_edf_init_params_t.MPSCPs[NUM_SECTOR]
```

- enum {
  PDSCH_SECTOR_BA_ALOCATION_0 = 0x00000000, PDSCH_SECTOR_BA_ALOCATION↩
  _1 = 0x01000000, PDSCH_SECTOR_BA_ALOCATION_2 = 0x02000000,
  PDSCH_SECTOR_BA_ALOCATION_3 = 0x03000000, PDSCH_SECTOR_BA_ALOCATION↩
  _4 = 0x04000000, PDSCH_SECTOR_BA_ALOCATION_5 = 0x05000000,
  PDSCH_SECTOR_BA_ALOCATION_6 = 0x06000000, PDSCH_SECTOR_BA_ALOCATION↩
  _7 = 0x07000000 }
- enum {
  PDSCH_SECTOR_RES_ALOCATION_0 = 0x00000000, PDSCH_SECTOR_RES_ALOCATI↩
  ON_1 = 0x10000000, PDSCH_SECTOR_RES_ALOCATION_2 = 0x20000000,
  PDSCH_SECTOR_RES_ALOCATION_3 = 0x30000000, PDSCH_SECTOR_RES_ALOCATI↩
  ON_4 = 0x40000000, PDSCH_SECTOR_RES_ALOCATION_5 = 0x50000000,
  PDSCH_SECTOR_RES_ALOCATION_6 = 0x60000000, PDSCH_SECTOR_RES_ALOCATI↩
  ON_7 = 0x70000000 }
- enum { PDSCH_NUM_CELL_REF_PORTS_1 = 0x00000000, PDSCH_NUM_CELL_REF_PO↩
  RTS_2 = 0x00400000, PDSCH_NUM_CELL_REF_PORTS_4 = 0x00800000 }
- enum { PDSCH_USE_1_FTPE = 0x00001000, PDSCH_USE_2_FTPE = 0x00002000, PDSCH_↩
  USE_3_FTPE = 0x00003000 }
- enum { PDSCH_NUM_PBCH_ANT_PORTS_2 = 0x00000000, PDSCH_NUM_PBCH_ANT_P↩
  ORTS_4 = 0x00000800 }
- enum {
  PDSCH_NUM_VIRT_ANT_1 = 0x00000000, PDSCH_NUM_VIRT_ANT_2 = 0x00000100, P↩
  DSCH_NUM_VIRT_ANT_4 = 0x00000300,
  PDSCH_NUM_VIRT_ANT_8 = 0x00000700 }
- enum {
  PDSCH_NUM_PHYS_ANT_1 = 0x00000000, PDSCH_NUM_PHYS_ANT_2 = 0x00000010, P↩
  DSCH_NUM_PHYS_ANT_4 = 0x00000030,
  PDSCH_NUM_PHYS_ANT_8 = 0x00000070 }
- enum pdsch_mode_sys_bw_t {
  PDSCH_MODE_BW_1_4MHZ = 0x00000000, PDSCH_MODE_BW_3MHZ = 0x00000001, P↩
  DSCH_MODE_BW_5MHZ = 0x00000002,
  PDSCH_MODE_BW_10MHZ = 0x00000003, PDSCH_MODE_BW_15MHZ = 0x00000004, P↩
  DSCH_MODE_BW_20MHZ = 0x00000005 }
- INLINE uint32_t PDSCH_SECTOR_BA (uint8_t S_BA)
- #define PDSCH_CYCLIC_ENALE 0x00004000

- #define PDSCH_CYCLIC_ENABLE PDSCH_CYCLIC_ENALE
- #define PDSCH_BF_ENABLE 0x00000008

3.7.7.5.2 Data Structure Documentation

3.7.7.5.2.1 struct pdsch_config_param_t

PDSCH sector configuration parameters.

Data Fields

- uint32_t mnos:4
- uint32_t cell_id:9
- uint32_t mbsfn_id:9

3.7.7.5.2.1.1 Field Documentation

3.7.7.5.2.1.2 uint32_t pdsch_config_param_t::mnos

Maximum Number of OFDM Symbols; Valid only if SYNC_FLOW == 1 & IFFT_DIS == 0.

Valid values: 2 to 15

3.7.7.5.2.1.3 uint32_t pdsch_config_param_t::cell_id

Cell ID number; Valid values: 0...503.

3.7.7.5.2.1.4 uint32_t pdsch_config_param_t::mbsfn_id

MBSFN ID used for scrambling initialization calculation of MBSFN.

3.7.7.5.2.2 struct pdsch_open_params_t

PDSCH open device LLD parameters.

Data Fields

- void * maple_handle
- void * antx_out_addr [MAPLE_NUM_SECTORS][MAPLE_NUM_ANT]
- uint32_t antx_out_size [MAPLE_NUM_SECTORS]
- pdsch_config_param_t cell_config [MAPLE_NUM_SECTORS]
- uint32_t bf_config [MAPLE_PDSCH_BF_CONFIG_LAST][MAPLE_NUM_SECTORS]

3.7.7.5.2.2.1 Field Documentation

3.7.7.5.2.2.2 void* pdsch_open_params_t::maple_handle

Handle returned from osCopDeviceOpen() for MAPLE controller.

3.7.7.5.2.2.3 void* pdsch_open_params_t::antx_out_addr[MAPLE_NUM_SECTORS][MAPLE_NUM_ANT]

Base addresses of the output buffer of antenna; Driver won't translate these addresses.

3.7.7.5.2.2.4 uint32_t pdsch_open_params_t::antx_out_size[MAPLE_NUM_SECTORS]

Antenna Output buffer size.

3.7.7.5.2.2.5 pdsch_config_param_t pdsch_open_params_t::cell_config[MAPLE_NUM_SECTORS]

Sector configuration.

3.7.7.5.2.2.6 uint32_t pdsch_open_params_t::bf_config[MAPLE_PDSCH_BF_CONFIG_LAST][MAPLE_NUM_SECTORS]

Beam Forming Configuration.

3.7.7.5.2.3 struct maple_pdsch_ch_add_int_params_t

MAPLE PDSCH channel open parameters for symbol interrupt.

Data Fields

- uint32_t int_enable:1
- uint32_t multiplexed_int:1
- os_hwi_dispatcher int_dispatcher
- maple_pdsch_int_cb_t int_callback
- os_hwi_handle int_num
- os_hwi_priority int_priority

3.7.7.5.2.3.1 Field Documentation

3.7.7.5.2.3.2 uint32_t maple_pdsch_ch_add_int_params_t::int_enable

Set to 1 to enable symbol interrupt for this channel.

3.7.7.5.2.3.3 uint32_t maple_pdsch_ch_add_int_params_t::multiplexed_int

Set to 1 for the driver if this is a multiplexed interrupt line, for example if this interrupt line is used for multiple channels.

3.7.7.5.2.3.4 os_hwi_dispatcher maple_pdsch_ch_add_int_params_t::int_dispatcher

Interrupt dispatcher to be called when interrupt occurs.

3.7.7.5.2.3.5 maple_pdsch_int_cb_t maple_pdsch_ch_add_int_params_t::int_callback

User call back function to be called on interrupt assertion.

3.7.7.5.2.3.6 os_hwi_handle maple_pdsch_ch_add_int_params_t::int_num

Which interrupt line on this device should the interrupt handler register to.

3.7.7.5.2.3.7 os_hwi_priority maple_pdsch_ch_add_int_params_t::int_priority

Interrupt priority.

3.7.7.5.2.4 struct maple_pdsch_ch_open_params_t

MAPLE PDSCH channel open parameters type.

3.7.7.5.2.4.1 Field Documentation

3.7.7.5.2.4.2 uint32_t maple_pdsch_ch_open_params_t::flags

Users should set to 0 prior to setting any of the individual flags to ensure future compatibility.

Future releases may add more optimizations and flags to this 32 bit field. In order to ensure that these features, which will always be enabled by setting the relevant bit(s) to a value other than 0, won't be used unintentionally - the user should set flags to 0

3.7.7.5.2.4.3 uint32_t maple_pdsch_ch_open_params_t::late_pcr_en

Set to 1 to enable LATE PCR on the channel, if this bit is not set, the LATE PCR will be issued beofre returning from channel Dispatch.

3.7.7.5.3 Macro Definition Documentation

3.7.7.5.3.1 #define PDSCH_MAX_USER_PER_SLOT 200

Maximum number of users per slot.

3.7.7.5.3.2 #define PDSCH_CW_PER_USER 2

Number of Code Words per user.

3.7.7.5.3.3 #define PDSCH_VANT_LAYERS 8

Number of virtual antenna layers.

3.7.7.5.3.4 #define MAPLE_PDSCH_CONFIG_BF_EN 0x80000000

Enable BeamForaming.

3.7.7.5.3.5 #define MAPLE_PDSCH_CONFIG_BF_OFFSET(*BF_OFFSET*) ((uint16_t)BF_OFFSET << 2)

BF_COEFF<slot>_ADDRESS + (BF_OFFSET << 2) points to the byte address of the first beam forming matrix.

used for BF processing.

3.7.7.5.3.6 #define PDSCH_SYNC_ENALE 0x00008000

Enable synchronized flow for sector NUM_SECTOR.

3.7.7.5.3.7 #define PDSCH_IFFT_DISABLE 0x00004000

If set, the output of the PDSCH EDF will consist of the data prior to the guard insertion (and IFFT and CP processing)

3.7.7.5.3.8 #define PDSCH_DNOSD 0x00002000

DNOS Disable - As response to the DNOS interrupt the MAPLE resets the interrupt bit, decrements the NOS counter and issues an interrupt to the host.

3.7.7.5.3.9 #define PDSCH_MNOSEM_EN 0x00001000

This field enables setting the PD_UR_ERRfield when the NNOSE<s>is set.

3.7.7.5.3.10 #define PDSCH_MODE_ALT_DBW_EN 0x00000008

ALT_DBW Enable This bit determines whether S_BW or ALT_DBW determines how many input samples are going to be read for IFFT operation as opposed to how many inputs samples are complemented as guardbands.

3.7.7.5.3.11 #define PDSCH_CYCLIC_ENALE 0x00004000

Enable cyclic flow for sector NUM_SECTOR.

3.7.7.5.3.12 #define PDSCH_CYCLIC_ENABLE PDSCH_CYCLIC_ENALE

Enable cyclic flow for sector NUM_SECTOR.

3.7.7.5.3.13 #define PDSCH_BF_ENABLE 0x00000008

Beam forming of data and UE specific Reference signal is enable for sector NUM_SECTOR.

3.7.7.5.4 Typedef Documentation

3.7.7.5.4.1 typedef void(∗ maple_pdsch_int_cb_t)(uint32_t status, os_virt_ptr bd_addr)

Pointer to user call back function to be called on interrupt assertion.

3.7.7.5.4.2 typedef maple_pe_init_params_t maple_pdsch_init_params_t

MAPLE PDSCH initialization parameters type for multiple PEs.

3.7.7.5.4.3 typedef struct maple_pe_init_params_s maple_pdsch_init_params_s

MAPLE PDSCH initialization parameters type for one PE.

3.7.7.5.5 Enumeration Type Documentation

3.7.7.5.5.1 anonymous enum

MAPLE PDSCH BeamForaming configuration pointers mapping.

Use it for accessing relevant maple_pe_ch_open_params_t.bf_config[x][sector]

Enumerator

MAPLE_PDSCH_BF_CONFIG_CSRS CS RS Beam Forming Configuration Parameter.
MAPLE_PDSCH_BF_CONFIG_MBSFNRS BSFN RS Beam Forming Configuration Parameter.
MAPLE_PDSCH_BF_CONFIG_POSRS POS Beam Forming Configuration Parameter.
MAPLE_PDSCH_BF_CONFIG_PSS PSS Beam Forming Configuration Parameter.
MAPLE_PDSCH_BF_CONFIG_SSS SSS Beam Forming Configuration Parameter.
MAPLE_PDSCH_BF_CONFIG_CSIRS CSI RS Beam Forming Configuration Parameter.
MAPLE_PDSCH_BF_CONFIG_PBCH PBCH Beam Forming Configuration Parameter.
MAPLE_PDSCH_BF_CONFIG_UE UE Beam Forming Configuration Parameter.

3.7.7.5.5.2 anonymous enum

MAPLE PDSCH Number of Resource elements in one Beam Forming Matrix mapping.

maple_pe_ch_open_params_t.bf_config[MAPLE_PDSCH_BF_CONFIG_n][sector]; Use for setting the number of resource elements in one beam forming matrix used in MAPLE_PDSCH_BF_CONFIG_n.

Enumerator

MAPLE_PDSCH_1_BF_MATRIX_1_RB_RS One beam forming matrix is handling one full resource block.
MAPLE_PDSCH_1_BF_MATRIX_1_2_RB_RS One beam forming matrix is handling 1/2 a resource block.
MAPLE_PDSCH_1_BF_MATRIX_1_3_RB_RS One beam forming matrix is handling 1/3 a resource block.
MAPLE_PDSCH_1_BF_MATRIX_1_4_RB_RS One beam forming matrix is handling 1/4 a resource block.
MAPLE_PDSCH_1_BF_MATRIX_1_6_RB_RS One beam forming matrix is handling 1/6 a resource block.
MAPLE_PDSCH_1_BF_MATRIX_1_12_RB_RS One beam forming matrix is handling 1/12 a resource block.
MAPLE_PDSCH_1_BF_MATRIX_1_24_RB_RS One beam forming matrix is handling 1/24 a resource block.
MAPLE_PDSCH_1_BF_MATRIX_ALL_RB_RS One beam forming matrix is handling all resource blocks.

MAPLE_PDSCH_1_BF_MATRIX_25_RB_RS One beam forming matrix is handling 25 full resource block.
MAPLE_PDSCH_1_BF_MATRIX_20_RB_RS One beam forming matrix is handling 20 full resource block.
MAPLE_PDSCH_1_BF_MATRIX_15_RB_RS One beam forming matrix is handling 15 full resource block.
MAPLE_PDSCH_1_BF_MATRIX_10_RB_RS One beam forming matrix is handling 10 full resource block.
MAPLE_PDSCH_1_BF_MATRIX_5_RB_RS One beam forming matrix is handling 5 full resource block.
MAPLE_PDSCH_1_BF_MATRIX_4_RB_RS One beam forming matrix is handling 4 full resource block.
MAPLE_PDSCH_1_BF_MATRIX_3_RB_RS One beam forming matrix is handling 3 full resource block.
MAPLE_PDSCH_1_BF_MATRIX_2_RB_RS One beam forming matrix is handling 2 full resource block.

3.7.7.5.5.3 anonymous enum

MAPLE PDSCH steering bits pointers mapping.

Use it for accessing relevant maple_pe_ch_open_params_t.ch_params.steering_bits[x]

Enumerator

MAPLE_PDSCH_STEERING_USER_HEADER_PTR User Header.
MAPLE_PDSCH_STEERING_CW_HEADER_PTR Code Word Header.
MAPLE_PDSCH_STEERING_RB_MAP_TABLE_PTR RB Mapping Pointer.
MAPLE_PDSCH_STEERING_UE_RS_HEADER_PTR UE RS Header.
MAPLE_PDSCH_STEERING_ANT0_DATA_PTR Antenna 0 Output.
MAPLE_PDSCH_STEERING_ANT1_DATA_PTR Antenna 1 Output.
MAPLE_PDSCH_STEERING_ANT2_DATA_PTR Antenna 2 Output.
MAPLE_PDSCH_STEERING_ANT3_DATA_PTR Antenna 3 Output.
MAPLE_PDSCH_STEERING_ANT4_DATA_PTR Antenna 4 Output.
MAPLE_PDSCH_STEERING_ANT5_DATA_PTR Antenna 5 Output.
MAPLE_PDSCH_STEERING_ANT6_DATA_PTR Antenna 6 Output.
MAPLE_PDSCH_STEERING_ANT7_DATA_PTR Antenna 7 Output.
MAPLE_PDSCH_STEERING_EXT_OFDM_INPUT_PTR External OFDM symbol input.
MAPLE_PDSCH_STEERING_BF_PTR Beam forming parameters.
MAPLE_PDSCH_STEERING_SM_PTR Spatial Multiplexing parameters.
MAPLE_PDSCH_STEERING_PSS_SSS_INPUT_PTR PSS/SSS input data.
MAPLE_PDSCH_STEERING_TB_INPUT_PTR Transport block Data input.
MAPLE_PDSCH_STEERING_DNOSIBA_PTR Steering for ack address pointer.
MAPLE_PDSCH_STEERING_BF_COEFF_PTR BF_COEFFS_ADDR pointer steering bits.
MAPLE_PDSCH_STEERING_POD_PTR POD_ADDRESS steering bits.
