

Linux QDMA Driver

2000-0142

Generated by Doxygen 1.8.11

Contents

1	Data Structure Index	1
1.1	Data Structures	1
2	File Index	3
2.1	File List	3
3	Data Structure Documentation	5
3.1	descriptor Union Reference	5
3.2	descq_csr_info Struct Reference	5
3.2.1	Detailed Description	5
3.3	drv_mode_name Struct Reference	5
3.4	free_entry Struct Reference	6
3.4.1	Field Documentation	6
3.4.1.1	list_head	6
3.4.1.2	next_qbase	6
3.4.1.3	free	6
3.4.1.4	index	6
3.5	global_csr_conf Struct Reference	6
3.5.1	Detailed Description	7
3.5.2	Field Documentation	7
3.5.2.1	ring_sz	7
3.5.2.2	c2h_timer_cnt	7
3.5.2.3	c2h_cnt_th	7
3.5.2.4	c2h_buf_sz	7

3.5.2.5	<code>cmpl_status_acc</code>	7
3.6	<code>hw_descq_context</code> Struct Reference	7
3.6.1	Detailed Description	8
3.6.2	Field Documentation	8
3.6.2.1	<code>sw</code>	8
3.6.2.2	<code>prefetch</code>	8
3.6.2.3	<code>cmpt</code>	8
3.6.2.4	<code>hw</code>	8
3.6.2.5	<code>cr</code>	8
3.6.2.6	<code>fmap</code>	8
3.7	<code>intr_coal_conf</code> Struct Reference	8
3.7.1	Detailed Description	9
3.7.2	Field Documentation	9
3.7.2.1	<code>vec_id</code>	9
3.7.2.2	<code>intr_rng_num_entries</code>	9
3.7.2.3	<code>intr_ring_base</code>	9
3.7.2.4	<code>color</code>	9
3.8	<code>intr_info_t</code> Struct Reference	9
3.8.1	Field Documentation	10
3.8.1.1	<code>msix_name</code>	10
3.8.1.2	<code>intr_list</code>	10
3.8.1.3	<code>intr_list_cnt</code>	10
3.9	<code>intr_vec_map_type</code> Struct Reference	10
3.9.1	Detailed Description	10
3.9.2	Field Documentation	10
3.9.2.1	<code>intr_type</code>	10
3.9.2.2	<code>intr_vec_index</code>	10
3.9.2.3	<code>intr_handler</code>	11
3.10	<code>mbox_msg</code> Struct Reference	11
3.10.1	Detailed Description	11

3.10.2	Field Documentation	11
3.10.2.1	list	11
3.10.2.2	waitq	11
3.10.2.3	hdr	12
3.10.2.4	fmap	12
3.10.2.5	intr_ctxt	12
3.10.2.6	qctxt	12
3.10.2.7	csr	12
3.10.2.8	raw	12
3.11	mbox_msg_csr Struct Reference	12
3.11.1	Detailed Description	12
3.11.2	Field Documentation	13
3.11.2.1	hdr	13
3.12	mbox_msg_fmap Struct Reference	13
3.12.1	Detailed Description	13
3.12.2	Field Documentation	13
3.12.2.1	hdr	13
3.12.2.2	qbase	13
3.12.2.3	qmax	13
3.13	mbox_msg_hdr Struct Reference	14
3.13.1	Detailed Description	14
3.13.2	Field Documentation	14
3.13.2.1	src	14
3.13.2.2	dst	14
3.13.2.3	status	14
3.14	mbox_msg_intr_ctxt Struct Reference	14
3.14.1	Detailed Description	15
3.14.2	Field Documentation	15
3.14.2.1	hdr	15
3.14.2.2	clear	15

3.14.2.3	filler	15
3.14.2.4	vec_base	15
3.14.2.5	num_rings	15
3.14.2.6	ring_index_list	15
3.14.2.7	w	15
3.15	mbox_msg_qctx Struct Reference	15
3.15.1	Detailed Description	16
3.15.2	Field Documentation	16
3.15.2.1	hdr	16
3.15.2.2	clear	16
3.15.2.3	verify	16
3.15.2.4	c2h	16
3.15.2.5	st	16
3.15.2.6	intr_en	16
3.15.2.7	intr_id	17
3.15.2.8	qid	17
3.15.2.9	context	17
3.16	qconf_entry Struct Reference	17
3.16.1	Detailed Description	17
3.16.2	Field Documentation	17
3.16.2.1	list_head	17
3.16.2.2	idx	17
3.16.2.3	qbase	18
3.16.2.4	qmax	18
3.16.2.5	cfg_state	18
3.16.2.6	type	18
3.16.2.7	func_id	18
3.17	qconf_entry_head Struct Reference	18
3.17.1	Detailed Description	18
3.17.2	Field Documentation	19

3.17.2.1	vf_list	19
3.17.2.2	vf_free_list	19
3.17.2.3	pf_list	19
3.17.2.4	vf_qmax	19
3.17.2.5	pf_qmax	19
3.17.2.6	vf_qbase	19
3.17.2.7	vf_cnt	19
3.17.2.8	qcnt_cfgd_free	19
3.17.2.9	qcnt_init_free	19
3.17.2.10	qcnt_init_used	20
3.18	qdma_c2h_cmpt_cmpl_status Struct Reference	20
3.18.1	Detailed Description	20
3.18.2	Field Documentation	20
3.18.2.1	pidx	20
3.18.2.2	cidx	20
3.18.2.3	color_isr_status	20
3.19	qdma_c2h_desc Struct Reference	20
3.19.1	Detailed Description	21
3.19.2	Field Documentation	21
3.19.2.1	dst_addr	21
3.20	qdma_cdev Struct Reference	21
3.20.1	Detailed Description	21
3.20.2	Field Documentation	22
3.20.2.1	list_head	22
3.20.2.2	minor	22
3.20.2.3	cdevno	22
3.20.2.4	xcb	22
3.20.2.5	sys_device	22
3.20.2.6	cdev	22
3.20.2.7	c2h_qhndl	22

3.20.2.8	h2c_qhndl	22
3.20.2.9	dir_init	22
3.20.2.10	fp_open_extra	22
3.20.2.11	fp_close_extra	23
3.20.2.12	fp_ioctl_extra	23
3.20.2.13	fp_rw	23
3.20.2.14	name	23
3.21	qdma_cdev_cb Struct Reference	23
3.21.1	Detailed Description	23
3.21.2	Field Documentation	23
3.21.2.1	xpdev	23
3.21.2.2	lock	24
3.21.2.3	cdev_major	24
3.21.2.4	cdev_minor_cnt	24
3.22	qdma_cmpl_ctrl Struct Reference	24
3.22.1	Detailed Description	24
3.22.2	Field Documentation	24
3.22.2.1	cnt_th_idx	24
3.22.2.2	timer_idx	24
3.22.2.3	trigger_mode	25
3.22.2.4	en_stat_desc	25
3.22.2.5	cmpl_en_intr	25
3.23	qdma_csr_info Struct Reference	25
3.23.1	Field Documentation	25
3.23.1.1	array	25
3.23.1.2	idx_bufsz	25
3.24	qdma_desc_cmpl_status Struct Reference	26
3.24.1	Detailed Description	26
3.24.2	Field Documentation	26
3.24.2.1	pidx	26

3.24.2.2	<code>cidx</code>	26
3.24.2.3	<code>rsvd</code>	26
3.25	<code>qdma_descq</code> Struct Reference	26
3.25.1	Detailed Description	27
3.25.2	Field Documentation	28
3.25.2.1	<code>conf</code>	28
3.25.2.2	<code>lock</code>	28
3.25.2.3	<code>xdev</code>	28
3.25.2.4	<code>channel</code>	28
3.25.2.5	<code>err</code>	28
3.25.2.6	<code>color</code>	28
3.25.2.7	<code>cpu_assigned</code>	28
3.25.2.8	<code>proc_req_running</code>	28
3.25.2.9	<code>q_state</code>	28
3.25.2.10	<code>qidx_hw</code>	28
3.25.2.11	<code>intr_work_cpu</code>	29
3.25.2.12	<code>work</code>	29
3.25.2.13	<code>intr_list</code>	29
3.25.2.14	<code>legacy_intr_q_list</code>	29
3.25.2.15	<code>intr_id</code>	29
3.25.2.16	<code>work_list</code>	29
3.25.2.17	<code>cmplthp</code>	29
3.25.2.18	<code>cmplthp_list</code>	29
3.25.2.19	<code>pend_list</code>	29
3.25.2.20	<code>pend_list_wq</code>	29
3.25.2.21	<code>pend_list_empty</code>	30
3.25.2.22	<code>avail</code>	30
3.25.2.23	<code>io_batch_cnt</code>	30
3.25.2.24	<code>pend_req_desc</code>	30
3.25.2.25	<code>pidx</code>	30

3.25.2.26 cidx	30
3.25.2.27 credit	30
3.25.2.28 desc	30
3.25.2.29 desc_bus	30
3.25.2.30 desc_cmpl_status	30
3.25.2.31 fl_pg_order	31
3.25.2.32 cmpt_entry_len	31
3.25.2.33 rsvd	31
3.25.2.34 flq	31
3.25.2.35 udd_cnt	31
3.25.2.36 pkt_cnt	31
3.25.2.37 pkt_dlen	31
3.25.2.38 pidx_cmpt	31
3.25.2.39 cidx_cmpt	31
3.25.2.40 cidx_cmpt_pend	31
3.25.2.41 total_cmpl_descs	32
3.25.2.42 desc_cmpt_cur	32
3.25.2.43 desc_cmpt	32
3.25.2.44 desc_cmpt_bus	32
3.25.2.45 desc_cmpt_cmpl_status	32
3.26 qdma_dev Struct Reference	32
3.26.1 Detailed Description	32
3.26.2 Field Documentation	33
3.26.2.1 init_qrange	33
3.26.2.2 filler	33
3.26.2.3 qmax	33
3.26.2.4 qbase	33
3.26.2.5 lock	33
3.26.2.6 h2c_qcnt	33
3.26.2.7 c2h_qcnt	33