MAPLE_PDSCH_STEERING_PID_PTR PID_ADDRESS steering bits.
MAPLE_PDSCH_STEERING_PM_PTR PM_ADDRESS steering bits.

MAPLE_PDSCH_STEERING_PAD_PTR EXT_PAD_ADDRESS steering bits.
MAPLE_PDSCH_STEERING_IMPORT_PTR EXT_PAD_ADDRESS steering bits.
MAPLE_PDSCH_STEERING_CSI_HEADER_PTR CSI_HEADER_ADDRESS steering bits.
MAPLE_PDSCH_STEERING_CSRS_HEADER_PTR CSRS_HEADER_ADDRESS steering bits.

3.7.7.5.5.4 anonymous enum

MAPLE PDSCH additional interrupts access enumeration.

Enumerator

MAPLE_PDSCH_SYMBOL_INT Index of symbol interrupt at maple_pdsch_ch_open_params_t.add_int.
MAPLE_PDSCH_ACK_INT Index of symbol interrupt at maple_pdsch_ch_open_params_t.add_int.
MAPLE_PDSCH_NUM_ADD_INT Total number of additional PDSCH interrupts.

3.7.7.5.5.5 anonymous enum

Total number of sectors handled by PDSCH EDF.

3.7.7.5.5.6 anonymous enum

Help enumeration for accessing MPTACSs[NUM_SECTOR][x].

Enumerator

PDSCH_SSS_TGT_ANT Set antennas targeting bit mask at MPTACSs[NUM_SECTOR][PDSCH_SSS_TGT_ANT].
PDSCH_PSS_TGT_ANT Set antennas targeting bit mask at MPTACSs[NUM_SECTOR][PDSCH_PSS_TGT_ANT].
PDSCH_ALT_DBW_PARAM Set antennas targeting bit mask at MPTACSs[NUM_SECTOR][ADBEN + ALT_DBW].
PDSCH_POS_TGT_ANT Set antennas targeting bit mask at MPTACSs[NUM_SECTOR][PDSCH_POS_TGT_ANT].
PDSCH_MBSFN_TGT_ANT Set antennas targeting bit mask at MPTACSs[NUM_SECTOR][PDSCH_MBSFN_TGT_ANT].

3.7.7.5.5.7 enum pdsch_mode_alt_dbw_t

Enumerator

PDSCH_MODE_ALT_DBW_72_SAMPLES 72 Samples
PDSCH_MODE_ALT_DBW_180_SAMPLES 180 Samples
PDSCH_MODE_ALT_DBW_512_SAMPLES 512 Samples
PDSCH_MODE_ALT_DBW_1024_SAMPLES 1024 Samples
PDSCH_MODE_ALT_DBW_1536_SAMPLES 1536 Samples
PDSCH_MODE_ALT_DBW_2048_SAMPLES 2048 Samples

3.7.7.5.5.8 anonymous enum

Enumerator

PDSCH_SECTOR_BA_ALOCATION_0 The sector is allocated starting Address 0 ... (SR + 1) (SECTOR_RES_ALOCATION). Two sectors are not allowed to share the same resources.

PDSCH_SECTOR_BA_ALOCATION_1 The sector is allocated starting Address 1 ... 1 + (SR + 1) (SECTOR_RES_ALOCATION). Two sectors are not allowed to share the same resources.

PDSCH_SECTOR_BA_ALOCATION_2 The sector is allocated starting Address 2 ... 2 + (SR + 1) (SECTOR_RES_ALOCATION). Two sectors are not allowed to share the same resources.

PDSCH_SECTOR_BA_ALOCATION_3 The sector is allocated starting Address 3 ... 3 + (SR + 1) (SECTOR_RES_ALOCATION). Two sectors are not allowed to share the same resources.

PDSCH_SECTOR_BA_ALOCATION_4 The sector is allocated starting Address 4 ... 4 + (SR + 1) (SECTOR_RES_ALOCATION). Two sectors are not allowed to share the same resources.

PDSCH_SECTOR_BA_ALOCATION_5 The sector is allocated starting Address 5 ... 5 + (SR + 1) (SECTOR_RES_ALOCATION). Two sectors are not allowed to share the same resources.

PDSCH_SECTOR_BA_ALOCATION_6 The sector is allocated starting Address 6 ... 6 + (SR + 1) (SECTOR_RES_ALOCATION). Two sectors are not allowed to share the same resources.

PDSCH_SECTOR_BA_ALOCATION_7 The sector is allocated starting Address 7 ... 7 + (SR + 1) (SECTOR_RES_ALOCATION). Two sectors are not allowed to share the same resources.

3.7.7.5.5.9 anonymous enum

Enumerator

PDSCH_SECTOR_RES_ALOCATION_0 The sector is allocated maximum 10 users, 10 UE Specific RS tokens & 7 KB of beam forming coefficients per slot.

PDSCH_SECTOR_RES_ALOCATION_1 The sector is allocated maximum 55 users, 50 UE Specific RS tokens & 14 KB of beam forming coefficients per slot.

PDSCH_SECTOR_RES_ALOCATION_2 The sector is allocated maximum 100 users, 75 UE Specific RS tokens & 21 KB of beam forming coefficients per slot.

PDSCH_SECTOR_RES_ALOCATION_3 The sector is allocated maximum 145 users, 100 UE Specific RS tokens & 28 KB of beam forming coefficients per slot.

PDSCH_SECTOR_RES_ALOCATION_4 The sector is allocated maximum 190 users, 125 UE Specific RS tokens & 35 KB of beam forming coefficients per slot.

PDSCH_SECTOR_RES_ALOCATION_5 The sector is allocated maximum 235 users, 150 UE Specific RS tokens & 42 KB of beam forming coefficients per slot.

PDSCH_SECTOR_RES_ALOCATION_6 The sector is allocated maximum 280 users, 175 UE Specific RS tokens & 49 KB of beam forming coefficients per slot.

PDSCH_SECTOR_RES_ALOCATION_7 The sector is allocated maximum 325 users, 200 UE Specific RS tokens & 56 KB of beam forming coefficients per slot.

3.7.7.5.5.10 anonymous enum

Sector's Resources allocation.

Number of Cell Specific Reference Signal Ports for sector NUM_SECTOR ; CS_RS_PORTS

Enumerator

PDSCH_NUM_CELL_REF_PORTS_1 1 Cell Specific Reference Signal Port.
PDSCH_NUM_CELL_REF_PORTS_2 2 Cell Specific Reference Signal Ports.
PDSCH_NUM_CELL_REF_PORTS_4 4 Cell Specific Reference Signal Ports.

3.7.7.5.5.11 anonymous enum

This field specifies the number of eFTPE2 engines that are used by the PDSCH2_EDF.

Enumerator

PDSCH_USE_1_FTPE eFTPE2_0 is used for PDSCH2_EDF processing.
PDSCH_USE_2_FTPE eFTPE0, eFTPE1 and eFTPE2 are used for PDSCH2_EDF processing.
PDSCH_USE_3_FTPE eFTPE0, eFTPE1 and eFTPE2 are used for PDSCH2_EDF processing.

3.7.7.5.5.12 anonymous enum

Sector NUM_SECTOR Number Of PBCH Antenna Ports.

Enumerator

PDSCH_NUM_PBCH_ANT_PORTS_2 1 or 2 PBCH Antenna Ports should be taken into account when processing positioning reference signals.
PDSCH_NUM_PBCH_ANT_PORTS_4 4 PBCH Antenna Ports should be taken into account when processing positioning reference signals.

3.7.7.5.5.13 anonymous enum

Number of virtual Antennas.

Enumerator

PDSCH_NUM_VIRT_ANT_1 1 virtual antenna.
PDSCH_NUM_VIRT_ANT_2 2 virtual antennas.
PDSCH_NUM_VIRT_ANT_4 4 virtual antennas.
PDSCH_NUM_VIRT_ANT_8 8 virtual antennas.

3.7.7.5.5.14 anonymous enum

Number of physical Antennas.

Enumerator

PDSCH_NUM_PHYS_ANT_1 1 Physical antenna.
PDSCH_NUM_PHYS_ANT_2 2 Physical antennas.
PDSCH_NUM_PHYS_ANT_4 4 Physical antennas.
PDSCH_NUM_PHYS_ANT_8 8 Physical antennas.

3.7.7.5.5.15 enum pdsch_mode_sys_bw_t

This field specifies the System Bandwidth of Sector NUM_SECTOR.

Enumerator

PDSCH_MODE_BW_1_4MHZ 1.4 MHZ (15 RBs and 180 SC)
PDSCH_MODE_BW_3MHZ 3 MHZ (15 RBs and 180 SC)
PDSCH_MODE_BW_5MHZ 5 MHZ (25 RBs and 300 SC)
PDSCH_MODE_BW_10MHZ 10 MHZ (50 RBs and 600 SC)
PDSCH_MODE_BW_15MHZ 15 MHz (75 RBs and 900 SC)
PDSCH_MODE_BW_20MHZ 20 MHZ (100 RBs and 1200 SC)

3.7.7.5.6 Function Documentation

3.7.7.5.6.1 INLINE uint32_t PDSCH_SECTOR_BA (uint8_t *S_BA*)

calculates S_BA value for the sector

Parameters

in	*S_BA*	- Sector Base Address - specifies the starrting location of the resources allocated to the sector. A sector is allocated SR+1 resources and their indexes are S_BA, S_BA+1... S_BA+SR. Two sectors are not allowed to share the same resources

Returns

32 bits word with S_BA calculated value.

3.7.7.5.6.2 os_status maplePdschInitialize (maple_pdsch_init_params_s * *init_params,* unsigned int *num_devices,* os_status(*)(void *channel, void *jobs, int *num_jobs) *channel_dispatch,* void(*)(void *channel, void *maple) *channel_reap*)

Initializes the PDSCH driver's structures

Parameters

in	*init_params*	- PDSCH Initialization parameters

in	*num_devices*	- Number of PDSCH devices
in	*channel_dispatch*	- Pointer to channel dispatch function
in	*channel_reap*	- Pointer to channel reap function

Returns

OS_SUCCESS

Warning

This function is generally called by osArchInitialize() as part of the kernel and drivers

3.7.7.6 PDSCH Runtime

3.7.7.6.1 Overview

PDSCH Runtime API

Data Structures

- struct pdsch_sec_reconfig_param_t
- struct maple_pdsch_alt_dbw_conf_t

Macros

- #define PDSCH_DEV_CMD_RECONF_SEC_MASK (0xF0F30F7F)

MAPLE PDSCH Device Control Commands.

- #define PDSCH_DEV_CMD_RECONF_SEC (0x00000100 | COP_LLD_COMMAND)
- #define PDSCH_DEV_CMD_RECONF_SEC_NON_BLOCKING (PDSCH_DEV_CMD_RECONF_SEC | COP_NON_BLOCKING_COMMAND)
- #define PDSCH_DEV_CMD_FULL_RECONF_SEC (0x00000200 | COP_LLD_COMMAND)
- #define PDSCH_DEV_CMD_FULL_RECONF_SEC_NON_BLOCKING (PDSCH_DEV_CMD_FULL_RECONF_SEC | COP_NON_BLOCKING_COMMAND)
- #define PDSCH_DEV_CMD_PAD_DATA_SET (0x00000300 | COP_LLD_COMMAND)
- #define PDSCH_DEV_CMD_PAD_DATA_SET_NON_BLOCKING (PDSCH_DEV_CMD_PAD_DATA_SET | COP_NON_BLOCKING_COMMAND)
- #define PDSCH_DEV_CMD_DBG_GET_SECTOR_QUEUES (0x00000400 | COP_LLD_COMMAND)
- #define PDSCH_DEV_CMD_SET_CELL_CONFIG_0 (0x00000500 | COP_LLD_COMMAND)
- #define PDSCH_DEV_CMD_SET_CELL_CONFIG_1 (0x00000600 | COP_LLD_COMMAND)
- #define PDSCH_DEV_CMD_CONFIG_ALT_DBW (0x00000700 | COP_LLD_COMMAND)
- #define PDSCH_DEV_CMD_DBG_GET_SECTOR_STATUS (0x00000800 | COP_LLD_COMMAND)
- #define PDSCH_DEV_CMD_PAD_DATA1_SET (0x00000900 | COP_LLD_COMMAND)
- #define PDSCH_DEV_CMD_PAD_DATA2_SET (0x00000a00 | COP_LLD_COMMAND)

MAPLE PDSCH Channel Control Commands

- #define PDSCH_CH_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL
- #define PDSCH_CH_CMD_VIRT_TRANS_ENABLE MAPLE_PE_CH_CMD_VIRT_TRANS↩ _ENABLE
- #define PDSCH_CH_CMD_VIRT_TRANS_DISABLE MAPLE_PE_CH_CMD_VIRT_TRANS↩ _DISABLE
- #define PDSCH_CH_CMD_GO MAPLE_PE_CH_CMD_GO
- #define PDSCH_CH_CMD_INFO_GET MAPLE_PE_CH_INFO_GET
- #define PDSCH_CH_CMD_GET_NUM_SYMS_DONE (0x00000001 | COP_LLD_COMMAND)
- #define PDSCH_CH_CMD_UPDATE_BEFORE_LATE_PCR (0x00000002 | COP_LLD_COM↩ MAND)
- #define PDSCH_CH_CMD_UPDATE_BEFORE_EXT (0x00000003 | COP_LLD_COMMAND)

3.7.7.6.2 Data Structure Documentation

3.7.7.6.2.1 struct pdsch_sec_reconfig_param_t

PDSCH sector reconfiguration parameters.

Data Fields

- pdsch_config_param_t cell_config
- uint32_t sec_num: 8
- uint32_t ant_out_size
- void * antx_out_addr [MAPLE_NUM_ANT]
- uint32_t * bf_config
- uint8_t sss_tgt
- uint8_t pss_tgt
- uint16_t alt_dbw_en: 1
- uint16_t alt_dbw: 3
- uint16_t alt_bw_conf
- void * dnosiba
- uint32_t dnosid
- uint8_t pos_tgt
- uint8_t mbsfn_tgt
- uint32_t sr: 3
- uint32_t s_ba: 3
- uint32_t cs_rs_ports: 2
- uint32_t cyclic_out: 1
- uint32_t efsel: 2
- uint32_t pbch_ports: 1
- uint32_t s_nva: 3
- uint32_t s_npa: 3
- uint32_t s_bfe: 1
- uint32_t pdsch_api

3.7.7.6.2.1.1 Field Documentation

3.7.7.6.2.1.2 pdsch_config_param_t pdsch_sec_reconfig_param_t::cell_config

Sector configuration.

3.7.7.6.2.1.3 uint32_t pdsch_sec_reconfig_param_t::sec_num

0 - 3 number indicating sector number to be updated

3.7.7.6.2.1.4 uint32_t pdsch_sec_reconfig_param_t::ant_out_size

Antenna Output buffer size.

3.7.7.6.2.1.5 void* pdsch_sec_reconfig_param_t::antx_out_addr[MAPLE_NUM_ANT]

Base addresses of the output buffer of antenna; Driver won't translate these addresses.

3.7.7.6.2.1.6 uint8_t pdsch_sec_reconfig_param_t::sss_tgt

Set antennas targeting bit mask at MPTACSs[NUM_SECTOR][PDSCH_SSS_TGT_ANT].

3.7.7.6.2.1.7 uint8_t pdsch_sec_reconfig_param_t::pss_tgt

Set antennas targeting bit mask at MPTACSs[NUM_SECTOR][PDSCH_PSS_TGT_ANT].

3.7.7.6.2.1.8 uint16_t pdsch_sec_reconfig_param_t::alt_dbw_en

ALT_DBW Enable - This bit determines whether S_BW or ALT_DBW determines how many input samples are going to be read for IFFT operation as opposed to how many inputs samples are complemented as guardbands.

3.7.7.6.2.1.9 uint16_t pdsch_sec_reconfig_param_t::alt_dbw

ALT_DBW - If ALT_DBW is enabled, This bit determines how many input samples are going to be read for IFFT operation as opposed to how many inputs samples are complemented as guardbands.

can be used with pdsch_mode_alt_dbw_t enum.

3.7.7.6.2.1.10 uint16_t pdsch_sec_reconfig_param_t::alt_bw_conf

ALT_DBW - This bit fields determines whether S_BW or ALT_DBW determines how many input samples are going to be read for IFFT operation as opposed to how many inputs samples are complemented as guardbands.

3.7.7.6.2.1.11 void* pdsch_sec_reconfig_param_t::dnosiba

DNOS Interrupt Base Address.

3.7.7.6.2.1.12 uint32_t pdsch_sec_reconfig_param_t::dnosid

DNOS Interrupt data.

3.7.7.6.2.1.13 uint8_t pdsch_sec_reconfig_param_t::pos_tgt

Set antennas targeting bit mask at MPTACSs[NUM_SECTOR][PDSCH_POS_TGT_ANT].

3.7.7.6.2.1.14 uint8_t pdsch_sec_reconfig_param_t::mbsfn_tgt

Set antennas targeting bit mask at MPTACSs[NUM_SECTOR][PDSCH_MBSFN_TGT_ANT].

3.7.7.6.2.1.15 uint32_t pdsch_sec_reconfig_param_t::sr

Sector Resources.

3.7.7.6.2.1.16 uint32_t pdsch_sec_reconfig_param_t::s_ba

Sector Base Address.

This field specifies the starting location of the resources allocated to the sector. A sector is allocated SR+1 resources and their indexes are S_BA, S_BA+1... S_BA+SR. Two sectors are not allowed to share the same resources.

3.7.7.6.2.1.17 uint32_t pdsch_sec_reconfig_param_t::cs_rs_ports

Number of Cell Specific Reference Signal Ports.

3.7.7.6.2.1.18 uint32_t pdsch_sec_reconfig_param_t::cyclic_out

CYCLIC_OUT Enable cyclic output buffer.

3.7.7.6.2.1.19 uint32_t pdsch_sec_reconfig_param_t::efsel

This field specifies which eFTPE2 engine is used by the PDSCH2_EDF.

0 Reserved 1 eFTPE0 is used 2 eFTPE0 & eFTPE1 are used. 3 eFTPE0, eFTPE1 and eFTPE2 are used.

3.7.7.6.2.1.20 uint32_t pdsch_sec_reconfig_param_t::pbch_ports

Number Of PBCH Antenna Ports.

3.7.7.6.2.1.21 uint32_t pdsch_sec_reconfig_param_t::s_nva

Number of virtual antennas.

3.7.7.6.2.1.22 uint32_t pdsch_sec_reconfig_param_t::s_npa

Number of physical antennas.

3.7.7.6.2.1.23 uint32_t pdsch_sec_reconfig_param_t::s_bfe

Sector Beam forming enable.

3.7.7.6.2.1.24 uint32_t pdsch_sec_reconfig_param_t::pdsch_api

MAPLE PDSCH2_EDF sector API configuration parameter.

3.7.7.6.2.1.25 uint32_t* pdsch_sec_reconfig_param_t::bf_config

Beam Forming Configuration - Pointer to an array the size of [MAPLE_PDSCH_BF_CONFIG_LAST].

3.7.7.6.2.2 struct maple_pdsch_alt_dbw_conf_t

PDSCH ALT Downling BW reconfiguration parameters.

Data Fields

- uint8_t sec_num
- uint8_t cell_set
- uint16_t alt_dbw_en: 1
- uint16_t alt_dbw: 3
- uint16_t alt_dbw_conf

3.7.7.6.2.2.1 Field Documentation

3.7.7.6.2.2.2 uint16_t maple_pdsch_alt_dbw_conf_t::alt_dbw_en

ALT_DBW Enable - This bit determines whether S_BW or ALT_DBW determines how many input samples are going to be read for IFFT operation as opposed to how many inputs samples are complemented as guardbands.

3.7.7.6.2.2.3 uint16_t maple_pdsch_alt_dbw_conf_t::alt_dbw

ALT_DBW - If ALT_DBW is enabled, This bit determines how many input samples are going to be read for IFFT operation as opposed to how many inputs samples are complemented as guardbands.

can be used with pdsch_mode_alt_dbw_t enum.

3.7.7.6.2.2.4 uint16_t maple_pdsch_alt_dbw_conf_t::alt_dbw_conf

ALT_DBW - This bit fields determines whether S_BW or ALT_DBW determines how many input samples are going to be read for IFFT operation as opposed to how many inputs samples are complemented as guardbands.

3.7.7.6.2.2.5 uint8_t maple_pdsch_alt_dbw_conf_t::sec_num

0 - 3 number indicating sector number to configure.

3.7.7.6.2.2.6 uint8_t maple_pdsch_alt_dbw_conf_t::cell_set

0/1 - Indicating which cell set to configure.

3.7.7.6.3 Macro Definition Documentation

3.7.7.6.3.1 #define PDSCH_DEV_CMD_RECONF_SEC (0x00000100 | COP_LLD_COMMAND)

PDSCH sector x reconfiguration, configure only cell configuration, pass pointer to pdsch_sec_reconfig_param_t as parameter and uses only cell_config field.

3.7.7.6.3.2 #define PDSCH_DEV_CMD_RECONF_SEC_NON_BLOCKING (PDSCH_DEV_CMD_RECONF_SEC | COP_NON_BLOCKING_COMMAND)

PDSCH sector x reconfiguration, tries to configure only cell configuration, pass pointer to pdsch_sec_reconfig_param_t as parameter and uses only cell_config field.

3.7.7.6.3.3 #define PDSCH_DEV_CMD_FULL_RECONF_SEC (0x00000200 | COP_LLD_COMMAND)

PDSCH sector x reconfiguration, configure entire sector, pass pointer to pdsch_sec_reconfig_param_t as parameter.

3.7.7.6.3.4 #define PDSCH_DEV_CMD_FULL_RECONF_SEC_NON_BLOCKING (PDSCH_DEV_CMD_FULL_RECONF_SEC | COP_NON_BLOCKING_COMMAND)

PDSCH sector x reconfiguration, tries to configure entire sector, pass pointer to pdsch_sec_reconfig_param_t as parameter.

3.7.7.6.3.5 #define PDSCH_DEV_CMD_PAD_DATA_SET (0x00000300 | COP_LLD_COMMAND)

Update MAPLE PDSCH PAD Data Configuration register, pass pointer to 4 bytes pattern as parameter.

3.7.7.6.3.6 #define PDSCH_DEV_CMD_PAD_DATA_SET_NON_BLOCKING (PDSCH_DEV_CMD_PAD_DATA_SET | COP_NON_BLOCKING_COMMAND)

Tries to update MAPLE PDSCH PAD Data Configuration register, pass pointer to uint32_t as parameter.

Fails if PDSCH spinlock is already taken by another core

3.7.7.6.3.7 #define PDSCH_DEV_CMD_DBG_GET_SECTOR_QUEUES (0x00000400 | COP_LLD_COMMAND)

Gets pdsch_dbg_queues_param_t as a parameter and writes queues specified in it to the PDSCH sector x Queues accordingly.

3.7.7.6.3.8 #define PDSCH_DEV_CMD_SET_CELL_CONFIG_0 (0x00000500 | COP_LLD_COMMAND)

Configure Cell configuration set #0 of the cell given as a parameter.

3.7.7.6.3.9 #define PDSCH_DEV_CMD_SET_CELL_CONFIG_1 (0x00000600 | COP_LLD_COMMAND)

Configure Cell configuration set #1 of the cell given as a parameter - expects pdsch_sec_reconfig_param_t * as a parameter.

3.7.7.6.3.10 #define PDSCH_DEV_CMD_CONFIG_ALT_DBW (0x00000700 | COP_LLD_COMMAND)

Configure Cell configuration set #1 of the cell given as a parameter - expects maple_pdsch_alt_dbw_conf_t * as a parameter.

3.7.7.6.3.11 #define PDSCH_DEV_CMD_DBG_GET_SECTOR_STATUS (0x00000800 | COP_LLD_COMMAND)

Retrieves PDSCH sector x status to the pdsch_dbg_status_param_t parameter - expects pdsch_sec_reconfig_param_t * as a parameter.

3.7.7.6.3.12 #define PDSCH_DEV_CMD_PAD_DATA1_SET (0x00000900 | COP_LLD_COMMAND)

Update MAPLE PDSCH PAD Data 1 Configuration register, pass pointer to 4 bytes pattern as parameter.

3.7.7.6.3.13 #define PDSCH_DEV_CMD_PAD_DATA2_SET (0x00000a00 | COP_LLD_COMMAND)

Update MAPLE PDSCH PAD Data 2 Configuration register, pass pointer to 4 bytes pattern as parameter.

3.7.7.6.3.14 #define PDSCH_CH_CMD_RX_POLL MAPLE_PE_CH_CMD_RX_POLL

Polling the channel for finished jobs.

3.7.7.6.3.15 #define PDSCH_CH_CMD_VIRT_TRANS_ENABLE MAPLE_PE_CH_CMD_VIRT_TRANS_ENABLE

Update channel translation mode: enable virtual to physical translation.

3.7.7.6.3.16 #define PDSCH_CH_CMD_VIRT_TRANS_DISABLE MAPLE_PE_CH_CMD_VIRT_TRANS_DISABLE

Update channel translation mode: disable virtual to physical translation.

3.7.7.6.3.17 #define PDSCH_CH_CMD_GO MAPLE_PE_CH_CMD_GO

Go function for channel that have been opened with use_go_function flag; It sets OWNER bit on the first dispatched job thus triggers maple to start processing this batch.

3.7.7.6.3.18 #define PDSCH_CH_CMD_INFO_GET MAPLE_PE_CH_INFO_GET

Get channel handle information; helpful for debug.

3.7.7.6.3.19 #define PDSCH_CH_CMD_GET_NUM_SYMS_DONE (0x00000001 | COP_LLD_COMMAND)

Returns the number of symbols done for the given a PDSCH job in the given channel.

params should be a pointer to pdsch_job_t

3.7.7.6.3.20 #define PDSCH_CH_CMD_UPDATE_BEFORE_LATE_PCR (0x00000002 | COP_LLD_COMMAND)

Updates a job before issuing a LATE PCR host command.

params should be a pointer to an updated pdsch_job.

3.7.7.6.3.21 #define PDSCH_CH_CMD_UPDATE_BEFORE_EXT (0x00000003 | COP_LLD_COMMAND)

Updates a job before issuing a EXT SYMBOL START PCR host command.

params should be a pointer to an updated pdsch_job.

3.7.7.6.3.22 #define PDSCH_DEV_CMD_RECONF_SEC_MASK (0xF0F30F7F)

PDSCH sector x reconfiguration, pass pointer to pdsch_sec_reconfig_param_t as parameter.

3.7.7.7 VSG Memory Map

3.7.7.7.1 Overview

```
Virtual Interrupts, HW Semaphores, GPIO Memory maps.
```

Data Structures

- struct vint_map_t
- struct hs_map_t
- struct gpio_map_t

3.7.7.7.2 Data Structure Documentation

3.7.7.7.2.1 struct vint_map_t

Virtual Interrupts Generator (GIC) Registers.

Data Fields

- volatile uint32_t vigr
- volatile uint32_t visr

3.7.7.7.2.1.1 Field Documentation

3.7.7.7.2.1.2 volatile uint32_t vint_map_t::vigr

Virtual Interrupt Generation Register.

3.7.7.7.2.1.3 volatile uint32_t vint_map_t::visr

Virtual Interrupt Status Register.

3.7.7.7.2.2 struct hs_map_t

Hardware Semaphore Registers.

Data Fields

- volatile uint32_t hsmpr

3.7.7.7.2.2.1 Field Documentation

3.7.7.7.2.2.2 volatile uint32_t hs_map_t::hsmpr

Hardware Semaphore Register, 8 LST bits.

3.7.7.7.2.3 struct gpio_map_t

General Purpose I/O (GPIO) Registers.

Data Fields

- volatile uint32_t podr
- volatile uint32_t pdat
- volatile uint32_t pdir
- volatile uint32_t par
- volatile uint32_t psor

3.7.7.7.2.3.1 Field Documentation

3.7.7.7.2.3.2 volatile uint32_t gpio_map_t::podr

Pin Open-Drain Register.

3.7.7.7.2.3.3 volatile uint32_t gpio_map_t::pdat

Pin Data Register.

3.7.7.7.2.3.4 volatile uint32_t gpio_map_t::pdir

Pin Data Direction Registers.

3.7.7.7.2.3.5 volatile uint32_t gpio_map_t::par

Pin Assignment Register.

3.7.7.7.2.3.6 volatile uint32_t gpio_map_t::psor

Pin Special Options Registers.

3.7.7.8 B486x Heterogeneous API

3.7.7.8.1 Overview

Data Structures

- struct os_het_control_t

Macros

- #define OS_HET_IPC_HW_SEMAPHORE_NUM 0
- #define OS_HET_BOOT_HW_SEMAPHORE_NUM 1

Variables

- os_het_control_t * g_os_het_control

3.7.7.8.2 Data Structure Documentation

3.7.7.8.2.1 struct os_het_control_t

Heterogeneous OS global control structure.

Data Fields

- uint32_t start_validation_value
- os_het_init_t initialized
- os_het_mem_t pa_shared_mem
- os_het_mem_t sc_shared_mem
- het_phys_ptr ipc
- het_phys_ptr l1_defense
- het_phys_ptr smartdsp_debug
- os_het_debug_print_t het_debug_print
- uint32_t shared_ctrl_size
- uint32_t num_ipc_regions
- uint32_t end_validation_value

3.7.7.8.2.1.1 Field Documentation

3.7.7.8.2.1.2 uint32_t os_het_control_t::start_validation_value

validation value for checking for corrpution in case of reset

3.7.7.8.2.1.3 os_het_init_t os_het_control_t::initialized

Initialization indication strcuture.

3.7.7.8.2.1.4 os_het_mem_t os_het_control_t::pa_shared_mem

SET BY PA: PA shared memory region;.

3.7.7.8.2.1.5 os_het_mem_t os_het_control_t::sc_shared_mem

SET BY DSP: SC shared memory region;.

3.7.7.8.2.1.6 het_phys_ptr os_het_control_t::ipc

Pointer to IPC heterogeneous structure os_het_ipc_t.

3.7.7.8.2.1.7 het_phys_ptr os_het_control_t::l1_defense

Physical pointer to L1 defense heterogeneous structure - os_het_l1_defense_t.

3.7.7.8.2.1.8 het_phys_ptr os_het_control_t::smartdsp_debug

SET BY PA: Pointer to where SmartDSP logs system (os_het_smartdsp_log_t); PA initializes an array with the number of entries as there is SC cores.

3.7.7.8.2.1.9 os_het_debug_print_t os_het_control_t::het_debug_print

a pointer to SmartDSP Debug print structure

3.7.7.8.2.1.10 uint32_t os_het_control_t::shared_ctrl_size

SET BY DSP: Size of the shared memory for control information in bytes - Mimumum size is 4 KB.

3.7.7.8.2.1.11 uint32_t os_het_control_t::num_ipc_regions

Number of IPC regions - this parameter is only for multimode usages, and should be accessed ONLY if initialized.pa_initialized equals OS_HET_INITIALIZED_MULTIMODE.

3.7.7.8.2.1.12 uint32_t os_het_control_t::end_validation_value

validation value for checking for corrpution in case of reset

3.7.7.8.3 Macro Definition Documentation

3.7.7.8.3.1 #define OS_HET_IPC_HW_SEMAPHORE_NUM 0

Hardware semaphore to use in case of need for mutual exclusion in the IPC module.

3.7.7.8.3.2 #define OS_HET_BOOT_HW_SEMAPHORE_NUM 1

Hardware semapohore for synchronizing the boot processes between SC and PA.

3.7.7.8.4 Variable Documentation

3.7.7.8.4.1 os_het_control_t* g_os_het_control

Pointer to the base address of the heterogeneous OS control strcuture.

3.7.7.9 B486x Heterogeneous Common API

3.7.7.9.1 Overview

Data Structures

- struct os_het_init_t
- struct os_het_tracker_t

Macros

- #define FALSE (0)
- #define TRUE (1)
- #define OS_HET_CALCULATE_ADDR(BASE, OFFSET) (void *)((uint8_t *)(BASE) + (uint32_t)(OFFSET))

Initialization indicaiton values

- #define OS_HET_INITIALIZED 0xFEDCBA98UL
- #define OS_HET_INITIALIZED_MULTIMODE 0xEDCBA987UL
- #define OS_HET_UNINITIALIZED 0x00000000UL

Hardware/Software semaphore values

- #define OS_HET_PA_SEMAPHORE_VAL 0xFF
- #define OS_HET_SC_SEMAPHORE_VAL 0xFE
- #define OS_HET_FREE_SEMAPHORE_VAL 0x00

3.7.7.9.2 Data Structure Documentation

3.7.7.9.2.1 struct os_het_init_t

Initialization Control structure.