3.26.2.8	<code>h2c_descq</code>	33
3.26.2.9	<code>c2h_descq</code>	33
3.27	<code>qdma_dev_conf</code> Struct Reference	34
3.27.1	Detailed Description	34
3.27.2	Field Documentation	34
3.27.2.1	<code>pdev</code>	34
3.27.2.2	<code>qsets_max</code>	34
3.27.2.3	<code>rsvd2</code>	35
3.27.2.4	<code>zerolen_dma</code>	35
3.27.2.5	<code>master_pf</code>	35
3.27.2.6	<code>isr_top_q_en</code>	35
3.27.2.7	<code>rsvd1</code>	35
3.27.2.8	<code>vf_max</code>	35
3.27.2.9	<code>intr_rngsz</code>	35
3.27.2.10	<code>msix_qvec_max</code>	35
3.27.2.11	<code>uld</code>	36
3.27.2.12	<code>qdma_drv_mode</code>	36
3.27.2.13	<code>name</code>	36
3.27.2.14	<code>bar_num_config</code>	36
3.27.2.15	<code>bar_num_user</code>	36
3.27.2.16	<code>bar_num_bypass</code>	36
3.27.2.17	<code>bar_num_stm</code>	36
3.27.2.18	<code>qsets_base</code>	36
3.27.2.19	<code>bdf</code>	36
3.27.2.20	<code>idx</code>	36
3.27.2.21	<code>cur_cfg_state</code>	37
3.27.2.22	<code>tm_mode_en</code>	37
3.27.2.23	<code>tm_one_cdh_en</code>	37
3.27.2.24	<code>fp_user_isr_handler</code>	37
3.27.2.25	<code>fp_q_isr_top_dev</code>	37

3.28 qdma_flq Struct Reference	37
3.28.1 Detailed Description	38
3.28.2 Field Documentation	38
3.28.2.1 size	38
3.28.2.2 pg_order	38
3.28.2.3 pg_shift	38
3.28.2.4 desc	38
3.28.2.5 udd_cnt	38
3.28.2.6 pkt_cnt	38
3.28.2.7 pkt_dlen	38
3.28.2.8 avail	38
3.28.2.9 alloc_fail	38
3.28.2.10 mapping_err	39
3.28.2.11 cidx	39
3.28.2.12 pidx	39
3.28.2.13 pidx_pend	39
3.28.2.14 sdesc	39
3.28.2.15 sdesc_info	39
3.29 qdma_h2c_desc Struct Reference	39
3.29.1 Detailed Description	39
3.29.2 Field Documentation	40
3.29.2.1 cdh_flags	40
3.29.2.2 pld_len	40
3.29.2.3 len	40
3.29.2.4 flags	40
3.29.2.5 src_addr	40
3.30 qdma_intr_ring Struct Reference	40
3.30.1 Detailed Description	41
3.30.2 Field Documentation	41
3.30.2.1 pidx	41

3.30.2.2	<code>cidx</code>	41
3.30.2.3	<code>s_color</code>	41
3.30.2.4	<code>intr_satus</code>	41
3.30.2.5	<code>error</code>	41
3.30.2.6	<code>rsvd</code>	41
3.30.2.7	<code>intr_type</code>	41
3.30.2.8	<code>qid</code>	41
3.30.2.9	<code>coal_color</code>	42
3.31	<code>qdma_io_cb</code> Struct Reference	42
3.31.1	Detailed Description	42
3.31.2	Field Documentation	42
3.31.2.1	<code>buf</code>	42
3.31.2.2	<code>len</code>	42
3.31.2.3	<code>pages_nr</code>	42
3.31.2.4	<code>sgl</code>	42
3.31.2.5	<code>pages</code>	43
3.31.2.6	<code>req</code>	43
3.32	<code>qdma_kthread</code> Struct Reference	43
3.32.1	Detailed Description	43
3.32.2	Field Documentation	43
3.32.2.1	<code>lock</code>	43
3.32.2.2	<code>name</code>	44
3.32.2.3	<code>cpu</code>	44
3.32.2.4	<code>id</code>	44
3.32.2.5	<code>timeout</code>	44
3.32.2.6	<code>flag</code>	44
3.32.2.7	<code>waitq</code>	44
3.32.2.8	<code>task</code>	44
3.32.2.9	<code>work_cnt</code>	44
3.32.2.10	<code>work_list</code>	44

3.32.2.11	finit	44
3.32.2.12	fpending	45
3.32.2.13	fproc	45
3.32.2.14	fdone	45
3.33	qdma_mbox Struct Reference	45
3.33.1	Detailed Description	45
3.33.2	Field Documentation	45
3.33.2.1	lock	45
3.33.2.2	hw_tx_lock	46
3.33.2.3	hw_rx_lock	46
3.33.2.4	workq	46
3.33.2.5	xdev	46
3.33.2.6	tx_work	46
3.33.2.7	rx_work	46
3.33.2.8	rx	46
3.33.2.9	list_lock	46
3.33.2.10	tx_todo_list	46
3.33.2.11	rx_pend_list	46
3.33.2.12	timer	47
3.34	qdma_mm_desc Struct Reference	47
3.34.1	Detailed Description	47
3.34.2	Field Documentation	47
3.34.2.1	src_addr	47
3.34.2.2	flag_len	47
3.34.2.3	rsvd0	47
3.34.2.4	dst_addr	47
3.34.2.5	rsvd1	48
3.35	qdma_queue_conf Struct Reference	48
3.35.1	Detailed Description	49
3.35.2	Field Documentation	49

3.35.2.1	qidx	49
3.35.2.2	st	49
3.35.2.3	c2h	49
3.35.2.4	pipe	49
3.35.2.5	irq_en	49
3.35.2.6	desc_rng_sz_idx	49
3.35.2.7	cmpl_status_en	49
3.35.2.8	cmpl_status_acc_en	49
3.35.2.9	cmpl_status_pend_chk	50
3.35.2.10	desc_bypass	50
3.35.2.11	pfetch_en	50
3.35.2.12	fetch_credit	50
3.35.2.13	st_pkt_mode	50
3.35.2.14	c2h_use_fl	50
3.35.2.15	c2h_buf_sz_idx	50
3.35.2.16	cmpl_rng_sz_idx	50
3.35.2.17	cmpl_desc_sz	50
3.35.2.18	cmpl_stat_en	50
3.35.2.19	cmpl_udd_en	51
3.35.2.20	cmpl_timer_idx	51
3.35.2.21	cmpl_cnt_th_idx	51
3.35.2.22	cmpl_trig_mode	51
3.35.2.23	cmpl_en_intr	51
3.35.2.24	sw_desc_sz	51
3.35.2.25	pfetch_bypass	51
3.35.2.26	cmpl_ovf_chk_dis	51
3.35.2.27	port_id	51
3.35.2.28	at	51
3.35.2.29	cdh_max	52
3.35.2.30	pipe_gl_max	52

3.35.2.31 pipe_flow_id	52
3.35.2.32 pipe_slr_id	52
3.35.2.33 pipe_tdest	52
3.35.2.34 quld	52
3.35.2.35 fp_descq_isr_top	52
3.35.2.36 fp_descq_c2h_packet	52
3.35.2.37 name	53
3.35.2.38 rngsz	53
3.35.2.39 rngsz_cmpt	53
3.35.2.40 c2h_bufsz	53
3.36 qdma_request Struct Reference	53
3.36.1 Detailed Description	53
3.36.2 Field Documentation	54
3.36.2.1 opaque	54
3.36.2.2 uld_data	54
3.36.2.3 fp_done	54
3.36.2.4 timeout_ms	54
3.36.2.5 count	54
3.36.2.6 ep_addr	54
3.36.2.7 write	54
3.36.2.8 dma_mapped	54
3.36.2.9 h2c_eot	54
3.36.2.10 udd_len	54
3.36.2.11 sgcnt	55
3.36.2.12 sgl	55
3.36.2.13 udd	55
3.37 qdma_sdesc_info Struct Reference	55
3.37.1 Detailed Description	55
3.37.2 Field Documentation	55
3.37.2.1 next	55

3.37.2.2	fbits	56
3.37.2.3	valid	56
3.37.2.4	sop	56
3.37.2.5	eop	56
3.37.2.6	filler	56
3.37.2.7	rsvd	56
3.37.2.8	cidx	56
3.38	qdma_sgt_req_cb Struct Reference	56
3.38.1	Detailed Description	57
3.38.2	Field Documentation	57
3.38.2.1	list	57
3.38.2.2	wq	57
3.38.2.3	desc_nr	57
3.38.2.4	offset	57
3.38.2.5	left	57
3.38.2.6	sg_offset	57
3.38.2.7	sg_idx	58
3.38.2.8	status	58
3.38.2.9	done	58
3.38.2.10	unmap_needed	58
3.39	qdma_sw_sg Struct Reference	58
3.39.1	Detailed Description	58
3.39.2	Field Documentation	58
3.39.2.1	next	58
3.39.2.2	pg	59
3.39.2.3	offset	59
3.39.2.4	len	59
3.39.2.5	dma_addr	59
3.40	qdma_version_info Struct Reference	59
3.41	stm_descq_context Struct Reference	59

3.41.1 Detailed Description	60
3.41.2 Field Documentation	60
3.41.2.1 stm	60
3.42 xlnx_dma_dev Struct Reference	60
3.42.1 Detailed Description	61
3.42.2 Field Documentation	61
3.42.2.1 mod_name	61
3.42.2.2 conf	61
3.42.2.3 list_head	61
3.42.2.4 lock	61
3.42.2.5 hw_prg_lock	61
3.42.2.6 flr_prsnt	61
3.42.2.7 st_mode_en	61
3.42.2.8 mm_mode_en	61
3.42.2.9 stm_en	61
3.42.2.10 vf_info	62
3.42.2.11 vf_count	62
3.42.2.12 func_id	62
3.42.2.13 pf_count	62
3.42.2.14 stm_rev	62
3.42.2.15 stm_regs	62
3.42.2.16 num_vecs	62
3.42.2.17 msix	62
3.42.2.18 dev_intr_info_list	62
3.42.2.19 dvec_start_idx	62
3.42.2.20 dev_priv	63
3.42.2.21 pipe_stm_max_pkt_size	63
3.42.2.22 intr_coal_list	63
3.42.2.23 mbox	63
3.42.2.24 total_st_c2h_pkts	63