This structure will be used to indicatye that both OS domains have initialized a specific structure

Data Fields

- uint32_t pa_initialized
- uint32_t sc_initialized

3.7.7.9.2.1.1 Field Documentation

3.7.7.9.2.1.2 uint32_t os_het_init_t::pa_initialized

Indicates whether the overall control structure is initialized (PA side)

3.7.7.9.2.1.3 uint32_t os_het_init_t::sc_initialized

Indicates whether the overall control structure is initialized (SC side)

3.7.7.9.2.2 struct os_het_tracker_t

Producer/Consumer tracker.

The producer and consumer each perform counter++ to their counter. It is assumed the size of what the tracker is tracking is less than < MAX_UINT_32

Warning

The counters must only be incremented, never decremented

Data Fields

- uint32_t producer_num
- uint32_t consumer_num

3.7.7.9.2.2.1 Field Documentation

3.7.7.9.2.2.2 uint32_t os_het_tracker_t::producer_num

Number of items the producer produced.

3.7.7.9.2.2.3 uint32_t os_het_tracker_t::consumer_num

Number of items the consumer consumed.

3.7.7.9.3 Macro Definition Documentation

3.7.7.9.3.1 #define FALSE (0)

Boolean false.

3.7.7.9.3.2 #define TRUE (1)

Boolean true.

3.7.7.9.3.3 #define OS_HET_INITIALIZED 0xFEDCBA98UL

The resource is initializaed.

3.7.7.9.3.4 #define OS_HET_INITIALIZED_MULTIMODE 0xEDCBA987UL

The resource is initializaed for multimode mode - to be used ONLY for control channel "os_het_init_t initialized" field.

3.7.7.9.3.5 #define OS_HET_UNINITIALIZED 0x00000000UL

The resource is uninitialized.

3.7.7.9.3.6 #define OS_HET_PA_SEMAPHORE_VAL 0xFF

Sempahore is taken by the PA domain.

3.7.7.9.3.7 #define OS_HET_SC_SEMAPHORE_VAL 0xFE

Sempahore is taken by the SC domain.

3.7.7.9.3.8 #define OS_HET_FREE_SEMAPHORE_VAL 0x00

Semaphore is free.

3.7.7.9.3.9 #define OS_HET_CALCULATE_ADDR(*BASE, OFFSET*) (void ∗)((uint8_t ∗)(BASE) + (uint32_t)(OFFSET))

Used by the various OS to calculate an address in it’s own virtual address space.

3.7.7.10 B486x Heterogeneous Debug API

3.7.7.10.1 Overview

Data Structures

- struct os_het_smartdsp_log_t

3.7.7.10.2 Data Structure Documentation

3.7.7.10.2.1 struct os_het_smartdsp_log_t

SmartDSP Event log.

Data Fields

- uint32_t start_validation_value
- het_phys_ptr base_address
- uint32_t size
- het_phys_ptr last_error
- uint32_t end_validation_value

3.7.7.10.2.1.1 Field Documentation

3.7.7.10.2.1.2 uint32_t os_het_smartdsp_log_t::start_validation_value

validation value for checking for corrpution in case of reset

3.7.7.10.2.1.3 het_phys_ptr os_het_smartdsp_log_t::base_address

Pointer to the core's event log.

3.7.7.10.2.1.4 uint32_t os_het_smartdsp_log_t::size

Size of the event log (in bytes)

3.7.7.10.2.1.5 het_phys_ptr os_het_smartdsp_log_t::last_error

Pointer to the last error in SmartDSP.

A value of 1 (OS_SUCCESS) means no error

3.7.7.10.2.1.6 uint32_t os_het_smartdsp_log_t::end_validation_value

validation value for checking for corrpution in case of reset

3.7.7.11 B486x Heterogeneous Debug print API

3.7.7.11.1 Overview

Data Structures

- struct os_het_debug_print_sc_t

Macros

- #define MAX_NUM_OF_SEGMENT 32

3.7.7.11.2 Data Structure Documentation

3.7.7.11.2.1 struct os_het_debug_print_sc_t

SmartDSP Debug print structure.

Data Fields

- uint64_t buffer_location
- uint32_t segment_size
- uint32_t num_of_segments
- os_het_tracker_t tracker
- debug_print_clocks_t segment_clock [MAX_NUM_OF_SEGMENT]
- uint32_t overflow

3.7.7.11.2.1.1 Field Documentation

3.7.7.11.2.1.2 uint64_t os_het_debug_print_sc_t::buffer_location

Pointer to the base address of the SC VTB.

3.7.7.11.2.1.3 uint32_t os_het_debug_print_sc_t::segment_size

Size of each segment in VTB.

3.7.7.11.2.1.4 uint32_t os_het_debug_print_sc_t::num_of_segments

Number of VTB segments.

3.7.7.11.2.1.5 os_het_tracker_t os_het_debug_print_sc_t::tracker

Tracker for segment number; SC client is the producer and PA engine is the consumer.

3.7.7.11.2.1.6 debug_print_clocks_t os_het_debug_print_sc_t::segment_clock[MAX_NUM_OF↩ _SEGMENT]

64 bits clock for each segment

3.7.7.11.2.1.7 uint32_t os_het_debug_print_sc_t::overflow

Overflow indicator.

3.7.7.11.3 Macro Definition Documentation

3.7.7.11.3.1 #define MAX_NUM_OF_SEGMENT 32

The Maximum nuber of segments of the buffer.

3.7.7.12 B486x Heterogeneous IPC API

3.7.7.12.1 Overview

Data Structures

- struct os_het_ipc_bd_t
- struct os_het_ipc_channel_t
- struct os_het_ipc_t

Macros

- #define OS_HET_CH_EMPTY(CH) ((CH)->tracker.producer_num == (CH)->tracker.consumer_num)
- #define OS_HET_CH_FREE_BDS(CH)
- #define OS_HET_CH_FULL(CH) (OS_HET_CH_FREE_BDS(CH) == 0)
- #define OS_HET_INCREMENT_INDEX(CH, INDEX) ((CH)->tracker.INDEX)++
- #define OS_HET_UNSPECIFIED_LEN 0

Enumerations

- enum os_het_ipc_ind_t { OS_HET_VIRTUAL_INT = 0, OS_HET_PA_MPIC, OS_HET_DSP_MESH, OS_HET_DSP_MPIC_MSI, OS_HET_NO_INT, OS_HET_NUM_INDICATIONS }
- enum os_het_ipc_ch_types_t { OS_HET_IPC_MESSAGE_CH = 0 }

3.7.7.12.2 Data Structure Documentation

3.7.7.12.2.1 struct os_het_ipc_bd_t

IPC message descriptor.

Data Fields

- het_phys_ptr msg_ptr
- uint32_t msg_len

3.7.7.12.2.1.1 Field Documentation

3.7.7.12.2.1.2 het_phys_ptr os_het_ipc_bd_t::msg_ptr

Pointer to the message; as an offset from the base of the shared address space.

3.7.7.12.2.1.3 uint32_t os_het_ipc_bd_t::msg_len

Size of the message; may be set to OS_HET_UNSPECIFIED_LEN on OS_HET_IPC_POINTER_CH channel types only.

3.7.7.12.2.2 struct os_het_ipc_channel_t

IPC channel control structure.

Warning

Using the macros OS_HET_INCREMENT_CONSUMER() and OS_HET_INCREMENT_PRODUCER() is the recommended way of incrementing the tracker counters; Direct access should be avoided.

Data Fields

- uint32_t start_validation_value
- uint32_t producer_initialized
- uint32_t consumer_initialized
- uint32_t id
- os_het_tracker_t tracker
- uint32_t bd_ring_size
- uint32_t max_msg_size
- os_het_ipc_ch_types_t ch_type
- het_phys_ptr bd_base
- os_het_ipc_ind_t ipc_ind
- uint32_t ind_offset
- uint32_t ind_value
- uint32_t pa_reserved [2]
- het_phys_ptr semaphore_pointer
- uint32_t end_validation_value

3.7.7.12.2.2.1 Field Documentation

3.7.7.12.2.2.2 uint32_t os_het_ipc_channel_t::start_validation_value

validation value for checking for corrpution in case of reset

3.7.7.12.2.2.3 uint32_t os_het_ipc_channel_t::producer_initialized

Indicates whether the mailbox is initialized by the producer; This field is written by the producer.

3.7.7.12.2.2.4 uint32_t os_het_ipc_channel_t::consumer_initialized

Indicates whether the mailbox is initialized by the consumer; This field is written by the consumer.

3.7.7.12.2.2.5 uint32_t os_het_ipc_channel_t::id

Mailbox ID - may be used by the applicaiton to identify the channel; This field is written by Linux during boot.

Can be an arbitrary number provided that it is unique in the system

3.7.7.12.2.2.6 os_het_tracker_t os_het_ipc_channel_t::tracker

Producer/consumer tracker; This field is set to {0} by Linux during boot.

3.7.7.12.2.2.7 uint32_t os_het_ipc_channel_t::bd_ring_size

Size of the mailbox BD ring; This field is set by the consumer.

May not be larger than os_het_ipc_t.ipc_max_bd_size

3.7.7.12.2.2.8 uint32_t os_het_ipc_channel_t::max_msg_size

Size (in Bytes) of the maximal message that can be passed on this channel; This field is set by the consumer ONLY if ch_type is OS_HET_IPC_MESSAGE_CH otherwise MUST be set to 0xFFFFFFFF.

3.7.7.12.2.2.9 os_het_ipc_ch_types_t os_het_ipc_channel_t::ch_type

Type of the channel; This field is set by the consumer.

3.7.7.12.2.2.10 het_phys_ptr os_het_ipc_channel_t::bd_base

Pointer to array of os_het_ipc_bd_t Base of the mailbox pointer BD; as an offset from the base of the shared address space; For heterogeneous channels, this field is allocated by Linux during boot based on os_het_ipc_t.ipc_max_bd_size.

3.7.7.12.2.2.11 os_het_ipc_ind_t os_het_ipc_channel_t::ipc_ind

Type of indication to generate to the destination; This field is written by the consumer.

3.7.7.12.2.2.12 uint32_t os_het_ipc_channel_t::ind_offset

Offset address from base of specific indication register set; This field is written by the consumer.

3.7.7.12.2.2.13 uint32_t os_het_ipc_channel_t::ind_value

Value to write to ind_offset; This field is written by the consumer.

3.7.7.12.2.2.14 uint32_t os_het_ipc_channel_t::pa_reserved[2]

reserved field for pa usage

3.7.7.12.2.2.15 het_phys_ptr os_het_ipc_channel_t::semaphore_pointer

Future compatibility semaphore pointer; This field is set to NULL by Linux during boot.

3.7.7.12.2.2.16 uint32_t os_het_ipc_channel_t::end_validation_value

validation value for checking for corrpution in case of reset

3.7.7.12.2.3 struct os_het_ipc_t

IPC global control structure.

Data Fields

- uint32_t start_validation_value
- uint32_t num_ipc_channels
- uint32_t ipc_max_bd_size
- het_phys_ptr ipc_channels
- uint32_t end_validation_value

3.7.7.12.2.3.1 Field Documentation

3.7.7.12.2.3.2 uint32_t os_het_ipc_t::start_validation_value

validation value for checking for corrpution in case of reset

3.7.7.12.2.3.3 uint32_t os_het_ipc_t::num_ipc_channels

Number of channels in the ipc_channels array; this is a bootarg to Linux and will be set by Linux accordingly.

3.7.7.12.2.3.4 uint32_t os_het_ipc_t::ipc_max_bd_size

Maximal size that the ipc_channel.bd_ring_size may have; this is a bootarg to Linux and will be set by Linux accordingly.

3.7.7.12.2.3.5 het_phys_ptr os_het_ipc_t::ipc_channels

Pointer to the mailboxes control structure array os_het_ipc_channel_t; has os_het_ipc_t.num_ipc_↩ channels entries as an offset from the base of the shared address space.

3.7.7.12.2.3.6 uint32_t os_het_ipc_t::end_validation_value

validation value for checking for corrpution in case of reset

3.7.7.12.3 Macro Definition Documentation

3.7.7.12.3.1 #define OS_HET_CH_EMPTY(*CH*) ((CH)->tracker.producer_num == (CH)->tracker.consumer_num)

Evaluate whether the channel is empty.

3.7.7.12.3.2 #define OS_HET_CH_FREE_BDS(*CH*)

Value:

```
(( ((CH)->tracker.producer_num >= (CH)->tracker.consumer_num)) ?    \
      ((CH)->bd_ring_size - ((CH)->tracker.producer_num - (CH)->tracker.consumer_num)) : \
      ((CH)->bd_ring_size - ((CH)->tracker.producer_num - (CH)->tracker.consumer_num + MAX_UINT_32 + 1)))
```

Evaluate the number of free BD in the channel.

3.7.7.12.3.3 #define OS_HET_CH_FULL(*CH*) (OS_HET_CH_FREE_BDS(CH) == 0)

Evaluate whether the channel is full.

3.7.7.12.3.4 #define OS_HET_INCREMENT_INDEX(*CH, INDEX*) ((CH)->tracker.INDEX)++

Increment the producer/consumer index.

3.7.7.12.3.5 #define OS_HET_UNSPECIFIED_LEN 0

Use to indicate that the message consists of a pointer only.

3.7.7.12.4 Enumeration Type Documentation

3.7.7.12.4.1 enum os_het_ipc_ind_t

Types of indications.

The IPC driver should use this to calculate the base address from which to generate the indication

Enumerator

OS_HET_VIRTUAL_INT Type of indication is virtual interrupt; channel offset reletive to GIC.
OS_HET_PA_MPIC Type of indication is MPIC message; channel offset reletive to MPIC.
OS_HET_DSP_MESH Type of indication is by DSP mesh; channel offset reletive DSP subsystem address.
OS_HET_DSP_MPIC_MSI Type of indication is by MPIC MSI; channel offset reletive to MPIC.
OS_HET_NO_INT Type of indication is none.
OS_HET_NUM_INDICATIONS MUST BE LEFT LAST.

3.7.7.12.4.2 enum os_het_ipc_ch_types_t

Types of IPC channels.

Enumerator

OS_HET_IPC_MESSAGE_CH Indicates that the consumer will pre-populate the channel's ptr_↩ bd_base[].msg_ptr entires; The producer MUST use the pre-populated pointers when producing messages on the channel.

3.7.7.13 B486x L1 defence heterogeneous API

3.7.7.13.1 Overview

Data Structures

- struct os_het_l1_defense_t

Enumerations

- enum os_het_l1d_mode_t { , MODE_1_ACTIVE = 0x1, MODE_2_ACTIVE = 0x2, MODE_3_↩ ACTIVE = 0x4 }
- enum os_het_l1d_reset_maple_t { RESET_MAPLE_NONE = 0x0, RESET_MAPLE_1 = 0x2, RESET_MAPLE_2 = 0x4, RESET_MAPLE_3 = 0x8 }
- enum os_het_l1d_status { OS_HET_ERR_L1D_MEMORY_CORRUPTED = 0x0037FFBF, OS_HET_ERR_L1D_MODE↩ _INVALID = 0x0037FFED, OS_HET_ERR_L1D_FUNCTION_INVALID = 0x0037FFEC, OS_HET_ERR_L1D_ALREADY_ACTIVE = 0x0037FFC3, OS_HET_ERR_L1D_UNKNOWN = 0x00370002, OS_HET_INFO_L1D_READY_FOR_RESET = 0x1037FFBE, BEGIN_WARM_RESET_OS_INIT = 0x1037FFBD, END_OF_WARM_RESET_OS_INIT = 0x1037FFBC, WARM_RESET_SUCCESS = 0x1037FFBB }

3.7.7.13.2 Data Structure Documentation

3.7.7.13.2.1 struct os_het_l1_defense_t

Heterogeneous OS L1 defense control structure.

Data Fields

- uint32_t start_validation_value
- uint32_t warm_reset_mode
- uint32_t reset_maple
- uint32_t reset_status [MAX_NUM_OF_DSP_CORES]
- uint32_t end_validation_value

3.7.7.13.2.1.1 Field Documentation

3.7.7.13.2.1.2 uint32_t os_het_l1_defense_t::start_validation_value

validation value for checking for corrpution in case of reset

3.7.7.13.2.1.3 uint32_t os_het_l1_defense_t::warm_reset_mode

definition of the current L1 defense mode to be used

3.7.7.13.2.1.4 uint32_t os_het_l1_defense_t::reset_maple

only relevant if MODE_2 is defined in reset_mode - whether/which maple engine should be reset

3.7.7.13.2.1.5 uint32_t os_het_l1_defense_t::reset_status[MAX_NUM_OF_DSP_CORES]

status for each of the DSP cores

3.7.7.13.2.1.6 uint32_t os_het_l1_defense_t::end_validation_value

validation value for checking for corrpution in case of reset

3.7.7.13.3 Enumeration Type Documentation

3.7.7.13.3.1 enum os_het_l1d_mode_t

warm reset modes

Enumerator

MODE_1_ACTIVE L1 scenario mode 1.
MODE_2_ACTIVE L1 scenario mode 2.
MODE_3_ACTIVE L1 scenario mode 3.

3.7.7.13.3.2 enum os_het_l1d_reset_maple_t

mode 2 maple reset activation

Enumerator

RESET_MAPLE_NONE None of the MAPLEs were rest.
RESET_MAPLE_1 Reset 1st MAPLE-B3LW engine.
RESET_MAPLE_2 Reset 2nd MAPLE-B3LW engine.
RESET_MAPLE_3 Reset MAPLE-B3W engine.

3.7.7.13.3.3 enum os_het_l1d_status

L1 level status enumeration.

Enumerator

OS_HET_ERR_L1D_MEMORY_CORRUPTED data was corrupted
OS_HET_ERR_L1D_MODE_INVALID Warm reset mode is invalid or unsupported.
OS_HET_ERR_L1D_FUNCTION_INVALID invalid L1-defense initiation parameters
OS_HET_ERR_L1D_ALREADY_ACTIVE System is already in reset mode.
OS_HET_ERR_L1D_UNKNOWN Unknown error.
OS_HET_INFO_L1D_READY_FOR_RESET NMI received, Indication that core is ready for reset.
BEGIN_WARM_RESET_OS_INIT beginning execution of warm reset os initialization
END_OF_WARM_RESET_OS_INIT ended execution of warm reset os initialization
WARM_RESET_SUCCESS warm reset was completed successfully

3.7.7.14 B486x Heterogeneous Memory Descriptor structure API

3.7.7.14.1 Overview

Data Structures

- struct os_het_mem_t

3.7.7.14.2 Data Structure Documentation

3.7.7.14.2.1 struct os_het_mem_t

Memory Descriptor structure.

This structure will be used to describe a shared memory slab.

Data Fields

- uint64_t start_addr
- uint64_t size

3.7.7.14.2.1.1 Field Documentation

3.7.7.14.2.1.2 uint64_t os_het_mem_t::start_addr

Indicates the start address of a memory region.

3.7.7.14.2.1.3 uint64_t os_het_mem_t::size

Indicates the size of a memory region.

3.7.7.15 B486x IPC (Inter Process Communication) API

3.7.7.15.1 Overview

Data Structures

- struct os_ipc_channel_consumer_open_params_t
- struct os_ipc_channel_producer_open_params_t
- struct os_ipc_dsp_channel_t
- struct os_ipc_dsp_t

Functions

- os_het_status_t osIpcMessagePtrGet (void *ch, void **data)
- os_het_status_t osIpcMessageSendPtr (void *ch, void *data, uint32_t length, bool indication)
- void osIpcMessageReceiveCb (os_hwi_arg ch)
- void osIpcMpicMsiMessageReceiveCb (os_hwi_arg msi_id)
- os_het_status_t osIpcChannelPeek (void *ch, void **data_pointer, uint32_t *data_length)
- os_het_status_t osIpcMessageChannelBufferReplace (void *ch)
- os_het_status_t osIpcMessageChannelBufferRelease (void *ch, void *ptr)
- os_het_status_t osIpcIndicationSend (void *ch)
- os_het_status_t osIpcConsumerIndexGet (void *ch, uint32_t *index)
- os_het_status_t osIpcChannelNumGet (void *ch, uint32_t *id)
- os_het_status_t osIpcInit (void *params)
- void * osIpcMultimodeChannelIdFind (uint32_t id, uint32_t channel_region_index)
- void * osIpcChannelIdFind (uint32_t id)
- void * osIpcDspChannelIdFind (uint32_t id)
- os_het_status_t osIpcChannelConsumerOpen (os_ipc_channel_consumer_open_params_t *params)

- os_het_status_t osIpcChannelProducerOpen (os_ipc_channel_producer_open_params_t *params)
- os_het_status_t osIpcMpicMsiCoalescingSet (uint32_t(*channels)[ ], uint32_t size)
- os_het_status_t osIpcChannelInitializedVerify (void *ch)

3.7.7.15.2 Data Structure Documentation

3.7.7.15.2.1 struct os_ipc_channel_consumer_open_params_t

IPC channel consumer open parameters.

This structure should be used for opening a channel when SDOS is the consumer .

Data Fields

- void * ch
- os_het_ipc_ch_types_t channel_type
- os_het_ipc_ind_t indication_type
- os_hwi_priority int_priority
- os_hwi_arg int_argument
- uint32_t bd_ring_size
- uint32_t max_msg_size
- void(* callback)(void *ch, void *data, uint32_t length)
- os_mem_part_t * buffers_pool
- uint32_t cache_policy

3.7.7.15.2.1.1 Field Documentation

3.7.7.15.2.1.2 void* os_ipc_channel_consumer_open_params_t::ch

Pointer to the relevant channel as received from osIpcChannelIdFind or osIpcMultimodeChannelIdFind.

3.7.7.15.2.1.3 os_het_ipc_ch_types_t os_ipc_channel_consumer_open_params_t::channel_type

type of channel - message channel or pointer channel

3.7.7.15.2.1.4 os_het_ipc_ind_t os_ipc_channel_consumer_open_params_t::indication_type

the indication type to use

3.7.7.15.2.1.5 os_hwi_priority os_ipc_channel_consumer_open_params_t::int_priority

priority of interrupt, if used

3.7.7.15.2.1.6 os_hwi_arg os_ipc_channel_consumer_open_params_t::int_argument

argument of interrupt, if used

3.7.7.15.2.1.7 uint32_t os_ipc_channel_consumer_open_params_t::bd_ring_size

size of the mailbox BD ring

3.7.7.15.2.1.8 uint32_t os_ipc_channel_consumer_open_params_t::max_msg_size

maximal size of the message that can be passed on this channel

3.7.7.15.2.1.9 void(* os_ipc_channel_consumer_open_params_t::callback)(void *ch, void *data, uint32_t length)

callback function to call when receiving a message

3.7.7.15.2.1.10 os_mem_part_t* os_ipc_channel_consumer_open_params_t::buffers_pool

only for a message channel - the buffer manager to allocate.

for heterogeneous channels, buffers has to be located in DSP heterogeneous memory the buffers from

3.7.7.15.2.1.11 uint32_t os_ipc_channel_consumer_open_params_t::cache_policy

caching policy to use in the cannel.

according to os_cache.h defines

3.7.7.15.2.2 struct os_ipc_channel_producer_open_params_t

IPC channel producer open parameters.

This structure should be used for opening a channel when SDOS is the producer .

Data Fields

- void * ch
- os_het_ipc_ch_types_t channel_type
- void * additional_params
- uint32_t cache_policy

3.7.7.15.2.2.1 Field Documentation

3.7.7.15.2.2.2 void* os_ipc_channel_producer_open_params_t::ch

Pointer to the relevant channel as received from osIpcChannelIdFind or osIpcMultimodeChannelIdFind.

3.7.7.15.2.2.3 os_het_ipc_ch_types_t os_ipc_channel_producer_open_params_t::channel_type

type of channel - message channel or pointer channel

3.7.7.15.2.2.4 void* os_ipc_channel_producer_open_params_t::additional_params

Pointer to future parameters for opening channel from Starcore to Starcore, should be NULL if not used.

3.7.7.15.2.2.5 uint32_t os_ipc_channel_producer_open_params_t::cache_policy

caching policy to use in the cannel.

according to os_cache.h defines

3.7.7.15.2.3 struct os_ipc_dsp_channel_t

IPC DSP to DSP channel parameter structure.

This structure should be used when defining a local channel to be used for DSP to DSP communication only For heterogeneous communication (DSP-PA) heterogeneous channels should be used (defined in heterogeneous control structure)

Data Fields

- uint32_t id
- uint32_t bd_ring_size
- os_het_ipc_bd_t(* bd_base)[]

3.7.7.15.2.3.1 Field Documentation

3.7.7.15.2.3.2 uint32_t os_ipc_dsp_channel_t::id

Mailbox ID - may be used by the applicaiton to identify the channel; Can be an arbitrary number provided that it is unique in the ipc region.

3.7.7.15.2.3.3 uint32_t os_ipc_dsp_channel_t::bd_ring_size

Size of the mailbox BD ring; This field is set by the consumer.

May not be larger than os_het_ipc_t.ipc_max_bd_size

3.7.7.15.2.3.4 os_het_ipc_bd_t(* os_ipc_dsp_channel_t::bd_base)[]

Base of the mailbox pointer BD; as an offset from the base of the shared address space;.

3.7.7.15.2.4 struct os_ipc_dsp_t

IPC DSP to DSP structure.

This structure should be passed in initialization parameters in order to initialize IPC for DSP-DSP communication

Data Fields

- uint32_t num_ipc_channels
- uint32_t ipc_max_bd_size
- os_ipc_dsp_channel_t(* ipc_dsp_channels)[]

3.7.7.15.2.4.1 Field Documentation

3.7.7.15.2.4.2 uint32_t os_ipc_dsp_t::num_ipc_channels

Number of channels in the ipc_channels array.

3.7.7.15.2.4.3 uint32_t os_ipc_dsp_t::ipc_max_bd_size

Maximal size that the ipc_channel.bd_ring_size may have.

3.7.7.15.2.4.4 os_ipc_dsp_channel_t(* os_ipc_dsp_t::ipc_dsp_channels)[]

Pointer to the mailboxes control structure array; has os_het_ipc_t.num_ipc_channels entries as an offset from the base of the shared address space.

3.7.7.15.3 Function Documentation

3.7.7.15.3.1 os_het_status_t osIpcMessagePtrGet (void * *ch*, void ** *data*)

Get a pointer for sending a message on a "message channel"

Parameters

in	*ch*	- Pointer to the relevant channel
out	*data*	- Pointer to the data.

Return values

OS_HETERO_SUCCESS	if succeedded in sending message
OS_HETERO_FAIL	if failed in sending message

Warning

to be used on a "message channel" only. must be used before each call to osIpcMessageSendPtr

3.7.7.15.3.2 os_het_status_t osIpcMessageSendPtr (void * *ch*, void * *data*, uint32_t *length*, bool *indication*)

Copy a message to the IPC channel

Parameters

in	*ch*	- Pointer to the relevant channel
in	*data*	- Pointer to the data (in message channel, the pointer received from osIpcMessagePtrGet)
in	*length*	- optional, the length of the data
in	*indication*	- Set to TRUE to send indication or FALSE no to.

Return values

OS_HETERO_SUCCESS	if succeedded in sending message
OS_HETERO_FAIL	if failed in sending message

Warning

None

3.7.7.15.3.3 void osIpcMessageReceiveCb (os_hwi_arg *ch*)

Calls a callback function if a message exists on the channel

This function incrememnts the consumer index after returning from the callback function. if callback is NULL, no calling is done (legal only in OS_HET_NO_INT channel)

Parameters

in	*ch*	- Pointer to the relevant channel

Return values

None	

Warning

None

3.7.7.15.3.4 void osIpcMpicMsiMessageReceiveCb (os_hwi_arg *msi_id*)

MPIC MSI interrupt callback function

This function checks on which channels a message was sent and calls the osIpcMessageReceiveCb functions accordingly.

Parameters

in	*msi_id*	- MPIC MSI interrupt id

Return values

None	

Warning

None

3.7.7.15.3.5 os_het_status_t osIpcChannelPeek (void * *ch*, void ** *data_pointer*, uint32_t * *data_length*)

peeks if channel contains a message to receive. if so, returns a pointer to it

This function does not incrememnt the consumer index and does not call a callback function

Parameters

in	*ch*	- Pointer to the relevant channel
out	*data_pointer*	- Pointer to pointer to the buffer
out	*data_length*	- Pointer to the data length

Return values

OS_HETERO_SUCCESS	if received a message
OS_HETERO_FAIL	if didn't receive a message

Warning

None

3.7.7.15.3.6 os_het_status_t osIpcMessageChannelBufferReplace (void * *ch*)

relevant only for a message channel replace the current buffer received to a different buffer which will be used in the next time the current BD will be reached. This is useful when application receive a buffer and decides that it would like to keep the buffer, so it will not be overrun

Parameters

in	*ch*	- Pointer to the relevant channel

Return values

OS_HETERO_SUCCESS	if received a message
OS_HETERO_FAIL	if didn't receive a message

Warning

can be called only during the running of the recieve callback function

3.7.7.15.3.7 os_het_status_t osIpcMessageChannelBufferRelease (void * *ch*, void * *ptr*)

free a buffer that was kept by the application using osIpcMessageChannelPointerReplace

Parameters

in	*ch*	- Pointer to the relevant channel
in	*ptr*	- Pointer to the buffer to be released

Return values

OS_HETERO_SUCCESS	if received a message
OS_HETERO_FAIL	if didn't receive a message

Warning

None

3.7.7.15.3.8 os_het_status_t oslpcIndicationSend (void * *ch*)

Send an indication to the consumer

Parameters

in	*ch*	- Pointer to the relevant channel

Return values

OS_HETERO_SUCCESS	if indication was successfull
OS_HETERO_FAIL	if indication failed

Warning

None

3.7.7.15.3.9 os_het_status_t oslpcConsumerIndexGet (void * *ch*, uint32_t * *index*)

Read the current consumer index

Parameters

in	*ch*	- Pointer to the relevant channel
out	*index*	- Pointer to where the index should be written

Return values

OS_HETERO_SUCCESS	if succeeded
OS_HETERO_FAIL	if failed (illegal channel)

Warning

None

3.7.7.15.3.10 os_het_status_t osIpcChannelNumGet (void * *ch*, uint32_t * *id*)

Retrive the channel ID

Parameters

in	*ch*	- Pointer to the relevant channel
out	*id*	- Pointer to where the channel ID should be written

Return values

OS_HETERO_SUCCESS	if succeeded
OS_HETERO_FAIL	if failed (illegal channel)

Warning

None

3.7.7.15.3.11 os_het_status_t osIpcInit (void * *params*)

Initialize IPC. this function will check validiy of the IPC channels This function should be called by osInitialize

Parameters

in	*params*	- pointer for future enhancement parameters

Return values

OS_HETERO_SUCCESS	if initializtion was done successfully

OS_HETERO_FAIL	if failed to initialize

Warning

None

3.7.7.15.3.12 void* osIpcMultimodeChannelIdFind (uint32_t *id*, uint32_t *channel_region_index*)

search for a channel with specified ID in a specific IPC region. returns a pointer to the channel.

Parameters

in	*id*	- channel ID for the needed channel
in	*channel_region_index*	- the channels region index to look for the channel on

Return values

Pointer	to the relevant channel if was found successfully
NULL	if failed to find or already in use

Warning

None

3.7.7.15.3.13 void* osIpcChannelIdFind (uint32_t *id*)

search for a channel with specified ID. returns a pointer to the channel.

Parameters

in	*id*	- channel ID for the needed channel

Return values

Pointer	to the relevant channel if was found successfully
NULL	if failed to find or already in use

Warning

None

3.7.7.15.3.14 void* osIpcDspChannelIdFind (uint32_t *id*)

search for a DSP-DSP channel with specified ID. returns a pointer to the channel.

Parameters

in	*id*	- dsp channel ID for the needed channel

Return values

Pointer	to the relevant channel if was found successfully
NULL	if failed to find or already in use

Warning

None

3.7.7.15.3.15 os_het_status_t osIpcChannelConsumerOpen (os_ipc_channel_consumer_open_↩ params_t * *params*)

initialize a channel in which SDOS is a consumer.

Parameters

in	*params*	- parameters for opening a channel as consumer

Return values

OS_HETERO_SUCCESS	if initializtion was done successfully
OS_HETERO_FAIL	if failed to initialize

Warning

If SC is the producer and consumer of a channel and MESH interrupt is used then osIpcChannel↩ ProducerOpen must be called before this function.

3.7.7.15.3.16 os_het_status_t osIpcChannelProducerOpen (os_ipc_channel_producer_open_↩ params_t * *params*)

initialize a channel in which SDOS is a producer.