3.43 xlnx_nl_work Struct Reference	63
3.44 xlnx_nl_work_q_ctrl Struct Reference	64
3.45 xlnx_pci_dev Struct Reference	64
3.45.1 Detailed Description	64
3.45.2 Field Documentation	64
3.45.2.1 list_head	64
3.45.2.2 pdev	64
3.45.2.3 dev_hndl	65
3.45.2.4 nl_task_wq	65
3.45.2.5 cdev_cb	65
3.45.2.6 cdev_lock	65
3.45.2.7 qmax	65
3.45.2.8 idx	65
3.45.2.9 user_bar_regs	65
3.45.2.10 bypass_bar_regs	65
3.45.2.11 qdata	65
3.46 xlnx_qdata Struct Reference	66
3.46.1 Detailed Description	66
3.46.2 Field Documentation	66
3.46.2.1 qhndl	66
3.46.2.2 xcdev	66
3.47 xreg_info Struct Reference	66

4 File Documentation	67
4.1 cdev.h File Reference	67
4.1.1 Detailed Description	67
4.1.2 Macro Definition Documentation	68
4.1.2.1 QDMA_CDEV_CLASS_NAME	68
4.1.2.2 QDMA_MINOR_MAX	68
4.1.3 Function Documentation	68
4.1.3.1 qdma_cdev_destroy(struct qdma_cdev *xcdev)	68
4.1.3.2 qdma_cdev_create(struct qdma_cdev_cb *xcb, struct pci_dev *pdev, struct qdma_queue_conf *qconf, unsigned int minor, unsigned long qhndl, struct qdma_cdev **xcdev_pp, char *ebuf, int ebuflen)	68
4.1.3.3 qdma_cdev_device_cleanup(struct qdma_cdev_cb *xcb)	69
4.1.3.4 qdma_cdev_device_init(struct qdma_cdev_cb *xcb)	69
4.1.3.5 qdma_cdev_cleanup(void)	69
4.1.3.6 qdma_cdev_init(void)	69
4.2 libqdma_config.h File Reference	69
4.2.1 Detailed Description	70
4.2.2 Macro Definition Documentation	70
4.2.2.1 QDMA_CONFIG_BAR	70
4.2.2.2 STM_BAR	71
4.2.2.3 QDMA_PF_MAX	71
4.2.2.4 QDMA_VF_MAX	71
4.2.2.5 QDMA_Q_PER_PF_MAX	71
4.2.2.6 MAX_DMA_DEV	71
4.2.2.7 TOTAL_QDMA_QS	71
4.2.2.8 QDMA_Q_PER_VF_MAX	71
4.2.2.9 TOTAL_VF_QS	71
4.2.2.10 TOTAL_PF_QS	71
4.2.2.11 MAX_QS_PER_PF	71
4.2.2.12 PCI_SHIFT_BUS	72
4.2.2.13 PCI_SHIFT_DEV	72

4.2.2.14	SHIFT_DEC_PCI_BUS	72
4.2.2.15	SHIFT_DEC_PCI_DEV	72
4.2.2.16	QDMA_DEV_MSIX_VEC_MAX	72
4.2.2.17	QDMA_INTR_COAL_RING_SIZE	72
4.2.2.18	QDMA_NUM_DATA_VEC_FOR_INTR_CXT	72
4.2.3	Function Documentation	72
4.2.3.1	qdma_set_qmax(unsigned long dev_hdl, u32 qsets_max, bool forced)	72
4.2.3.2	qdma_get_qmax(unsigned long dev_hdl)	73
4.2.3.3	qdma_set_intr_rngsz(unsigned long dev_hdl, u32 rngsz)	73
4.2.3.4	qdma_get_intr_rngsz(unsigned long dev_hdl)	73
4.2.3.5	qdma_set_cmpl_status_acc(unsigned long dev_hdl, u32 cmpl_status_acc)	74
4.2.3.6	qdma_get_cmpl_status_acc(unsigned long dev_hdl)	74
4.2.3.7	qdma_set_buf_sz(unsigned long dev_hdl, u32 *buf_sz)	74
4.2.3.8	qdma_get_buf_sz(unsigned long dev_hdl, u32 *buf_sz)	74
4.2.3.9	qdma_set_glbl_rng_sz(unsigned long dev_hdl, u32 *glbl_rng_sz)	75
4.2.3.10	qdma_get_glbl_rng_sz(unsigned long dev_hdl, u32 *glbl_rng_sz)	75
4.2.3.11	qdma_set_timer_cnt(unsigned long dev_hdl, u32 *tmr_cnt)	75
4.2.3.12	qdma_get_timer_cnt(unsigned long dev_hdl, u32 *tmr_cnt)	76
4.2.3.13	qdma_set_cnt_thresh(unsigned long dev_hdl, unsigned int *cnt_th)	76
4.2.3.14	qdma_get_cnt_thresh(unsigned long dev_hdl, u32 *cnt_th)	76
4.3	libqdma_export.h File Reference	77
4.3.1	Detailed Description	79
4.3.2	Macro Definition Documentation	79
4.3.2.1	QDMA_FUNC_ID_INVALID	79
4.3.2.2	QDMA_DEV_NAME_MAXLEN	80
4.3.2.3	DEVICE_VERSION_INFO_STR_LENGTH	80
4.3.2.4	QDMA_GLOBAL_CSR_ARRAY_SZ	80
4.3.2.5	QDMA_QUEUE_NAME_MAXLEN	80
4.3.2.6	QDMA_QUEUE_IDX_INVALID	80
4.3.2.7	QDMA_QUEUE_VEC_INVALID	80

4.3.2.8	QDMA_REQ_OPAQUE_SIZE	80
4.3.2.9	QDMA_UDD_MAXLEN	80
4.3.3	Enumeration Type Documentation	81
4.3.3.1	qdma_error_codes	81
4.3.3.2	qdma_drv_mode	81
4.3.3.3	intr_ring_size_sel	81
4.3.3.4	qdma_dev_qmax_state	82
4.3.3.5	cmpt_desc_sz_t	82
4.3.3.6	desc_sz_t	82
4.3.3.7	tigger_mode_t	82
4.3.4	Function Documentation	82
4.3.4.1	libqdma_init(enum qdma_drv_mode qdma_drv_mode, unsigned int num_threads)	82
4.3.4.2	libqdma_exit(void)	83
4.3.4.3	qdma_device_open(const char *mod_name, struct qdma_dev_conf *conf, unsigned long *dev_hdl)	83
4.3.4.4	qdma_device_close(struct pci_dev *pdev, unsigned long dev_hdl)	83
4.3.4.5	qdma_device_offline(struct pci_dev *pdev, unsigned long dev_hdl)	84
4.3.4.6	qdma_device_online(struct pci_dev *pdev, unsigned long dev_hdl)	84
4.3.4.7	qdma_device_flr_quirk_set(struct pci_dev *pdev, unsigned long dev_hdl)	84
4.3.4.8	qdma_device_flr_quirk_check(struct pci_dev *pdev, unsigned long dev_hdl)	84
4.3.4.9	qdma_device_get_config(unsigned long dev_hdl, struct qdma_dev_conf *conf, char *ebuf, int ebuflen)	85
4.3.4.10	qdma_device_clear_stats(unsigned long dev_hdl)	85
4.3.4.11	qdma_device_get_mmh2c_pkts(unsigned long dev_hdl, unsigned long long *mmh2c_pkts)	85
4.3.4.12	qdma_device_get_mmc2h_pkts(unsigned long dev_hdl, unsigned long long *mmc2h_pkts)	86
4.3.4.13	qdma_device_get_sth2c_pkts(unsigned long dev_hdl, unsigned long long *sth2c_pkts)	86
4.3.4.14	qdma_device_get_stc2h_pkts(unsigned long dev_hdl, unsigned long long *stc2h_pkts)	86
4.3.4.15	qdma_device_set_config(unsigned long dev_hdl, struct qdma_dev_conf *conf)	87

4.3.4.16	<code>qdma_device_set_cfg_state(unsigned long dev_hdl, enum qdma_dev_qmax_↵ state new_cfg_state)</code>	87
4.3.4.17	<code>qdma_device_sriov_config(struct pci_dev *pdev, unsigned long dev_hdl, int num_vfs)</code>	87
4.3.4.18	<code>qdma_device_read_config_register(unsigned long dev_hdl, unsigned int reg_↵ addr)</code>	87
4.3.4.19	<code>qdma_device_write_config_register(unsigned long dev_hdl, unsigned int reg_↵ addr, u32 value)</code>	88
4.3.4.20	<code>qdma_device_version_info(unsigned long dev_hdl, struct qdma_version_info *version_info)</code>	88
4.3.4.21	<code>qdma_software_version_info(char *software_version)</code>	88
4.3.4.22	<code>qdma_global_csr_get(unsigned long dev_hdl, struct global_csr_conf *csr)</code> . . .	89
4.3.4.23	<code>qdma_global_csr_set(unsigned long dev_hdl, struct global_csr_conf *csr)</code> . . .	89
4.3.4.24	<code>qdma_queue_add(unsigned long dev_hdl, struct qdma_queue_conf *qconf, un- signed long *qhndl, char *buf, int buflen)</code>	89
4.3.4.25	<code>qdma_queue_reconfig(unsigned long dev_hdl, unsigned long id, struct qdma_↵ _queue_conf *qconf, char *buf, int buflen)</code>	90
4.3.4.26	<code>qdma_queue_start(unsigned long dev_hdl, unsigned long id, char *buf, int buflen)</code>	90
4.3.4.27	<code>qdma_queue_stop(unsigned long dev_hdl, unsigned long id, char *buf, int buflen)</code>	90
4.3.4.28	<code>qdma_queue_prog_stm(unsigned long dev_hdl, unsigned long id, char *buf, int buflen)</code>	91
4.3.4.29	<code>qdma_queue_remove(unsigned long dev_hdl, unsigned long id, char *buf, int buflen)</code>	91
4.3.4.30	<code>qdma_queue_get_config(unsigned long dev_hdl, unsigned long id, char *buf, int buflen)</code>	91
4.3.4.31	<code>qdma_queue_list(unsigned long dev_hdl, char *buf, int buflen)</code>	92
4.3.4.32	<code>qdma_queue_dump(unsigned long dev_hdl, unsigned long id, char *buf, int buflen)</code>	92
4.3.4.33	<code>qdma_queue_dump_desc(unsigned long dev_hdl, unsigned long id, unsigned int start, unsigned int end, char *buf, int buflen)</code>	92
4.3.4.34	<code>qdma_queue_dump_cmpt(unsigned long dev_hdl, unsigned long id, unsigned int start, unsigned int end, char *buf, int buflen)</code>	93
4.3.4.35	<code>qdma_request_submit(unsigned long dev_hdl, unsigned long id, struct qdma_↵ _request *req)</code>	93
4.3.4.36	<code>qdma_batch_request_submit(unsigned long dev_hdl, unsigned long id, un- signed long count, struct qdma_request **reqv)</code>	93
4.3.4.37	<code>qdma_queue_c2h_peek(unsigned long dev_hdl, unsigned long qhndl, unsigned int *udd_cnt, unsigned int *pkt_cnt, unsigned int *data_len)</code>	94