Parameters

in	*params*	- parameters for opening a channel as producer

Return values

OS_HETERO_SUCCESS	if initializtion was done successfully
OS_HETERO_FAIL	if failed to initialize

Warning

None

3.7.7.15.3.17 os_het_status_t osIpcMpicMsiCoalescingSet (uint32_t(∗) *channels[],* uint32_t *size*)

Set MPIC MSI coalesing

Parameters

in	*channels*	- channels array to set coalescing on
in	*size*	- number of channels in the array

Return values

OS_HETERO_SUCCESS	if initializtion was done successfully
OS_HETERO_FAIL	if failed to initialize

Warning

The consumer of all channels set for coalesing must be the same core. All the channels must use MPIC MSI interrupts

3.7.7.15.3.18 os_het_status_t osIpcChannelInitializedVerify (void ∗ *ch*)

Check if channel is initialized by both consiumer and producer In case it is not clear that a channel is ready before beginning to use it, it is possible to use this function.

Parameters

in	*ch*	- Handle of the relevant channel as received from osIpcChannelIdFind

Return values

OS_HETERO_SUCCESS	if channel is initialized by both producer and consumer
OS_HETERO_FAIL	if channel is not initialized by producer or/and consumer

Warning

None s

3.7.7.16 B4860 SmartDsp OS Devices

3.7.7.16.1 Overview

Macros

- #define OS_MEM_RESERVED(start_addr, next_addr) volatile uint8_t reserved ## start_addr [next_addr - start_addr]
- #define g_dsp_ccsr_map g_soc_ccsr_map
- #define OS_SOC_DSP_NUM_CLUSTERS 3
- #define OS_SOC_DSP_CORES_IN_CLUSTER 2
- #define OS_SOC_MAX_NUM_OF_CORES (OS_SOC_DSP_NUM_CLUSTERS * OS_SOC_D↩ SP_CORES_IN_CLUSTER)
- #define OS_SOC_DSP_LOWEST_CORENUM 4
- #define OS_SOC_PA_NUM_CLUSTERS 1
- #define OS_SOC_PA_CORES_IN_CLUSER 4
- #define OS_SOC_PA_THREADS_PER_CORE 2
- #define OS_SOC_MAX_NUM_OF_PA_CORES (OS_SOC_PA_NUM_CLUSTERS * OS_SOC↩ _PA_CORES_IN_CLUSER)
- #define OS_SOC_MAX_NUM_OF_PA_THREADS (OS_SOC_MAX_NUM_OF_PA_CORES * OS_SOC_PA_THREADS_PER_CORE)
- #define OS_SOC_PA_LOWEST_CORENUM 0
- #define OS_SOC_MAX_NUM_OF_MESSAGES 2
- #define DSP_CLUSTER_REGS_DEFAULT 0xFEC00000
- #define DSP_CCSR_REGS_DEFAULT 0x7F000000
- #define SOC_DSP_MAPLE_DEFAULT 0x7F800000
- #define SOC_DSP_MAPLE_MBUS_DEFAULT 0x79000000
- #define NUM_OF_MEMMAP_CPRI_UNITS 8
- #define MAX_NUM_OF_VIRTS 32
- #define NUMBER_OF_WDTS 2
- #define MAPLE_IN_PLATFORM TRUE
- #define NUM_OF_MAPLES 3
- #define NUM_OF_HW_TIMER_32b_GROUPS 8
- #define NUM_OF_HW_TIMERS_32b_PER_MODULE 4
- #define NUM_OF_HW_TIMERS_32b_MODULES_PER_GROUP 4
- #define NUM_OF_HW_TIMERS_32b (NUM_OF_HW_TIMER_32b_MODULES * NUM_OF_↩ HW_TIMERS_32b_PER_MODULE)
- #define NUM_OF_HW_TIMER_32b_MODULES (NUM_OF_HW_TIMERS_32b_MODULES↩ _PER_GROUP * NUM_OF_HW_TIMER_32b_GROUPS)
- #define SC39XX_NUM_OF_TIMERS 4
- #define NUMBER_OF_HW_SEMAPHORES 8
- #define NUMBER_OF_OCN_DMA 2
- #define NUMBER_OF_OCN_DMA_CH 8
- #define NUMBER_OF_OCN_DMA_ATMU_WIN 10
- #define NUMBER_OF_HSSI_PHY_PORTS (LYNX0_IN_PLATFORM + LYNX1_IN_PLATF↩ ORM)
- #define MAX_NUM_SRIO_PORTS 16
- #define NUMBER_OF_SRIO_PHY_PORTS NUMBER_OF_HSSI_PHY_PORTS
- #define NUMBER_OF_RIO_ATMU_IN_WIN 5
- #define NUMBER_OF_RIO_ATMU_OUT_WIN 9
- #define SRIO_SUPPORTS_ACCEPT_ALL TRUE
- #define MAPLE_DIRECT_ACCESSES TRUE
- #define MAX_NUM_MAPLE 3

- #define MAX_NUM_OF_MAPLE_PE 8
- #define MAX_NUM_OF_CHANNELS_PER_PE 16
- #define CLEAR_LEVEL_INTERRUPT(ADDR, DATA)

Tick Configuration

```
Values for a 1ms and 10ms clock are listed below.
```

- #define INTERRUPTS_PER_SEC_1000 1000
- #define INTERRUPTS_PER_SEC_100 100
- #define SOC_TICK_001MS INTERRUPTS_PER_SEC_1000
- #define SOC_TICK_010MS INTERRUPTS_PER_SEC_100
- #define SOC_TICK_DEFAULT SOC_TICK_010MS

Core IDs

- #define CORE0_ID 0x0000U
- #define CORE1_ID 0x0001U
- #define CORE2_ID 0x0002U
- #define CORE3_ID 0x0003U
- #define CORE4_ID 0x0004U
- #define CORE5_ID 0x0005U
- #define ANY_CORE_ID 0xFFFEU
- #define EXTERNAL_MASTER_ID 0xFFFFU

CLuster IDs

- #define CLUSTER0_ID 0x0000U
- #define CLUSTER1_ID 0x0001U
- #define CLUSTER2_ID 0x0002U
- #define ANY_CLUSTER_ID 0xFFFEU

MMU Defines

- #define NUMBER_OF_MMU_PROG_MATT 16
- #define NUMBER_OF_MMU_DATA_MATT 32
- #define SUPPORTS_MMU_FSM TRUE
- #define SUPPORTS_MMU_NEXT_LINE_PREFETCH TRUE

CME Defines

- #define NUMBER_OF_CME_DATA_CHANNELS 8
- #define NUMBER_OF_CME_PROG_CHANNELS 8

Cluster L2 Cache

- #define PRIVATE_L2_CACHE FALSE
- #define NUMBER_L2_CACHE_PARTITIONS 8

3.7.7.16.2 Macro Definition Documentation

3.7.7.16.2.1 #define OS_MEM_RESERVED(*start_addr, next_addr*) volatile uint8_t reserved ## start_addr [next_addr - start_addr]

Reserved uint8_t space generation; Useful for memory maps.

3.7.7.16.2.2 #define g_dsp_ccsr_map g_soc_ccsr_map

Backward compatibility.

3.7.7.16.2.3 #define INTERRUPTS_PER_SEC_1000 1000

tick interrupts per 1000 seconds

3.7.7.16.2.4 #define INTERRUPTS_PER_SEC_100 100

tick interrupts per 100 seconds

3.7.7.16.2.5 #define SOC_TICK_001MS INTERRUPTS_PER_SEC_1000

Tick interrupts per 1 mili seconds.

3.7.7.16.2.6 #define SOC_TICK_010MS INTERRUPTS_PER_SEC_100

Tick interrupts per 10 mili seconds.

3.7.7.16.2.7 #define SOC_TICK_DEFAULT SOC_TICK_010MS

default Tick interrupts interval

3.7.7.16.2.8 #define OS_SOC_DSP_NUM_CLUSTERS 3

Number of DSP clusters in device.

3.7.7.16.2.9 #define OS_SOC_DSP_CORES_IN_CLUSTER 2

Number of DSP cores in each cluster.

3.7.7.16.2.10 #define OS_SOC_MAX_NUM_OF_CORES (OS_SOC_DSP_NUM_CLUSTERS * OS_SOC_DSP_CORES_IN_CLUSTER)

Number of DSP cores in the SoC.

3.7.7.16.2.11 #define OS_SOC_DSP_LOWEST_CORENUM 4

Core enumeration of the lowest numbered DSP core in the SoC.

3.7.7.16.2.12 #define OS_SOC_PA_NUM_CLUSTERS 1

Number of PA clusters in device.

3.7.7.16.2.13 #define OS_SOC_PA_CORES_IN_CLUSER 4

Number of PA cores in each cluster.

3.7.7.16.2.14 #define OS_SOC_PA_THREADS_PER_CORE 2

Number of PA hyper threads per core.

3.7.7.16.2.15 #define OS_SOC_MAX_NUM_OF_PA_CORES (OS_SOC_PA_NUM_CLUSTERS * OS_SOC_PA_CORES_IN_CLUSER)

Number of PA cores in the SoC.

3.7.7.16.2.16 #define OS_SOC_MAX_NUM_OF_PA_THREADS (OS_SOC_MAX_NUM_OF_PA_↩ CORES * OS_SOC_PA_THREADS_PER_CORE)

Number of PA hyper threads in the SoC.

3.7.7.16.2.17 #define OS_SOC_PA_LOWEST_CORENUM 0

Core enumeration of the lowest numbered PA core in the SoC.

3.7.7.16.2.18 #define OS_SOC_MAX_NUM_OF_MESSAGES 2

Number of interrupt lines from each core to each other for point to point messages.

3.7.7.16.2.19 #define CORE0_ID 0x0000U

SC core 0 ID.

3.7.7.16.2.20 #define CORE1_ID 0x0001U

SC core 1 ID.

3.7.7.16.2.21 #define CORE2_ID 0x0002U

SC core 2 ID.

3.7.7.16.2.22 #define CORE3_ID 0x0003U

SC core 3 ID.

3.7.7.16.2.23 #define CORE4_ID 0x0004U

SC core 4 ID.

3.7.7.16.2.24 #define CORE5_ID 0x0005U

SC core 5 ID.

3.7.7.16.2.25 #define ANY_CORE_ID 0xFFFEU

any SC core ID

3.7.7.16.2.26 #define EXTERNAL_MASTER_ID 0xFFFFU

external master ID

3.7.7.16.2.27 #define CLUSTER0_ID 0x0000U

Cluster 0 ID.

3.7.7.16.2.28 #define CLUSTER1_ID 0x0001U

Cluster 1 ID.

3.7.7.16.2.29 #define CLUSTER2_ID 0x0002U

Cluster 2 ID.

3.7.7.16.2.30 #define ANY_CLUSTER_ID 0xFFFEU

Any Cluster ID.

3.7.7.16.2.31 #define DSP_CLUSTER_REGS_DEFAULT 0xFEC00000

Hardwired by SoC; When the processor exits reset, only MMU descriptor 0 is enabled for base virtual address 0xFEC00000, size 256 KByte, shared between all tasks.

3.7.7.16.2.32 #define DSP_CCSR_REGS_DEFAULT 0x7F000000

Set by OS.

3.7.7.16.2.33 #define SOC_DSP_MAPLE_DEFAULT 0x7F800000

MapleB3LW 0 SBUS virtual address base.

3.7.7.16.2.34 #define SOC_DSP_MAPLE_MBUS_DEFAULT 0x79000000

MapleB3 MBUS virtual base address.

3.7.7.16.2.35 #define NUM_OF_MEMMAP_CPRI_UNITS 8

Number of CPRI unit entries in memory map (not necessarily conforms to real number of CPRI units in device)

3.7.7.16.2.36 #define MAX_NUM_OF_VIRTS 32

The number of available virtual interrupts.

3.7.7.16.2.37 #define NUMBER_OF_MMU_PROG_MATT 16

number of mmu program matt

3.7.7.16.2.38 #define NUMBER_OF_MMU_DATA_MATT 32

number of mmu data matt

3.7.7.16.2.39 #define SUPPORTS_MMU_FSM TRUE

supports in MMU flexibale segmant

3.7.7.16.2.40 #define SUPPORTS_MMU_NEXT_LINE_PREFETCH TRUE

number of mmu next line prefetch

3.7.7.16.2.41 #define NUMBER_OF_CME_DATA_CHANNELS 8

number of cme data channels

3.7.7.16.2.42 #define NUMBER_OF_CME_PROG_CHANNELS 8

number of cme program channels

3.7.7.16.2.43 #define PRIVATE_L2_CACHE FALSE

L2 cache per core.

3.7.7.16.2.44 #define NUMBER_L2_CACHE_PARTITIONS 8

number of l2 partition id

3.7.7.16.2.45 #define NUMBER_OF_WDTS 2

Number of DSP Cluster WDT.

3.7.7.16.2.46 #define MAPLE_IN_PLATFORM TRUE

Boolean - Is MAPLE be present on device.
Should be set to FALSE if using an MAPLE off the SoC

3.7.7.16.2.47 #define NUM_OF_MAPLES 3

Number of maples in the system.

3.7.7.16.2.48 #define NUM_OF_HW_TIMER_32b_GROUPS 8

Number of 32b timer groups.

3.7.7.16.2.49 #define NUM_OF_HW_TIMERS_32b_PER_MODULE 4

Number of 32b timer per modules.

3.7.7.16.2.50 #define NUM_OF_HW_TIMERS_32b_MODULES_PER_GROUP 4

Number of 32b mpdules per group.

3.7.7.16.2.51 #define NUM_OF_HW_TIMERS_32b (NUM_OF_HW_TIMER_32b_MODULES ∗ NUM_OF_HW_TIMERS_32b_PER_MODULE)

Number of 32b timers.

3.7.7.16.2.52 #define NUM_OF_HW_TIMER_32b_MODULES (NUM_OF_HW_TIMERS_32b_M↩ ODULES_PER_GROUP ∗ NUM_OF_HW_TIMER_32b_GROUPS)

Number of timer 32b modules.

3.7.7.16.2.53 #define SC39XX_NUM_OF_TIMERS 4

Number of sc3900 timers.

3.7.7.16.2.54 #define NUMBER_OF_HW_SEMAPHORES 8

Hardware Semaphore defines.

Indicated that the WDT can’t reset the SoC

3.7.7.16.2.55 #define NUMBER_OF_OCN_DMA 2

Number of OCeaN DMA (for PCI-e and sRIO.

3.7.7.16.2.56 #define NUMBER_OF_OCN_DMA_CH 8

Number of channels for each OCeaN DMA.

3.7.7.16.2.57 #define NUMBER_OF_OCN_DMA_ATMU_WIN 10

Number of ATMU windows per OCeaN DMA.

3.7.7.16.2.58 #define NUMBER_OF_HSSI_PHY_PORTS (LYNX0_IN_PLATFORM + LYNX1_IN_PLATFORM)

Number of HSSI (sRIO, PCI-e, SGMII_ ports.

3.7.7.16.2.59 #define MAX_NUM_SRIO_PORTS 16

Maximum number of sRIO ports defined in the standard.

3.7.7.16.2.60 #define NUMBER_OF_SRIO_PHY_PORTS NUMBER_OF_HSSI_PHY_PORTS

Number of sRIO ports on DSP.

Number of sRIO mailboxes

3.7.7.16.2.61 #define NUMBER_OF_RIO_ATMU_IN_WIN 5

Number of sRIO inbound ATMU windows per port.

3.7.7.16.2.62 #define NUMBER_OF_RIO_ATMU_OUT_WIN 9

Number of sRIO outbound ATMU windows per port.

3.7.7.16.2.63 #define SRIO_SUPPORTS_ACCEPT_ALL TRUE

Defines whether the DSP can be configured to accept incoming packets for any sRIO ID.

3.7.7.16.2.64 #define MAPLE_DIRECT_ACCESSES TRUE

Sets whether accesses to MAPLE are performed by OS functions or application provided functions.

3.7.7.16.2.65 #define MAX_NUM_MAPLE 3

Number of MAPLE driver can handle.

3.7.7.16.2.66 #define MAX_NUM_OF_MAPLE_PE 8

Number of MAPLE PEs (supported by sdos) in the SoC.

3.7.7.16.2.67 #define MAX_NUM_OF_CHANNELS_PER_PE 16

Max number of channels in a singel PE.

3.7.7.16.2.68 #define CLEAR_LEVEL_INTERRUPT(*ADDR, DATA*)

Value:

```
{ register uint32_t ensure_read;\
    WRITE_UINT32_DBAR_SCFG(ADDR,DATA); \
    READ_UINT32(ensure_read,ADDR);}
```

CLEAR_LEVEL_INTERRUPT is the mechanism used to clear level interrutps at their source (not EPIC) where the driver doesn't know how to detect false positives on interrupt assertions.