4.3.4.38	<code>qdma_queue_avail_desc(unsigned long dev_hdl, unsigned long qhdl)</code>	94
4.3.4.39	<code>qdma_queue_cmpl_ctrl(unsigned long dev_hdl, unsigned long qhdl, struct qdma_cmpl_ctrl *cctrl, bool set)</code>	94
4.3.4.40	<code>qdma_queue_packet_read(unsigned long dev_hdl, unsigned long qhdl, struct qdma_request *req, struct qdma_cmpl_ctrl *cctrl)</code>	95
4.3.4.41	<code>qdma_queue_packet_write(unsigned long dev_hdl, unsigned long qhdl, struct qdma_request *req)</code>	95
4.3.4.42	<code>qdma_queue_service(unsigned long dev_hdl, unsigned long qhdl, int budget, bool c2h_upd_cmpl)</code>	96
4.3.4.43	<code>qdma_intr_ring_dump(unsigned long dev_hdl, unsigned int vector_idx, int start_idx, int end_idx, char *buf, int buflen)</code>	96
4.3.4.44	<code>qdma_descq_get_cmpt_udd(unsigned long dev_hdl, unsigned long qhdl, char *buf, int buflen)</code>	96
4.4	<code>nl.h</code> File Reference	97
4.4.1	Detailed Description	97
4.4.2	Function Documentation	97
4.4.2.1	<code>xnl_respond_buffer(struct genl_info *info, char *buf, int buflen)</code>	97
4.5	<code>pci_ids.h</code> File Reference	97
4.5.1	Detailed Description	97
4.5.2	Function Documentation	98
4.5.2.1	<code>MODULE_DEVICE_TABLE(pci, pci_ids)</code>	98
4.6	<code>qdma_compat.h</code> File Reference	98
4.6.1	Detailed Description	98
4.6.2	Macro Definition Documentation	98
4.6.2.1	<code>qdma_wait_queue</code>	98
4.7	<code>qdma_context.h</code> File Reference	98
4.7.1	Detailed Description	99
4.7.2	Function Documentation	99
4.7.2.1	<code>qdma_intr_context_setup(struct xlnx_dma_dev *xdev)</code>	99
4.7.2.2	<code>qdma_prog_intr_context(struct xlnx_dma_dev *xdev, struct mbox_msg_intr_ctxt *ictxt)</code>	99
4.7.2.3	<code>qdma_descq_context_setup(struct qdma_descq *descq)</code>	100
4.7.2.4	<code>qdma_descq_stm_setup(struct qdma_descq *descq)</code>	100

4.7.2.5	qdma_descq_stm_clear(struct qdma_descq *descq)	100
4.7.2.6	qdma_descq_context_clear(struct xlnx_dma_dev *xdev, unsigned int qid_hw, bool st, bool c2h, bool clr)	101
4.7.2.7	qdma_descq_context_read(struct xlnx_dma_dev *xdev, unsigned int qid_hw, bool st, bool c2h, struct hw_descq_context *ctxt)	102
4.7.2.8	qdma_intr_context_read(struct xlnx_dma_dev *xdev, int ring_index, unsigned int ctxt_sz, u32 *context)	102
4.7.2.9	qdma_descq_context_program(struct xlnx_dma_dev *xdev, unsigned int qid_hw, bool st, bool c2h, struct hw_descq_context *ctxt)	103
4.7.2.10	qdma_descq_stm_read(struct xlnx_dma_dev *xdev, unsigned int qid_hw, u8 pipe_flow_id, bool c2h, bool map, bool ctxt, struct stm_descq_context *context)	103
4.7.2.11	qdma_descq_stm_program(struct xlnx_dma_dev *xdev, unsigned int qid_hw, uint8_t pipe_flow_id, bool c2h, bool clear, struct stm_descq_context *context)	103
4.8	qdma_descq.h File Reference	104
4.8.1	Detailed Description	105
4.8.2	Macro Definition Documentation	105
4.8.2.1	lock_descq	105
4.8.2.2	unlock_descq	105
4.8.2.3	qdma_req_cb_get	105
4.8.3	Enumeration Type Documentation	105
4.8.3.1	q_state_t	105
4.8.4	Function Documentation	106
4.8.4.1	qdma_descq_init(struct qdma_descq *descq, struct xlnx_dma_dev *xdev, int idx_hw, int idx_sw)	106
4.8.4.2	qdma_descq_config(struct qdma_descq *descq, struct qdma_queue_conf *qconf, int reconfig)	107
4.8.4.3	qdma_descq_config_complete(struct qdma_descq *descq)	107
4.8.4.4	qdma_descq_cleanup(struct qdma_descq *descq)	107
4.8.4.5	qdma_descq_alloc_resource(struct qdma_descq *descq)	108
4.8.4.6	qdma_descq_free_resource(struct qdma_descq *descq)	108
4.8.4.7	qdma_descq_prog_hw(struct qdma_descq *descq)	108
4.8.4.8	qdma_descq_prog_stm(struct qdma_descq *descq, bool clear)	108
4.8.4.9	qdma_descq_context_cleanup(struct qdma_descq *descq)	109

4.8.4.10	<code>qdma_descq_service_cmpl_update(struct qdma_descq *descq, int budget, bool c2h_upd_cmpl)</code>	109
4.8.4.11	<code>qdma_descq_dump(struct qdma_descq *descq, char *buf, int buflen, int detail)</code>	109
4.8.4.12	<code>qdma_descq_dump_desc(struct qdma_descq *descq, int start, int end, char *buf, int buflen)</code>	110
4.8.4.13	<code>qdma_descq_dump_state(struct qdma_descq *descq, char *buf, int buflen)</code>	110
4.8.4.14	<code>intr_cidx_update(struct qdma_descq *descq, unsigned int sw_cidx, int ring_index)</code>	110
4.8.4.15	<code>incr_cmpl_desc_cnt(struct qdma_descq *descq, unsigned int cnt)</code>	111
4.8.4.16	<code>qdma_descq_proc_sgt_request(struct qdma_descq *descq)</code>	111
4.8.4.17	<code>qdma_sgt_req_done(struct qdma_descq *descq, struct qdma_sgt_req_cb *cb, int error)</code>	111
4.8.4.18	<code>sgl_map(struct pci_dev *pdev, struct qdma_sw_sg *sg, unsigned int sgcnt, enum dma_data_direction dir)</code>	111
4.8.4.19	<code>sgl_unmap(struct pci_dev *pdev, struct qdma_sw_sg *sg, unsigned int sgcnt, enum dma_data_direction dir)</code>	112
4.8.4.20	<code>descq_flq_free_resource(struct qdma_descq *descq)</code>	112
4.9	<code>qdma_device.h</code> File Reference	112
4.9.1	Detailed Description	113
4.9.2	Macro Definition Documentation	113
4.9.2.1	<code>xdev_2_qdev</code>	113
4.9.3	Enumeration Type Documentation	113
4.9.3.1	<code>csr_type</code>	113
4.9.4	Function Documentation	113
4.9.4.1	<code>qdma_device_init(struct xlnx_dma_dev *xdev)</code>	113
4.9.4.2	<code>qdma_device_cleanup(struct xlnx_dma_dev *xdev)</code>	114
4.9.4.3	<code>qdma_device_get_id_from_descq(struct xlnx_dma_dev *xdev, struct qdma_descq *descq)</code>	114
4.9.4.4	<code>qdma_device_get_descq_by_id(struct xlnx_dma_dev *xdev, unsigned long idx, char *buf, int buflen, int init)</code>	114
4.9.4.5	<code>qdma_device_get_descq_by_hw_qid(struct xlnx_dma_dev *xdev, unsigned long qidx_hw, u8 c2h)</code>	115
4.9.4.6	<code>qdma_device_prep_q_resource(struct xlnx_dma_dev *xdev)</code>	115
4.9.4.7	<code>qdma_csr_read_cmpl_status_acc(struct xlnx_dma_dev *xdev, unsigned int *cs_acc)</code>	115

4.9.4.8	qdma_csr_read_rngsz(struct xlnx_dma_dev *xdev, unsigned int *rngsz)	116
4.9.4.9	qdma_csr_read_bufsz(struct xlnx_dma_dev *xdev, unsigned int *bufsz)	116
4.9.4.10	qdma_csr_read_timer_cnt(struct xlnx_dma_dev *xdev, unsigned int *cnt)	116
4.9.4.11	qdma_csr_read_cnt_thresh(struct xlnx_dma_dev *xdev, unsigned int *th)	116
4.9.4.12	qdma_csr_read(struct xlnx_dma_dev *xdev, struct qdma_csr_info *csr_info, unsigned int timeout_ms)	117
4.10	qdma_intr.h File Reference	117
4.10.1	Detailed Description	118
4.10.2	Function Documentation	118
4.10.2.1	intr_teardown(struct xlnx_dma_dev *xdev)	118
4.10.2.2	intr_setup(struct xlnx_dma_dev *xdev)	118
4.10.2.3	intr_ring_teardown(struct xlnx_dma_dev *xdev)	118
4.10.2.4	intr_context_setup(struct xlnx_dma_dev *xdev)	119
4.10.2.5	intr_ring_setup(struct xlnx_dma_dev *xdev)	119
4.10.2.6	intr_legacy_init(void)	119
4.10.2.7	intr_legacy_setup(struct qdma_descq *descq)	119
4.10.2.8	intr_legacy_clear(struct qdma_descq *descq)	119
4.10.2.9	intr_work(struct work_struct *work)	120
4.10.2.10	qdma_err_intr_setup(struct xlnx_dma_dev *xdev, u8 rearm)	120
4.10.2.11	qdma_enable_hw_err(struct xlnx_dma_dev *xdev, u8 hw_err_type)	120
4.10.2.12	get_intr_ring_index(struct xlnx_dma_dev *xdev, u32 vector_index)	121
4.11	qdma_mbox.h File Reference	121
4.11.1	Detailed Description	123
4.11.2	Macro Definition Documentation	123
4.11.2.1	MBOX_BASE	123
4.11.2.2	MBOX_FN_STATUS	123
4.11.2.3	S_MBOX_FN_STATUS_IN_MSG	124
4.11.2.4	M_MBOX_FN_STATUS_IN_MSG	124
4.11.2.5	F_MBOX_FN_STATUS_IN_MSG	124
4.11.2.6	S_MBOX_FN_STATUS_OUT_MSG	124
4.11.2.7	M_MBOX_FN_STATUS_OUT_MSG	124