This is not used by all drivers - ONLY where the condition above is fullfilled

CLEAR_LEVEL_INTERRUPT - Generate read after write to the interrupt status register. The write clears the interrupt and the read stalls the core until the interrupt is actually cleard. same address for both read

and write.

3.7.7.17 SC39XX Definitions

3.7.7.17.1 Overview

Macros

- #define ARCH_CACHE_LINE_SIZE 128
- #define DSP_CORE_NUM_DID 256
- #define DSP_CORE_NUM_PID 256
- #define OS_SYSTEM_DID 1
- #define OS_SYSTEM_PID 1
- #define OS_SHARED_DID 0
- #define OS_SHARED_PID 0
- #define ARCH_STACK_MAX_SIZE 0x00400000
- #define ARCH_SYS_STACK_VIRT_BASE 0x7B400000
- #define ARCH_APP_STACK_VIRT_BASE 0x7BC00000
- #define TASK_STATUS_REG 0x0000000C
- #define TASK_STATUS2_REG 0x00000004
- #define STACK_DEFAULT
- #define TASK_PC_REG_PLACE REG_OFF_LR0_0
- #define TASK_STATUS_REG_PLACE REG_OFF_LR0_1
- #define TASK_STATUS2_REG_PLACE REG_OFF_LR0_2
- #define TASK_ARG_REG_PLACE REG_OFF_R0
- #define TASK_ARG_PROCID_REG_PLACE REG_OFF_PROCID

Enumerations

- enum register_offset_in_stack

SC39XX HWI Priorities

- enum {
  SC39XX_STACK_M_DSDA_PL, SC39XX_STACK_M_DSDB_PL, SC39XX_STACK_M_DS↩
  DC_PL,
  SC39XX_CONTEXT_TID, SC39XX_CONTEXT_SIZE }
- #define SR_PRIORITIES_NUMBER 32
- #define SR_PRIORITY_SHIFT 16
- #define SR_PRIORITY_MASK ((SR_PRIORITIES_NUMBER-1) << SR_PRIORITY_SHIFT)
- #define OS_HWI_PRIORITY0 1
- #define OS_HWI_PRIORITY1 2
- #define OS_HWI_PRIORITY2 3
- #define OS_HWI_PRIORITY3 4
- #define OS_HWI_PRIORITY4 5
- #define OS_HWI_PRIORITY5 6
- #define OS_HWI_PRIORITY6 7
- #define OS_HWI_PRIORITY7 8
- #define OS_HWI_PRIORITY8 9
- #define OS_HWI_PRIORITY9 10
- #define OS_HWI_PRIORITY10 11

- #define OS_HWI_PRIORITY11 12
- #define OS_HWI_PRIORITY12 13
- #define OS_HWI_PRIORITY13 14
- #define OS_HWI_PRIORITY14 15
- #define OS_HWI_PRIORITY15 16
- #define OS_HWI_PRIORITY16 17
- #define OS_HWI_PRIORITY17 18
- #define OS_HWI_PRIORITY18 19
- #define OS_HWI_PRIORITY19 20
- #define OS_HWI_PRIORITY20 21
- #define OS_HWI_PRIORITY21 22
- #define OS_HWI_PRIORITY22 23
- #define OS_HWI_PRIORITY23 24
- #define OS_HWI_PRIORITY24 25
- #define OS_HWI_PRIORITY25 26
- #define OS_HWI_PRIORITY26 27
- #define OS_HWI_PRIORITY27 28
- #define OS_HWI_PRIORITY28 29
- #define OS_HWI_PRIORITY29 30
- #define OS_HWI_PRIORITY30 31
- #define OS_HWI_LAST_PRIORITY OS_HWI_PRIORITY30
- #define NUMBER_OF_CORE_INTERRUPTS 14
- #define ARCH_CONTEXT_SIZE 4
- #define OS_HWI_PRIORITY_ALL SR_PRIORITIES_NUMBER
- #define OS_HWI_PRIORITY_NMI (SR_PRIORITIES_NUMBER + 1)

3.7.7.17.2 Macro Definition Documentation

3.7.7.17.2.1 #define SR_PRIORITIES_NUMBER 32

number of hardware interrupts priorities

3.7.7.17.2.2 #define SR_PRIORITY_SHIFT 16

hardware interrupts priorities shift

3.7.7.17.2.3 #define SR_PRIORITY_MASK ((SR_PRIORITIES_NUMBER-1) << SR_PRIORITY_SHIFT)

hardware interrupts priorities mask

3.7.7.17.2.4 #define OS_HWI_PRIORITY0 1

Highest HWI priority.

3.7.7.17.2.5 #define OS_HWI_PRIORITY1 2

hardware interrupt priority 2

3.7.7.17.2.6 #define OS_HWI_PRIORITY2 3

hardware interrupt priority 3

3.7.7.17.2.7 #define OS_HWI_PRIORITY3 4

hardware interrupt priority 4

3.7.7.17.2.8 #define OS_HWI_PRIORITY4 5

hardware interrupt priority 5

3.7.7.17.2.9 #define OS_HWI_PRIORITY5 6

hardware interrupt priority 6

3.7.7.17.2.10 #define OS_HWI_PRIORITY6 7

hardware interrupt priority 7

3.7.7.17.2.11 #define OS_HWI_PRIORITY7 8

hardware interrupt priority 8

3.7.7.17.2.12 #define OS_HWI_PRIORITY8 9

hardware interrupt priority 9

3.7.7.17.2.13 #define OS_HWI_PRIORITY9 10

hardware interrupt priority 10

3.7.7.17.2.14 #define OS_HWI_PRIORITY10 11

hardware interrupt priority 11

3.7.7.17.2.15 #define OS_HWI_PRIORITY11 12

hardware interrupt priority 12

3.7.7.17.2.16 #define OS_HWI_PRIORITY12 13

hardware interrupt priority 13

3.7.7.17.2.17 #define OS_HWI_PRIORITY13 14

hardware interrupt priority 14

3.7.7.17.2.18 #define OS_HWI_PRIORITY14 15

hardware interrupt priority 15

3.7.7.17.2.19 #define OS_HWI_PRIORITY15 16

hardware interrupt priority 16

3.7.7.17.2.20 #define OS_HWI_PRIORITY16 17

hardware interrupt priority 17

3.7.7.17.2.21 #define OS_HWI_PRIORITY17 18

hardware interrupt priority 18

3.7.7.17.2.22 #define OS_HWI_PRIORITY18 19

hardware interrupt priority 19

3.7.7.17.2.23 #define OS_HWI_PRIORITY19 20

hardware interrupt priority 20

3.7.7.17.2.24 #define OS_HWI_PRIORITY20 21

hardware interrupt priority 21

3.7.7.17.2.25 #define OS_HWI_PRIORITY21 22

hardware interrupt priority 22

3.7.7.17.2.26 #define OS_HWI_PRIORITY22 23

hardware interrupt priority 23

3.7.7.17.2.27 #define OS_HWI_PRIORITY23 24

hardware interrupt priority 24

3.7.7.17.2.28 #define OS_HWI_PRIORITY24 25

hardware interrupt priority 25

3.7.7.17.2.29 #define OS_HWI_PRIORITY25 26

hardware interrupt priority 26

3.7.7.17.2.30 #define OS_HWI_PRIORITY26 27

hardware interrupt priority 27

3.7.7.17.2.31 #define OS_HWI_PRIORITY27 28

hardware interrupt priority 28

3.7.7.17.2.32 #define OS_HWI_PRIORITY28 29

hardware interrupt priority 29

3.7.7.17.2.33 #define OS_HWI_PRIORITY29 30

hardware interrupt priority 30

3.7.7.17.2.34 #define OS_HWI_PRIORITY30 31

hardware interrupt priority 31

3.7.7.17.2.35 #define OS_HWI_LAST_PRIORITY OS_HWI_PRIORITY30

Lowest HWI priority.

3.7.7.17.2.36 #define NUMBER_OF_CORE_INTERRUPTS 14

number of core interrupts

3.7.7.17.2.37 #define ARCH_CONTEXT_SIZE 4

Architecture dependent context size.

3.7.7.17.2.38 #define OS_HWI_PRIORITY_ALL SR_PRIORITIES_NUMBER

number of hwi priorities

3.7.7.17.2.39 #define OS_HWI_PRIORITY_NMI (SR_PRIORITIES_NUMBER + 1)

NMI interrupt priority.

3.7.7.17.2.40 #define ARCH_CACHE_LINE_SIZE 128

Cache line size.

3.7.7.17.2.41 #define DSP_CORE_NUM_DID 256

Number of DID supported by core.

3.7.7.17.2.42 #define DSP_CORE_NUM_PID 256

Number of PID supported by core.

3.7.7.17.2.43 #define OS_SYSTEM_DID 1

This is reserved and default DID for system context - MAY NOT BE CHANGED.

3.7.7.17.2.44 #define OS_SYSTEM_PID 1

This is reserved and default PID for system context - MAY NOT BE CHANGED.

3.7.7.17.2.45 #define OS_SHARED_DID 0

This is reserved and default DID for shared segments - MAY NOT BE CHANGED.

3.7.7.17.2.46 #define OS_SHARED_PID 0

This is reserved and default DID for shared segments - MAY NOT BE CHANGED.

3.7.7.17.2.47 #define ARCH_STACK_MAX_SIZE 0x00400000

All application stacks will have the same virtual address.

3.7.7.17.2.48 #define ARCH_SYS_STACK_VIRT_BASE 0x7B400000

In order to provide stack protection, the maximum size needs to be known (set to 4MB)

3.7.7.17.2.49 #define ARCH_APP_STACK_VIRT_BASE 0x7BC00000

System stack virtual address.

3.7.7.17.2.50 #define TASK_STATUS_REG 0x0000000C

Task SR create value: Dual 16-bit, no saturation No scaling Twos-complement rounding 32-bit Arithmetic Saturation enabled.

3.7.7.17.2.51 #define TASK_STATUS2_REG 0x00000004

Task SR2 create value: Interrupts enabled Interrupt level 0 Exception Mode ESP is the alternate stack pointer TSP is the stack pointer.

3.7.7.17.2.52 #define STACK_DEFAULT

Default stack content.

3.7.7.17.2.53 #define TASK_PC_REG_PLACE REG_OFF_LR0_0

Task PC register place.

3.7.7.17.2.54 #define TASK_STATUS_REG_PLACE REG_OFF_LR0_1

Task status register place.

3.7.7.17.2.55 #define TASK_STATUS2_REG_PLACE REG_OFF_LR0_2

Task status register place.

3.7.7.17.2.56 #define TASK_ARG_REG_PLACE REG_OFF_R0

Task argument register place.

3.7.7.17.2.57 #define TASK_ARG_PROCID_REG_PLACE REG_OFF_PROCID

Task PROCID register place.

3.7.7.17.3 Enumeration Type Documentation

3.7.7.17.3.1 anonymous enum

Enumerator

SC39XX_STACK_M_DSDA_PL Task's stack MMU Register A configuration.
SC39XX_STACK_M_DSDB_PL Task's stack MMU Register B configuration.
SC39XX_STACK_M_DSDC_PL Task's stack MMU Register C configuration.
SC39XX_CONTEXT_TID Task's OS allocated TID.
SC39XX_CONTEXT_SIZE number of bytes in the control structure

3.7.7.17.3.2 enum register_offset_in_stack

Registers offsets in the stack; the names reflect the real register names.

3.7.7.17.4 Tasks Runtime API.

3.7.7.17.4.1 Overview

Data Structures

- struct os_task_init_param_t

Macros

- #define TASK_NAME_LEN 12
- #define TASK_IDLE_ID 0
- #define TASK_NULL_ID 0xFFFFFFFF
- #define OS_NUM_OF_PRIORITIES 32

Functions

- os_status osTaskFind (os_task_handle *task_handle)
- os_status osTaskCreate (os_task_handle task_handle, os_task_init_param_t *task_init_param)
- os_status osTaskActivate (os_task_handle task_handle)
- os_status osTaskSuspend (os_task_handle task_handle)
- os_status osTaskDelete (os_task_handle task_handle)
- os_status osTaskDelay (uint32_t timeout)
- os_status osTaskPrioritySet (os_task_handle task_handle, os_task_priority task_priority)
- void osTaskSchedulerLock ()
- void osTaskSchedulerUnlock ()
- os_status osTaskSelf (os_task_handle *task_handle)
- uint32_t osTaskStatusGet (os_task_handle task_handle)
- uint32_t osTaskPrivateDataGet (os_task_handle task_handle)
- void osTaskPrivateDataSet (os_task_handle task_handle, uint32_t private_data)
- os_task_priority osTaskPriorityGet (os_task_handle task_handle)
- uint32_t osTaskPriorityReadyCount (uint32_t priority)
- os_status osTaskYield (os_task_priority task_priority, os_task_handle next_task, os_task_handle *yielded_to)
- char * osTaskNameGet (os_task_handle task_handle)

Available task priorities.

- #define OS_TASK_PRIORITY_HIGHEST 0
- #define OS_TASK_PRIORITY_01 1
- #define OS_TASK_PRIORITY_02 2
- #define OS_TASK_PRIORITY_03 3
- #define OS_TASK_PRIORITY_04 4
- #define OS_TASK_PRIORITY_05 5
- #define OS_TASK_PRIORITY_06 6
- #define OS_TASK_PRIORITY_07 7
- #define OS_TASK_PRIORITY_08 8
- #define OS_TASK_PRIORITY_09 9
- #define OS_TASK_PRIORITY_10 10
- #define OS_TASK_PRIORITY_11 11
- #define OS_TASK_PRIORITY_12 12
- #define OS_TASK_PRIORITY_13 13
- #define OS_TASK_PRIORITY_14 14
- #define OS_TASK_PRIORITY_15 15
- #define OS_TASK_PRIORITY_16 16
- #define OS_TASK_PRIORITY_17 17
- #define OS_TASK_PRIORITY_18 18
- #define OS_TASK_PRIORITY_19 19
- #define OS_TASK_PRIORITY_20 20
- #define OS_TASK_PRIORITY_21 21
- #define OS_TASK_PRIORITY_22 22
- #define OS_TASK_PRIORITY_23 23
- #define OS_TASK_PRIORITY_24 24
- #define OS_TASK_PRIORITY_25 25
- #define OS_TASK_PRIORITY_26 26
- #define OS_TASK_PRIORITY_27 27
- #define OS_TASK_PRIORITY_28 28
- #define OS_TASK_PRIORITY_29 29
- #define OS_TASK_PRIORITY_30 30
- #define OS_TASK_PRIORITY_LOWEST 31

Task status codes.

- #define OS_TASK_UNUSED 0x0000001
- #define OS_TASK_ACQUIRED 0x0000002
- #define OS_TASK_SUSPEND 0x0000004
- #define OS_TASK_PEND 0x0000008
- #define OS_TASK_TIMEOUT 0x0000010
- #define OS_TASK_DELAY 0x0000020
- #define OS_TASK_READY 0x0000040
- #define OS_TASK_RUNNING 0x0000080
- #define OS_TASK_EXPIRED 0x0000100

3.7.7.17.4.2 Data Structure Documentation

3.7.7.17.4.2.1 struct os_task_init_param_t

Config parameters for a task.

Data Fields

- os_task_function task_function

- os_task_arg task_arg
- uint32_t top_of_stack
- uint32_t stack_size
- uint32_t task_flags
- os_task_priority task_priority
- char * task_name
- uint32_t private_data
- bool init_stack

3.7.7.17.4.2.2 Field Documentation

3.7.7.17.4.2.3 os_task_function os_task_init_param_t::task_function

Task function - This function will run when the task will be activated.

This function should never exit

3.7.7.17.4.2.4 os_task_arg os_task_init_param_t::task_arg

Task argument - Will be the parameter of task_function.

3.7.7.17.4.2.5 uint32_t os_task_init_param_t::top_of_stack

Top of task's stack.

3.7.7.17.4.2.6 uint32_t os_task_init_param_t::stack_size

The size of the above space.

3.7.7.17.4.2.7 uint32_t os_task_init_param_t::task_flags

Currently reserved.

3.7.7.17.4.2.8 os_task_priority os_task_init_param_t::task_priority

Task priority - From OS_TASK_PRIORITY_HIGHEST to OS_TASK_PRIORITY_LOWEST.

3.7.7.17.4.2.9 char* os_task_init_param_t::task_name

Task name - Should be not longer then TASK_NAME_LEN.