4.11.2.8 F_MBOX_FN_STATUS_OUT_MSG	124
4.11.2.9 S_MBOX_FN_STATUS_ACK	124
4.11.2.10 M_MBOX_FN_STATUS_ACK	124
4.11.2.11 F_MBOX_FN_STATUS_ACK	124
4.11.2.12 S_MBOX_FN_STATUS_SRC	124
4.11.2.13 M_MBOX_FN_STATUS_SRC	125
4.11.2.14 G_MBOX_FN_STATUS_SRC	125
4.11.2.15 MBOX_FN_STATUS_MASK	125
4.11.2.16 MBOX_FN_CMD	125
4.11.2.17 S_MBOX_FN_CMD_SND	125
4.11.2.18 M_MBOX_FN_CMD_SND	125
4.11.2.19 F_MBOX_FN_CMD_SND	125
4.11.2.20 S_MBOX_FN_CMD_RCV	125
4.11.2.21 M_MBOX_FN_CMD_RCV	125
4.11.2.22 F_MBOX_FN_CMD_RCV	126
4.11.2.23 S_MBOX_FN_CMD_VF_RESET	126
4.11.2.24 M_MBOX_FN_CMD_VF_RESET	126
4.11.2.25 MBOX_ISR_VEC	126
4.11.2.26 S_MBOX_ISR_VEC	126
4.11.2.27 M_MBOX_ISR_VEC	126
4.11.2.28 V_MBOX_ISR_VEC	126
4.11.2.29 MBOX_FN_TARGET	126
4.11.2.30 S_MBOX_FN_TARGET_ID	126
4.11.2.31 M_MBOX_FN_TARGET_ID	126
4.11.2.32 V_MBOX_FN_TARGET_ID	127
4.11.2.33 MBOX_ISR_EN	127
4.11.2.34 S_MBOX_ISR_EN	127
4.11.2.35 M_MBOX_ISR_EN	127
4.11.2.36 F_MBOX_ISR_EN	127
4.11.2.37 MBOX_PF_ACK_BASE	127

4.11.2.38	MBOX_PF_ACK_STEP	127
4.11.2.39	MBOX_PF_ACK_COUNT	127
4.11.2.40	MBOX_IN_MSG_BASE	127
4.11.2.41	MBOX_OUT_MSG_BASE	127
4.11.2.42	MBOX_MSG_STEP	128
4.11.2.43	MBOX_MSG_REG_MAX	128
4.11.2.44	MBOX_MSG_OP_PF_MASK	128
4.11.3	Enumeration Type Documentation	128
4.11.3.1	mbox_msg_op	128
4.11.4	Function Documentation	128
4.11.4.1	qdma_mbox_init(struct xlnx_dma_dev *xdev)	128
4.11.4.2	qdma_mbox_cleanup(struct xlnx_dma_dev *xdev)	129
4.11.4.3	qdma_mbox_msg_send(struct xlnx_dma_dev *xdev, struct mbox_msg *m, bool wait_resp, u8 resp_op, unsigned int timeout_ms)	129
4.11.4.4	qdma_mbox_msg_alloc(struct xlnx_dma_dev *xdev, enum mbox_msg_op op)	129
4.11.4.5	__qdma_mbox_msg_free(const char *fname, struct mbox_msg *m)	130
4.12	qdma_mod.h File Reference	130
4.12.1	Detailed Description	131
4.12.2	Function Documentation	131
4.12.2.1	xpdev_list_dump(char *buf, int buflen)	131
4.12.2.2	xpdev_find_by_idx(unsigned int idx, char *ebuf, int ebuflen)	131
4.12.2.3	xpdev_queue_get(struct xlnx_pci_dev *xpdev, unsigned int qidx, bool c2h, bool check_ghndl, char *ebuf, int ebuflen)	131
4.12.2.4	xpdev_queue_add(struct xlnx_pci_dev *xpdev, struct qdma_queue_conf *qconf, char *ebuf, int ebuflen)	132
4.12.2.5	xpdev_queue_delete(struct xlnx_pci_dev *xpdev, unsigned int qidx, bool c2h, char *ebuf, int ebuflen)	132
4.12.2.6	qdma_device_read_user_register(struct xlnx_pci_dev *xpdev, unsigned int reg↵ _addr)	132
4.12.2.7	qdma_device_write_user_register(struct xlnx_pci_dev *xpdev, unsigned int reg↵ _addr, u32 value)	133
4.12.2.8	qdma_device_read_bypass_register(struct xlnx_pci_dev *xpdev, unsigned int reg_addr)	133

4.12.2.9	qdma_device_write_bypass_register(struct xlnx_pci_dev *xpdev, unsigned int reg_addr, u32 value)	133
4.13	qdma_nl.h File Reference	133
4.13.1	Detailed Description	138
4.13.2	Macro Definition Documentation	138
4.13.2.1	XNL_NAME_PF	138
4.13.2.2	XNL_NAME_VF	138
4.13.2.3	XNL_VERSION	138
4.13.2.4	XNL_RESP_BUFLLEN_MIN	138
4.13.2.5	XNL_RESP_BUFLLEN_MAX	138
4.13.2.6	XNL_ERR_BUFLLEN	138
4.13.2.7	XNL_STR_LEN_MAX	138
4.13.2.8	XNL_QIDX_INVALID	138
4.13.2.9	XNL_F_QMODE_ST	139
4.13.2.10	XNL_F_QMODE_MM	139
4.13.2.11	XNL_F_QDIR_H2C	139
4.13.2.12	XNL_F_QDIR_C2H	139
4.13.2.13	XNL_F_QDIR_BOTH	139
4.13.2.14	XNL_F_PFETCH_EN	139
4.13.2.15	XNL_F_DESC_BYPASS_EN	139
4.13.2.16	XNL_F_FETCH_CREDIT	139
4.13.2.17	XNL_F_CMPL_STATUS_ACC_EN	139
4.13.2.18	XNL_F_CMPL_STATUS_EN	139
4.13.2.19	XNL_F_CMPL_STATUS_PEND_CHK	140
4.13.2.20	XNL_F_CMPL_STATUS_DESC_EN	140
4.13.2.21	XNL_F_C2H_CMPL_INTR_EN	140
4.13.2.22	XNL_F_CMPL_UDD_EN	140
4.13.2.23	XNL_F_PFETCH_BYPASS_EN	140
4.13.2.24	XNL_F_CMPT_OVF_CHK_DIS	140
4.13.2.25	MAX_QFLAGS	140
4.13.2.26	QDMA_MAX_INT_RING_ENTRIES	140

4.13.3	Enumeration Type Documentation	141
4.13.3.1	xnl_attr_t	141
4.13.3.2	xnl_st_c2h_cmpt_desc_size	142
4.13.3.3	xnl_op_t	142
4.13.3.4	qdma_err_idx	142
4.14	qdma_st_c2h.h File Reference	143
4.14.1	Detailed Description	143
4.14.2	Function Documentation	143
4.14.2.1	qdma_descq_rxq_read(struct qdma_descq *descq, struct qdma_request *req)	143
4.14.2.2	qdma_descq_dump_cmpt(struct qdma_descq *descq, int start, int end, char *buf, int buflen)	143
4.14.2.3	incr_cmpl_desc_cnt(struct qdma_descq *descq, unsigned int cnt)	144
4.14.2.4	descq_flq_free_resource(struct qdma_descq *descq)	144
4.14.2.5	descq_flq_alloc_resource(struct qdma_descq *descq)	144
4.14.2.6	descq_process_completion_st_c2h(struct qdma_descq *descq, int budget, bool upd_cmpl)	145
4.14.2.7	descq_st_c2h_read(struct qdma_descq *descq, struct qdma_request *req, bool update, bool refill)	145
4.15	qdma_thread.h File Reference	145
4.15.1	Detailed Description	146
4.15.2	Function Documentation	146
4.15.2.1	qdma_threads_create(unsigned int num_threads)	146
4.15.2.2	qdma_threads_destroy(void)	146
4.15.2.3	qdma_thread_remove_work(struct qdma_descq *descq)	146
4.15.2.4	qdma_thread_add_work(struct qdma_descq *descq)	146
4.16	thread.h File Reference	147
4.16.1	Detailed Description	147
4.16.2	Macro Definition Documentation	147
4.16.2.1	lock_thread	147
4.16.2.2	unlock_thread	148
4.16.2.3	qdma_kthread_wakeup	148
4.16.2.4	pr_debug_thread	148