3.7.7.17.4.2.10 uint32_t os_task_init_param_t::private_data

This field can be read and written from any point during the life span of the task.

3.7.7.17.4.2.11 bool os_task_init_param_t::init_stack

Whether to initialize the stack.

This initialization is used by kernel awareness to see usage of the stack.

3.7.7.17.4.3 Macro Definition Documentation

3.7.7.17.4.3.1 #define TASK_NAME_LEN 12

Maximal task name.

3.7.7.17.4.3.2 #define TASK_IDLE_ID 0

ID of background task.

3.7.7.17.4.3.3 #define TASK_NULL_ID 0xFFFFFFFF

This ID never belongs to a valid task.

3.7.7.17.4.3.4 #define OS_TASK_PRIORITY_HIGHEST 0

task priority 0

3.7.7.17.4.3.5 #define OS_TASK_PRIORITY_01 1

task priority 1

3.7.7.17.4.3.6 #define OS_TASK_PRIORITY_02 2

task priority 2

3.7.7.17.4.3.7 #define OS_TASK_PRIORITY_03 3

task priority 3

3.7.7.17.4.3.8 #define OS_TASK_PRIORITY_04 4

task priority 4

3.7.7.17.4.3.9 #define OS_TASK_PRIORITY_05 5

task priority 5

3.7.7.17.4.3.10 #define OS_TASK_PRIORITY_06 6

task priority 6

3.7.7.17.4.3.11 #define OS_TASK_PRIORITY_07 7

task priority 7

3.7.7.17.4.3.12 #define OS_TASK_PRIORITY_08 8

task priority 8

3.7.7.17.4.3.13 #define OS_TASK_PRIORITY_09 9

task priority 9

3.7.7.17.4.3.14 #define OS_TASK_PRIORITY_10 10

task priority 10

3.7.7.17.4.3.15 #define OS_TASK_PRIORITY_11 11

task priority 11

3.7.7.17.4.3.16 #define OS_TASK_PRIORITY_12 12

task priority 12

3.7.7.17.4.3.17 #define OS_TASK_PRIORITY_13 13

task priority 13

3.7.7.17.4.3.18 #define OS_TASK_PRIORITY_14 14

task priority 14

3.7.7.17.4.3.19 #define OS_TASK_PRIORITY_15 15

task priority 15

3.7.7.17.4.3.20 #define OS_TASK_PRIORITY_16 16

task priority 16

3.7.7.17.4.3.21 #define OS_TASK_PRIORITY_17 17

task priority 17

3.7.7.17.4.3.22 #define OS_TASK_PRIORITY_18 18

task priority 18

3.7.7.17.4.3.23 #define OS_TASK_PRIORITY_19 19

task priority 19

3.7.7.17.4.3.24 #define OS_TASK_PRIORITY_20 20

task priority 20

3.7.7.17.4.3.25 #define OS_TASK_PRIORITY_21 21

task priority 21

3.7.7.17.4.3.26 #define OS_TASK_PRIORITY_22 22

task priority 22

3.7.7.17.4.3.27 #define OS_TASK_PRIORITY_23 23

task priority 23

3.7.7.17.4.3.28 #define OS_TASK_PRIORITY_24 24

task priority 24

3.7.7.17.4.3.29 #define OS_TASK_PRIORITY_25 25

task priority 25

3.7.7.17.4.3.30 #define OS_TASK_PRIORITY_26 26

task priority 26

3.7.7.17.4.3.31 #define OS_TASK_PRIORITY_27 27

task priority 27

3.7.7.17.4.3.32 #define OS_TASK_PRIORITY_28 28

task priority 28

3.7.7.17.4.3.33 #define OS_TASK_PRIORITY_29 29

task priority 29

3.7.7.17.4.3.34 #define OS_TASK_PRIORITY_30 30

task priority 30

3.7.7.17.4.3.35 #define OS_TASK_PRIORITY_LOWEST 31

task priority 31

3.7.7.17.4.3.36 #define OS_NUM_OF_PRIORITIES 32

number of task priorities

Number of Available task priorities

3.7.7.17.4.3.37 #define OS_TASK_UNUSED 0x0000001

Task is not allocated.

3.7.7.17.4.3.38 #define OS_TASK_ACQUIRED 0x0000002

Task was allocated but not created.

3.7.7.17.4.3.39 #define OS_TASK_SUSPEND 0x0000004

Task was created but not activated yet.

It may also been activated and then suspended.

3.7.7.17.4.3.40 #define OS_TASK_PEND 0x0000008

Task is waiting for an event.

3.7.7.17.4.3.41 #define OS_TASK_TIMEOUT 0x0000010

Task is waiting for an event with a timeout.

At the beginning OS_TASK_PEND is also ON, but after timeout expires, it may stay alone.

3.7.7.17.4.3.42 #define OS_TASK_DELAY 0x0000020

Task is waiting for a timeout to expire.

3.7.7.17.4.3.43 #define OS_TASK_READY 0x0000040

A task is ready to run.

This state may be exist with or without OS_TASK_RUNNING.

3.7.7.17.4.3.44 #define OS_TASK_RUNNING 0x0000080

This task is the highest priority ready task and it is the first in its priority, thus it is the currently running task.

3.7.7.17.4.3.45 #define OS_TASK_EXPIRED 0x0000100

The timeout on which this task was waiting has expired.

3.7.7.17.4.4 Function Documentation

3.7.7.17.4.4.1 os_status osTaskFind (os_task_handle * *task_handle*)

Get the index of an unused task.

Parameters

out	*task_handle*	- Gets the task handle.

Return values

OS_SUCCESS	- Found
OS_ERR_TSK_UNAVAILABLE	- Not found

3.7.7.17.4.4.2 os_status osTaskCreate (os_task_handle *task_handle*, os_task_init_param_t * *task_init_param*)

Create a task.

This function initializes the stack of a task, including the function to run. The task stack may also be filled with special value to monitor its usage. Note, that task is created in suspended state. User should call osTaskActivate().

Parameters

in	*task_init_param*	- Task configuration parameters.
in	*task_handle*	- Task to be created.

Return values

OS_SUCCESS	- Created
OS_ERR_TSK_ALREADY_CREATED	- Not created

3.7.7.17.4.4.3 os_status osTaskActivate (os_task_handle *task_handle*)

This function removes task from suspend state.

If the task is not suspended, the function has no effect. If the task is not delayed or pended, it is moved to the ready state, and scheduler is called. If this is the highest priority ready task and it is the first in its priority queue, then it will resume or start.

Parameters

in	*task_handle*	- Handle of the task to be activated.

Return values

OS_SUCCESS	- Activated
OS_ERR_TSK_OS_ER↩ R_ALREADY_ACTIVE	- Not activated

3.7.7.17.4.4.4 os_status osTaskSuspend (os_task_handle *task_handle*)

Move task to suspend state.

```
When a task is created it is placed in suspend state. Then it can
be removed from that state by osTaskActivate() and returned to the
suspend state by this function.
```

Parameters

in	*task_handle*	- Handle of task to suspend.

Return values

OS_SUCCESS	- Suspended
OS_ERR_TSK_ALREA↩ DY_SUSPENDED	- Not suspended

Warning

Do not call this function for background

3.7.7.17.4.4.5 os_status osTaskDelete (os_task_handle *task_handle*)

Deletes a task.

```
If a task is created, then it is removed from any event its pending
on and its TCB is freed. If task deletes itself, then scheduler is called.
```

Parameters

in	*task_handle*	- Handle of task to delete.

Return values

OS_SUCCESS	- Deleted
OS_ERR_TSK_NOT_C↩ REATED	- Not deleted
OS_ERR_TSK_DELET↩ E_LOCKED	- Can not delete a running task when sceduler is locked. Will cause crash.

Warning

Do not call this function for background

3.7.7.17.4.4.6 os_status osTaskDelay (uint32_t *timeout*)

Delay a running task.

```
Blocks a task for the specified number of ticks. Background task
can never be delayed and must be always ready to run.
This functionality relies on Tick operation.
```

Parameters

in	*timeout*	- number ticks to delay.

Return values

OS_SUCCESS	- Delayed
OS_ERR_TSK_DELAY_IN_INTERR	- Can not delay a task from interrupt context.
OS_ERR_TSK_DELAY_IN_LOCK	- Can not delay a running task when sceduler is locked. This will prevent multiple delays on the same task.

Warning

Do not call this function for background

3.7.7.17.4.4.7 os_status osTaskPrioritySet (os_task_handle *task_handle*, os_task_priority *task_priority*)

Set priority of a task.

```
Task priority is dynamic. Users can change its priority
at any point and this change is immediate.
```

Parameters

in	*task_handle*	- Handle of task to set priority of.
in	*task_priority*	- Task's new priority.

Return values

OS_SUCCESS	- Priority was set
OS_ERR_TSK_NOT_C↩ REATED	- Task was uncreated

3.7.7.17.4.4.8 void osTaskSchedulerLock ( )

Lock scheduler so running task will not be replaced.

```
This is useful for critical sections where the task wants to
protect itself only from other tasks and not from HWI or SWI,
So it does not prolong interrupt latency.
```

Warning

Call this function only from task

3.7.7.17.4.4.9 void osTaskSchedulerUnlock ( )

Unlock scheduler so running task can be replaced. This is useful for critical sections where the task wants to protect itself only from other tasks and not from HWI or SWI, So it does not prolong interrupt latency.

Warning

Call this function only from task

3.7.7.17.4.4.10 os_status osTaskSelf (os_task_handle * *task_handle*)

Gets the handle of the running task.

Parameters

`out`	*task_handle*	- reference to a task handle.

Return values

OS_SUCCESS	- Handle is valid
OS_ERR_TSK_NOT_A↩ CTIVE	- In SWI or HWI.

Warning

Call this function only from task

3.7.7.17.4.4.11 uint32_t osTaskStatusGet (os_task_handle *task_handle*)

Gets the status of a task. Returns the status of a task. A status can be any valid combination of status bits as explained in chapter 4 of the documentation.

Parameters

in	*task_handle*	- Gets the task handle.

Returns

Task's Status, i.e. a legal combination of:

- OS_TASK_UNUSED
- OS_TASK_ACQUIRED
- OS_TASK_SUSPEND
- OS_TASK_PEND
- OS_TASK_TIMEOUT
- OS_TASK_DELAY
- OS_TASK_READY
- OS_TASK_RUNNING

3.7.7.17.4.4.12 uint32_t osTaskPrivateDataGet (os_task_handle *task_handle*)

Gets private data of a task. Private data is a 4 bytes field associated with task that are maintained through the life span of a task. This field can be retrieved by any entity in the system.

Parameters

in	*task_handle*	- Handle of task to get private data of.

Returns

Private data of a task.

3.7.7.17.4.4.13 void osTaskPrivateDataSet (os_task_handle *task_handle,* uint32_t *private_data*)

Sets private data of a task. Private data is a 4 bytes field associated with task that are maintained through the life span of a task. This field Can be set by any entity in the system to pass data into a task.

Parameters

in	*task_handle*	- Handle of task to set private data of.
in	*private_data*	- Private data.

3.7.7.17.4.4.14 os_task_priority osTaskPriorityGet (os_task_handle *task_handle*)

Gets task's priority.

Parameters

in	*task_handle*	- Handle of task to get the priority of.

Returns

Task's priority.

3.7.7.17.4.4.15 uint32_t osTaskPriorityReadyCount (uint32_t *priority*)

Counts ready tasks in specified priority.

```
This function is useful to see if user may call osTaskYield for this priority.
```

Parameters

in	*priority*	- Priority to count.

Returns

Number of tasks in the priority.

3.7.7.17.4.4.16 os_status osTaskYield (os_task_priority *task_priority*, os_task_handle *next_task*, os_task_handle * *yielded_to*)

Schedule a different task in same priority.

```
If there are multiple tasks in the same priority then this function
can be used. Either:to perform cooperative scheduling, when a task may
step a side and let another task in it's own priority take the
CPU, or, other thread of execution may determine what is
the next task that will run at this priority.
For instance by calling this function in tick software interrupt handler.
The function can let the next task in the same priority run, or let the
task specified by next_task run, if its in task_priority.
The task that was resumed is returned in yielded_to.
```

Parameters

in	*task_priority*	- priority to yield in.
in	*next_task*	- Move to if in same priority.
out	*yielded_to*	- The task that was promoted to be first in its priority.

Return values

OS_SUCCESS	
OS_ERR_TSK_YIELD_↩ TO_ILLEGAL_TASK	- If next_task is not in task_priority priority.
OS_ERR_TSK_YIELD_↩ TO_SINGLE_TASK	- there is only one task on this priority.

Warning

Do not call this before osStart

3.7.7.17.4.4.17 char* osTaskNameGet (os_task_handle *task_handle*)

Gets name of a task. The name is what was registered in the task_name field of os_task_init_param_t when osTaskCreate() was called.

Parameters

in	*task_handle*	- task

Returns

The name of the task.

Warning

Do not call this function in HWI/SWI

3.7.8 Hardware Timers Initialization API

3.7.8.1 Overview

Hardware timers in the SoC Initialization API.

Data Structures

- struct hw_timers32_data_t
- struct timer32_trigger_params_t
- struct timer32_group_init_params_t

Enumerations

- enum timer32_input_select_t

Functions

- os_status hwTimer32Initialize ()
- os_status hwTimer32GroupInitialize (timer32_group_init_params_t *params)

3.7.8.2 Data Structure Documentation

3.7.8.2.1 struct hw_timers32_data_t

structure of 32 bit timers data.

Data Fields

- uint8_t timer32_core_id [NUM_OF_HW_TIMERS_32b]

3.7.8.2.1.1 Field Documentation

3.7.8.2.1.1.1 uint8_t hw_timers32_data_t::timer32_core_id[NUM_OF_HW_TIMERS_32b]

owning core for each of the 32-bit timers

3.7.8.2.2 struct timer32_trigger_params_t

Timer32 counting trigger initialization structure.

Data Fields

- hw_timer_trigger_input_t trigger_type
- timer32_input_select_t trigger_input

3.7.8.2.2.1 Field Documentation

3.7.8.2.2.1.1 hw_timer_trigger_input_t timer32_trigger_params_t::trigger_type

counting trigger type

3.7.8.2.2.1.2 timer32_input_select_t timer32_trigger_params_t::trigger_input

trigger mux input selection

3.7.8.2.3 struct timer32_group_init_params_t

parameters for timer32 module initialization structure

Data Fields

- timer32_trigger_params_t trigger_params [TIMER32_NUM_OF_TRIGGERS_PER_GROUP]
- timer_clock_input_t clock_in
- soc_timer32_group_t group_num

3.7.8.2.3.1 Field Documentation

3.7.8.2.3.1.1 timer32_trigger_params_t timer32_group_init_params_t::trigger_params[TIMER32_NUM_OF_TRIGGERS_PER_GROUP]

counting trigger initialization parameters

3.7.8.2.3.1.2 timer_clock_input_t timer32_group_init_params_t::clock_in

clock input selection (architecture dependent)

3.7.8.2.3.1.3 soc_timer32_group_t timer32_group_init_params_t::group_num

HW timers module to initialize.

architecture dependent

3.7.8.3 Enumeration Type Documentation

3.7.8.3.1 enum timer32_input_select_t

SoC HW timers32 MUX input selection.

3.7.8.4 Function Documentation

3.7.8.4.1 os_status hwTimer32Initialize ( )

Initializes the hardware timers32 module.

Returns

OS_SUCCESS

3.7.8.4.2 os_status hwTimer32GroupInitialize (timer32_group_init_params_t * *params*)

Initialize a group of timer modules (4 modules per group).

Parameters

in	*params*	- The module initialization params

Return values

OS_SUCCESS	- The hardware timer was successfully created.
OS_ERR_HW_TM_INV↩ ALID	- Invalid hardware timer number.
OS_ERR_HW_TM_MO↩ DE_INVALID	- Invalid or unsupported hardware timer mode.
OS_ERR_HW_TM_INT↩ ERVAL_UNSUPPORT↩ ED	- Unsupported interval.
OS_ERR_HW_TM_SO↩ URCE_INVALID	- Invalid source clock.
OS_ERR_HW_TM_PRI↩ ORITY_INVALID	- Invalid HWI priority.
OS_ERR_HW_TM_ALR↩ EADY_CREATED	- The hardware timer is already created.

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support