4.16.3	Function Documentation	148
4.16.3.1	qdma_kthread_dump(struct qdma_kthread *thp, char *buf, int buflen, int detail)	148
4.16.3.2	qdma_kthread_start(struct qdma_kthread *thp, char *name, int id)	148
4.16.3.3	qdma_kthread_stop(struct qdma_kthread *thp)	149
4.17	xdev.h File Reference	149
4.17.1	Detailed Description	150
4.17.2	Macro Definition Documentation	151
4.17.2.1	QDMA_BAR_NUM	151
4.17.2.2	QDMA_MAX_BAR_LEN_MAPPED	151
4.17.2.3	QDMA_DESC_BLEN_BITS	151
4.17.2.4	QDMA_DESC_BLEN_MAX	151
4.17.2.5	PCI_DMA_H	151
4.17.2.6	PCI_DMA_L	151
4.17.2.7	XDEV_FLAG_OFFLINE	151
4.17.2.8	XDEV_FLAG_IRQ	151
4.17.2.9	XDEV_NUM_IRQ_MAX	151
4.17.2.10	RTL1_VERSION	151
4.17.2.11	xdev_sriov_disable	152
4.17.2.12	xdev_sriov_enable	152
4.17.2.13	xdev_sriov_vf_offline	152
4.17.2.14	xdev_sriov_vf_online	152
4.17.3	Typedef Documentation	152
4.17.3.1	f_intr_handler	152
4.17.4	Enumeration Type Documentation	152
4.17.4.1	intr_type_list	152
4.17.5	Function Documentation	152
4.17.5.1	xdev_find_by_pdev(struct pci_dev *pdev)	152
4.17.5.2	xdev_find_by_idx(int idx)	153
4.17.5.3	xdev_list_first(void)	153
4.17.5.4	xdev_list_next(struct xlnx_dma_dev *xdev)	153
4.17.5.5	xdev_list_dump(char *buf, int buflen)	154
4.17.5.6	xdev_check_hdl(const char *f, struct pci_dev *pdev, unsigned long hndl)	154

Chapter 1

Data Structure Index

1.1 Data Structures

Here are the data structures with brief descriptions:

descriptor	5
descq_csr_info	
Qdma Q csr register settings	5
drv_mode_name	5
free_entry	6
global_csr_conf	6
hw_descq_context	
Queue context information	7
intr_coal_conf	
Interrupt coalescing configuration	8
intr_info_t	9
intr_vec_map_type	
Interrupt vector map details	10
mbox_msg	
Mailbox message	11
mbox_msg_csr	
Mailbox csr reading message	12
mbox_msg_fmap	
FMAP programming command	13
mbox_msg_hdr	
Mailbox message header	14
mbox_msg_intr_ctxt	
Interrupt context mailbox message	14
mbox_msg_qctxt	
Queue context mailbox message header	15
qconf_entry	17
qconf_entry_head	18
qdma_c2h_cmpt_cmpl_status	
Qdma completion data descriptor	20
qdma_c2h_desc	
Qdma c2h descriptor	20
qdma_cdev	
QDMA character device book keeping parameters	21
qdma_cdev_cb	
QDMA character device call back data	23

qdma_cmpl_ctrl	24
qdma_csr_info	25
qdma_desc_cmpl_status	
Qdma writeback descriptor	26
qdma_descq	
Qdma software descriptor book keeping fields	26
qdma_dev	
Qdma device per function	32
qdma_dev_conf	34
qdma_flq	
Qdma free list q page allocation book keeping	37
qdma_h2c_desc	
Memory mapped descriptor format	39
qdma_intr_ring	
Interrupt ring entry definition	40
qdma_io_cb	
QDMA character device io call back book keeping parameters	42
qdma_kthread	
Qdma thread book keeping parameters	43
qdma_mbox	
Mailbox book keeping structure	45
qdma_mm_desc	
Memory mapped descriptor format	47
qdma_queue_conf	48
qdma_request	53
qdma_sdesc_info	
Qdma descriptor information	55
qdma_sgt_req_cb	
Qdma_sgt_req_cb fits in qdma_request.opaque	56
qdma_sw_sg	58
qdma_version_info	59
stm_descq_context	
Queue stm information	59
xlnx_dma_dev	
Xilinx DMA device details	60
xlnx_nl_work	63
xlnx_nl_work_q_ctrl	64
xlnx_pci_dev	
Xilinx pcie device data members	64
xlnx_qdata	
Queue data variables send while read/write request	66
xreg_info	66

Chapter 2

File Index

2.1 File List

Here is a list of all documented files with brief descriptions:

cdev.h	This file contains the declarations for qdma pcie kernel module	67
libqdma_config.h	This file contains the declarations for qdma configuration apis	69
libqdma_export.h	This file contains the declarations for libqdma interfaces	77
nl.h	This file contains the declarations for qdma netlink helper funnctions kernel module	97
pci_ids.h	This file contains the list of pcie devices supported for qdma driver	97
qdma_compat.h	This file is used to allow the driver to be compiled under multiple versions of Linux with as few obtrusive in-line ifdef's as possible	98
qdma_context.h	This file contains the declarations for qdma context handlers	98
qdma_debugfs.h	??
qdma_debugfs_dev.h	??
qdma_debugfs_queue.h	??
qdma_descq.h	This file contains the declarations for qdma descq processing	104
qdma_device.h	This file contains the declarations for QDMA device	112
qdma_intr.h	This file contains the declarations for qdma dev interrupt handlers	117
qdma_mbox.h	This file contains the declarations for qdma mailbox apis	121
qdma_mod.h	This file contains the declarations for qdma pcie kernel module	130
qdma_nl.h	This file contains the declarations for qdma netlink interfaces	133
qdma_qconf_mgr.h	??
qdma_regs.h	??
qdma_st_c2h.h	This file contains the declarations for qdma st c2h processing	143
qdma_thread.h	This file contains the declarations for qdma thread handlers	145

thread.h	
This file contains the declarations for qdma kernel threads	147
drv/version.h	??
libqdma/version.h	??
xdev.h	
This file contains the declarations for QDMA PCIe device	149
xdev_regs.h	??

Chapter 3

Data Structure Documentation

3.1 descriptor Union Reference

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

- [qdma_st_c2h.h](#)

3.2 descq_csr_info Struct Reference

```
#include <qdma_device.h>
```

3.2.1 Detailed Description

qdma Q csr register settings

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

- [qdma_device.h](#)

3.3 drv_mode_name Struct Reference

Data Fields

- enum [qdma_drv_mode](#) **drv_mode**
- char **name** [20]

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

- [libqdma_export.h](#)

3.4 free_entry Struct Reference

Data Fields

- struct list_head [list_head](#)
- u32 [next_qbase](#)
- u32 [free](#)
- u32 [index](#)

3.4.1 Field Documentation

3.4.1.1 struct list_head free_entry::list_head

to connect to free_list

3.4.1.2 u32 free_entry::next_qbase

next available qbase

3.4.1.3 u32 free_entry::free

free qs available in this entry

3.4.1.4 u32 free_entry::index

index of the entry

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

- [qdma_qconf_mgr.h](#)

3.5 global_csr_conf Struct Reference

```
#include <libqdma_export.h>
```

Data Fields

- unsigned int [ring_sz](#) [QDMA_GLOBAL_CSR_ARRAY_SZ]
- unsigned int [c2h_timer_cnt](#) [QDMA_GLOBAL_CSR_ARRAY_SZ]
- unsigned int [c2h_cnt_th](#) [QDMA_GLOBAL_CSR_ARRAY_SZ]
- unsigned int [c2h_buf_sz](#) [QDMA_GLOBAL_CSR_ARRAY_SZ]
- unsigned int [cpl_status_acc](#)

3.5.1 Detailed Description

struct [global_csr_conf](#) - global CSR configuration

3.5.2 Field Documentation

3.5.2.1 unsigned int `global_csr_conf::ring_sz`[QDMA_GLOBAL_CSR_ARRAY_SZ]

Descriptor ring size ie. queue depth

3.5.2.2 unsigned int `global_csr_conf::c2h_timer_cnt`[QDMA_GLOBAL_CSR_ARRAY_SZ]

C2H timer count list

3.5.2.3 unsigned int `global_csr_conf::c2h_cnt_th`[QDMA_GLOBAL_CSR_ARRAY_SZ]

C2H counter threshold list

3.5.2.4 unsigned int `global_csr_conf::c2h_buf_sz`[QDMA_GLOBAL_CSR_ARRAY_SZ]

C2H buffer size list

3.5.2.5 unsigned int `global_csr_conf::cmpl_status_acc`

wireback acculation enable/disable

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

- [libqdma_export.h](#)

3.6 hw_descq_context Struct Reference

```
#include <qdma_mbox.h>
```

Data Fields

- u32 [sw](#) [5]
- u32 [prefetch](#) [2]
- u32 [cmpt](#) [5]
- u32 [hw](#) [2]
- u32 [cr](#) [1]
- u32 [fmap](#) [2]

3.6.1 Detailed Description

queue context information

3.6.2 Field Documentation

3.6.2.1 u32 hw_descq_context::sw[5]

software descriptor context data: 4 data words

3.6.2.2 u32 hw_descq_context::prefetch[2]

prefetch context data: 2 data words

3.6.2.3 u32 hw_descq_context::cmpt[5]

queue completion context data: 4 data words

3.6.2.4 u32 hw_descq_context::hw[2]

hardware descriptor context data: 2 data words

3.6.2.5 u32 hw_descq_context::cr[1]

C2H or H2C context: 1 data word

3.6.2.6 u32 hw_descq_context::fmap[2]

FMAP context data

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

- [qdma_mbox.h](#)

3.7 intr_coal_conf Struct Reference

```
#include <xdev.h>
```

Data Fields

- u16 [vec_id](#)
- u16 [intr_rng_num_entries](#)
- dma_addr_t [intr_ring_bus](#)
- struct [qdma_intr_ring](#) * [intr_ring_base](#)
- u8 [color](#)
- unsigned int [cidx](#)

3.7.1 Detailed Description

interrut coalescing configuration

3.7.2 Field Documentation

3.7.2.1 u16 intr_coal_conf::vec_id

< interrupt vector index number of entries in interrupt ring per vector

3.7.2.2 u16 intr_coal_conf::intr_rng_num_entries

interrupt ring base address

3.7.2.3 struct [qdma_intr_ring](#)* intr_coal_conf::intr_ring_base

color value indicates the valid entry in the interrupt ring

3.7.2.4 u8 intr_coal_conf::color

interrupt ring consumer index

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

- [xdev.h](#)

3.8 intr_info_t Struct Reference

Data Fields

- char [msix_name](#) [[QDMA_DEV_NAME_MAXLEN](#)+16]
- struct list_head [intr_list](#)
- int [intr_list_cnt](#)
- struct [intr_vec_map_type](#) [intr_vec_map](#)

3.8.1 Field Documentation

3.8.1.1 `char intr_info_t::msix_name[QDMA_DEV_NAME_MAXLEN+16]`

< msix_entry list for all vectors queue list for each interrupt

3.8.1.2 `struct list_head intr_info_t::intr_list`

number of queues assigned for each interrupt

3.8.1.3 `int intr_info_t::intr_list_cnt`

interrupt vector map

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

- [xdev.h](#)

3.9 `intr_vec_map_type` Struct Reference

```
#include <xdev.h>
```

Data Fields

- enum [intr_type_list](#) `intr_type`
- int [intr_vec_index](#)
- [f_intr_handler](#) `intr_handler`

3.9.1 Detailed Description

interrupt vector map details

Interrupt info for MSI-X interrupt vectors per device

3.9.2 Field Documentation

3.9.2.1 `enum intr_type_list intr_vec_map_type::intr_type`

interrupt type

3.9.2.2 `int intr_vec_map_type::intr_vec_index`

interrupt vector index

3.9.2.3 f_intr_handler intr_vec_map_type::intr_handler

interrupt handler

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

- [xdev.h](#)

3.10 mbox_msg Struct Reference

```
#include <qdma_mbox.h>
```

Data Fields

- struct work_struct **work**
- struct list_head [list](#)
- [qdma_wait_queue](#) **waitq**
- struct kref **refcnt**
- u8 **wait_resp**
- u8 **wait_op**
- u8 **rsvd** [2]
- union {
 - struct [mbox_msg_hdr](#) **hdr**
 - struct [mbox_msg_fmap](#) **fmap**
 - struct [mbox_msg_intr_ctxt](#) **intr_ctxt**
 - struct [mbox_msg_qctxt](#) **qctxt**
 - struct [mbox_msg_csr](#) **csr**
 - u32 **raw** [MBOX_MSG_REG_MAX]

3.10.1 Detailed Description

mailbox message

3.10.2 Field Documentation

3.10.2.1 struct list_head mbox_msg::list

workqueue item

3.10.2.2 qdma_wait_queue mbox_msg::waitq

message list

3.10.2.3 struct mbox_msg_hdr mbox_msg::hdr

mailbox message header

3.10.2.4 struct mbox_msg_fmap mbox_msg::fmap

fmap mailbox message

3.10.2.5 struct mbox_msg_intr_ctxt mbox_msg::intr_ctxt

interrupt context mailbox message

3.10.2.6 struct mbox_msg_qctxt mbox_msg::qctxt

queue context mailbox message

3.10.2.7 struct mbox_msg_csr mbox_msg::csr

global csr mailbox message

3.10.2.8 u32 mbox_msg::raw[MBOX_MSG_REG_MAX]

buffer to hold raw data between pf and vf

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

- [qdma_mbox.h](#)

3.11 mbox_msg_csr Struct Reference

```
#include <qdma_mbox.h>
```

Data Fields

- struct [mbox_msg_hdr](#) **hdr**
- struct [qdma_csr_info](#) **csr_info**

3.11.1 Detailed Description

mailbox csr reading message

3.11.2 Field Documentation

3.11.2.1 struct mbox_msg_hdr mbox_msg_csr::hdr

mailbox message header

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

- [qdma_mbox.h](#)

3.12 mbox_msg_fmap Struct Reference

```
#include <qdma_mbox.h>
```

Data Fields

- struct [mbox_msg_hdr](#) [hdr](#)
- unsigned int [qbase](#)
- unsigned int [qmax](#)

3.12.1 Detailed Description

FMAP programming command.

3.12.2 Field Documentation

3.12.2.1 struct mbox_msg_hdr mbox_msg_fmap::hdr

mailbox message header

3.12.2.2 unsigned int mbox_msg_fmap::qbase

start queue number in the queue range

3.12.2.3 unsigned int mbox_msg_fmap::qmax

max queue number in the queue range(0-2k)

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

- [qdma_mbox.h](#)

3.13 mbox_msg_hdr Struct Reference

```
#include <qdma_mbox.h>
```

Data Fields

- u8 **op**
- u16 **src**
- u16 **dst**
- char **status**

3.13.1 Detailed Description

mailbox message header

3.13.2 Field Documentation

3.13.2.1 u16 mbox_msg_hdr::src

opcode

3.13.2.2 u16 mbox_msg_hdr::dst

src function

3.13.2.3 char mbox_msg_hdr::status

dst function

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

- [qdma_mbox.h](#)

3.14 mbox_msg_intr_ctxt Struct Reference

```
#include <qdma_mbox.h>
```

Data Fields

- struct [mbox_msg_hdr](#) **hdr**
- u16 **clear**:1
- u16 **filler**:15
- u8 **vec_base**
- u8 **num_rings**
- u32 **ring_index_list** [QDMA_NUM_DATA_VEC_FOR_INTR_CXT]
- u32 **w** [QDMA_NUM_DATA_VEC_FOR_INTR_CXT *3]

3.14.1 Detailed Description

interrupt context mailbox message

3.14.2 Field Documentation

3.14.2.1 struct mbox_msg_hdr mbox_msg_intr_ctxt::hdr

mailbox message header

3.14.2.2 u16 mbox_msg_intr_ctxt::clear

flag to indicate clear interrupt context

3.14.2.3 u16 mbox_msg_intr_ctxt::filler

filler variable

3.14.2.4 u8 mbox_msg_intr_ctxt::vec_base

start vector number

3.14.2.5 u8 mbox_msg_intr_ctxt::num_rings

number of intr context rings be assigned for virtual function

3.14.2.6 u32 mbox_msg_intr_ctxt::ring_index_list[QDMA_NUM_DATA_VEC_FOR_INTR_CXT]

ring index associated for each vector

3.14.2.7 u32 mbox_msg_intr_ctxt::w[QDMA_NUM_DATA_VEC_FOR_INTR_CXT * 3]

interrupt context data for all rings

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

- [qdma_mbox.h](#)

3.15 mbox_msg_qctx Struct Reference

```
#include <qdma_mbox.h>
```

Data Fields

- struct [mbox_msg_hdr](#) `hdr`
- u8 [clear](#):1
- u8 [verify](#):1
- u8 [c2h](#):1
- u8 [st](#):1
- u8 [intr_en](#):1
- u8 [intr_id](#)
- unsigned short [qid](#)
- struct [hw_descq_context](#) `context`

3.15.1 Detailed Description

queue context mailbox message header

3.15.2 Field Documentation

3.15.2.1 struct [mbox_msg_hdr](#) [mbox_msg_qctxt::hdr](#)

mailbox message header

3.15.2.2 u8 [mbox_msg_qctxt::clear](#)

flag to indicate to clear the queue context

3.15.2.3 u8 [mbox_msg_qctxt::verify](#)

flag to indicate to verify the queue context

3.15.2.4 u8 [mbox_msg_qctxt::c2h](#)

queue direction

3.15.2.5 u8 [mbox_msg_qctxt::st](#)

queue mode

3.15.2.6 u8 [mbox_msg_qctxt::intr_en](#)

flag to indicate to enable the interrupts

3.15.2.7 u8 mbox_msg_qctxt::intr_id

interrupt id

3.15.2.8 unsigned short mbox_msg_qctxt::qid

queue id

3.15.2.9 struct hw_descq_context mbox_msg_qctxt::context

complete hw context

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

- [qdma_mbox.h](#)

3.16 qconf_entry Struct Reference

```
#include <qdma_qconf_mgr.h>
```

Data Fields

- struct list_head [list_head](#)
- u32 [idx](#)
- u32 [qbase](#)
- u32 [qmax](#)
- enum q_cfg_state [cfg_state](#)
- enum pci_dev_type [type](#)
- u8 [func_id](#)

3.16.1 Detailed Description

q configuration entry

3.16.2 Field Documentation

3.16.2.1 struct list_head qconf_entry::list_head

to connect to list head of

3.16.2.2 u32 qconf_entry::idx

idx of the device

3.16.2.3 u32 qconf_entry::qbase

qbase for func_id

3.16.2.4 u32 qconf_entry::qmax

qmax for func_id

3.16.2.5 enum q_cfg_state qconf_entry::cfg_state

current configuration state

3.16.2.6 enum pci_dev_type qconf_entry::type

device type PF/VF

3.16.2.7 u8 qconf_entry::func_id

func_id of the device

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

- qdma_qconf_mgr.h

3.17 qconf_entry_head Struct Reference

```
#include <qdma_qconf_mgr.h>
```

Data Fields

- struct list_head [vf_list](#)
- struct list_head [vf_free_list](#)
- struct list_head [pf_list](#)
- u32 [vf_qmax](#)
- u32 [pf_qmax](#)
- u32 [vf_qbase](#)
- atomic_t [vf_cnt](#)
- u32 [qcnt_cfgd_free](#)
- u32 [qcnt_init_free](#)
- u32 [qcnt_init_used](#)

3.17.1 Detailed Description

for holding the [qconf_entry](#) structure

3.17.2 Field Documentation

3.17.2.1 struct list_head qconf_entry_head::vf_list

for holding vf [qconf_entry](#)

3.17.2.2 struct list_head qconf_entry_head::vf_free_list

for holding vf [free_entry](#)

3.17.2.3 struct list_head qconf_entry_head::pf_list

for holding pf [qconf_entry](#)

3.17.2.4 u32 qconf_entry_head::vf_qmax

for maximum qs for vf

3.17.2.5 u32 qconf_entry_head::pf_qmax

for maximum qs for pf

3.17.2.6 u32 qconf_entry_head::vf_qbase

for holding vf [qconf_entry](#)

3.17.2.7 atomic_t qconf_entry_head::vf_cnt

number of vfs attached to all pfs

3.17.2.8 u32 qconf_entry_head::qcnt_cfgd_free

free count of qs which can be configured

3.17.2.9 u32 qconf_entry_head::qcnt_init_free

number of qs free for initial cfg devices

3.17.2.10 u32 qconf_entry_head::qcnt_init_used

used by INITIAL state devices

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

- qdma_qconf_mgr.h

3.18 qdma_c2h_cmpt_cmpl_status Struct Reference

```
#include <qdma_regs.h>
```

Data Fields

- __be16 [pidx](#)
- __be16 [cidx](#)
- __be32 [color_isr_status](#)

3.18.1 Detailed Description

qdma completion data descriptor

3.18.2 Field Documentation

3.18.2.1 __be16 qdma_c2h_cmpt_cmpl_status::pidx

producer index

3.18.2.2 __be16 qdma_c2h_cmpt_cmpl_status::cidx

consumer index

3.18.2.3 __be32 qdma_c2h_cmpt_cmpl_status::color_isr_status

isr color and status

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

- qdma_regs.h

3.19 qdma_c2h_desc Struct Reference

```
#include <qdma_regs.h>
```

Data Fields

- [__be64 dst_addr](#)

3.19.1 Detailed Description

qdma c2h descriptor

3.19.2 Field Documentation

3.19.2.1 [__be64 qdma_c2h_desc::dst_addr](#)

destination address

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

- [qdma_regs.h](#)

3.20 qdma_cdev Struct Reference

```
#include <cdev.h>
```

Data Fields

- struct list_head [list_head](#)
- int [minor](#)
- dev_t [cdevno](#)
- struct [qdma_cdev_cb](#) * [xcb](#)
- struct device * [sys_device](#)
- struct cdev [cdev](#)
- unsigned long [c2h_qhndl](#)
- unsigned long [h2c_qhndl](#)
- unsigned short [dir_init](#)
- unsigned char [no_memcpy](#)
- int(* [fp_open_extra](#))(struct [qdma_cdev](#) *)
- int(* [fp_close_extra](#))(struct [qdma_cdev](#) *)
- long(* [fp_ioctl_extra](#))(struct [qdma_cdev](#) *, unsigned int, unsigned long)
- ssize_t(* [fp_rw](#))(unsigned long dev_hndl, unsigned long qhndl, struct [qdma_request](#) *)
- ssize_t(* [fp_aiorw](#))(unsigned long dev_hndl, unsigned long qhndl, unsigned long count, struct [qdma_request](#) **)
- char [name](#) [0]

3.20.1 Detailed Description

QDMA character device book keeping parameters.

3.20.2 Field Documentation

3.20.2.1 struct list_head qdma_cdev::list_head

list of qdma character devices

3.20.2.2 int qdma_cdev::minor

minor number

3.20.2.3 dev_t qdma_cdev::cdevno

character device number

3.20.2.4 struct qdma_cdev_cb* qdma_cdev::xcb

pointer to qdma character device call back data

3.20.2.5 struct device* qdma_cdev::sys_device

pointer to kernel device(struct device)

3.20.2.6 struct cdev qdma_cdev::cdev

pointer to kernel cdev(struct cdev)

3.20.2.7 unsigned long qdma_cdev::c2h_qhndl

c2h queue handle

3.20.2.8 unsigned long qdma_cdev::h2c_qhndl

hec queue handle

3.20.2.9 unsigned short qdma_cdev::dir_init

direction

3.20.2.10 int(* qdma_cdev::fp_open_extra) (struct qdma_cdev *)

call back function for open a device

3.20.2.11 `int(* qdma_cdev::fp_close_extra) (struct qdma_cdev *)`

call back function for close a device

3.20.2.12 `long(* qdma_cdev::fp_ioctl_extra) (struct qdma_cdev *, unsigned int, unsigned long)`

call back function to handle ioctl message

3.20.2.13 `ssize_t(* qdma_cdev::fp_rw) (unsigned long dev_hdl, unsigned long qhdl, struct qdma_request *)`

call back function to handle read write request

3.20.2.14 `char qdma_cdev::name[0]`

name of the character device

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

- [cdev.h](#)

3.21 qdma_cdev_cb Struct Reference

```
#include <cdev.h>
```

Data Fields

- struct [xlnx_pci_dev](#) * [xpdev](#)
- [spinlock_t](#) [lock](#)
- int [cdev_major](#)
- int [cdev_minor_cnt](#)

3.21.1 Detailed Description

QDMA character device call back data.

3.21.2 Field Documentation

3.21.2.1 `struct xlnx_pci_dev* qdma_cdev_cb::xpdev`

pointer to xilinx pcie device

3.21.2.2 `spinlock_t qdma_cdev_cb::lock`

character device lock

3.21.2.3 `int qdma_cdev_cb::cdev_major`

character device major number

3.21.2.4 `int qdma_cdev_cb::cdev_minor_cnt`

character device minor number count

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

- [cdev.h](#)

3.22 `qdma_cmpl_ctrl` Struct Reference

```
#include <libqdma_export.h>
```

Data Fields

- `u8 cnt_th_idx`:4
- `u8 timer_idx`:4
- `u8 trigger_mode`:3
- `u8 en_stat_desc`:1
- `u8 cmpl_en_intr`:1

3.22.1 Detailed Description

packet/streaming interfaces struct [qdma_cmpl_ctrl](#) - completion control

3.22.2 Field Documentation

3.22.2.1 `u8 qdma_cmpl_ctrl::cnt_th_idx`

[global_csr_conf.c2h_cnt_th\[N\]](#)

3.22.2.2 `u8 qdma_cmpl_ctrl::timer_idx`

[global_csr_conf.c2h_timer_cnt\[N\]](#)

3.22.2.3 u8 qdma_cmpl_ctrl::trigger_mode

trigger_mode_t

3.22.2.4 u8 qdma_cmpl_ctrl::en_stat_desc

enable status desc. for CMPT

3.22.2.5 u8 qdma_cmpl_ctrl::cmpl_en_intr

enable interrupt for CMPT

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

- [libqdma_export.h](#)

3.23 qdma_csr_info Struct Reference

Data Fields

- enum [csr_type](#) **type**
- u32 [array](#) [[QDMA_GLOBAL_CSR_ARRAY_SZ](#)]
- u8 [idx_rngsz](#)
- u8 [idx_bufsz](#)
- u8 [idx_timer_cnt](#)
- u8 [idx_cnt_th](#)
- u32 [rngsz](#)
- u32 [bufsz](#)
- u32 [timer_cnt](#)
- u32 [cnt_th](#)
- u32 [cmpl_status_acc](#)

3.23.1 Field Documentation

3.23.1.1 u32 qdma_csr_info::array[QDMA_GLOBAL_CSR_ARRAY_SZ]

one csr register array

3.23.1.2 u8 qdma_csr_info::idx_bufsz

1x index-value pair for each type

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

- [qdma_device.h](#)

3.24 qdma_desc_cmpl_status Struct Reference

```
#include <qdma_regs.h>
```

Data Fields

- `__be16` [pidx](#)
- `__be16` [cidx](#)
- `__be32` [rsvd](#)

3.24.1 Detailed Description

qdma writeback descriptor

3.24.2 Field Documentation

3.24.2.1 `__be16` `qdma_desc_cmpl_status::pidx`

producer index

3.24.2.2 `__be16` `qdma_desc_cmpl_status::cidx`

consumer index

3.24.2.3 `__be32` `qdma_desc_cmpl_status::rsvd`

reserved 32 bits

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

- `qdma_regs.h`

3.25 qdma_descq Struct Reference

```
#include <qdma_descq.h>
```

Data Fields

- struct [qdma_queue_conf](#) conf
- [spinlock_t](#) lock
- struct [xlrx_dma_dev](#) * xdev
- u8 [channel](#)
- u8 [err](#):1
- u8 [color](#):1
- u8 [cpu_assigned](#):1
- u8 [proc_req_running](#)
- enum [q_state_t](#) q_state
- unsigned int [qidx_hw](#)
- unsigned int [intr_work_cpu](#)
- struct [work_struct](#) work
- struct [list_head](#) [intr_list](#)
- struct [list_head](#) [legacy_intr_q_list](#)
- int [intr_id](#)
- struct [list_head](#) [work_list](#)
- struct [qdma_kthread](#) * [cmplthp](#)
- struct [list_head](#) [cmplthp_list](#)
- struct [list_head](#) [pend_list](#)
- [qdma_wait_queue](#) [pend_list_wq](#)
- unsigned int [pend_list_empty](#)
- unsigned int [q_stop_wait](#)
- unsigned int [avail](#)
- unsigned int [io_batch_cnt](#)
- unsigned int [pend_req_desc](#)
- unsigned int [pidx](#)
- unsigned int [cidx](#)
- unsigned int [credit](#)
- u8 * [desc](#)
- [dma_addr_t](#) [desc_bus](#)
- u8 * [desc_cmpl_status](#)
- unsigned char [fl_pg_order](#)
- unsigned char [cmpt_entry_len](#)
- unsigned char [rsvd](#) [2]
- unsigned char [flq](#) [QDMA_FLQ_SIZE]
- unsigned int [udd_cnt](#)
- unsigned int [pkt_cnt](#)
- unsigned int [pkt_dlen](#)
- unsigned int [pidx_cmpt](#)
- unsigned int [cidx_cmpt](#)
- unsigned int [cidx_cmpt_pend](#)
- unsigned long long [total_cmpl_descs](#)
- void * [desc_cmpt_cur](#)
- u8 * [desc_cmpt](#)
- [dma_addr_t](#) [desc_cmpt_bus](#)
- u8 * [desc_cmpt_cmpl_status](#)

3.25.1 Detailed Description

qdma software descriptor book keeping fields

3.25.2 Field Documentation

3.25.2.1 struct qdma_queue_conf qdma_descq::conf

qdma queue configuration

3.25.2.2 spinlock_t qdma_descq::lock

lock to protect access to software descriptor

3.25.2.3 struct xlnx_dma_dev* qdma_descq::xdev

pointer to dma device

3.25.2.4 u8 qdma_descq::channel

number of channels

3.25.2.5 u8 qdma_descq::err

flag to indicate error on the Q, in halted state

3.25.2.6 u8 qdma_descq::color

color bit for the queue

3.25.2.7 u8 qdma_descq::cpu_assigned

cpu attached

3.25.2.8 u8 qdma_descq::proc_req_running

state of the proc req

3.25.2.9 enum q_state_t qdma_descq::q_state

Indicate q state

3.25.2.10 unsigned int qdma_descq::qid_x_hw

hw qidx associated for this queue

3.25.2.11 unsigned int qdma_descq::intr_work_cpu

cpu attached to intr_work

3.25.2.12 struct work_struct qdma_descq::work

queue handler

3.25.2.13 struct list_head qdma_descq::intr_list

interrupt list

3.25.2.14 struct list_head qdma_descq::legacy_intr_q_list

legacy interrupt list

3.25.2.15 int qdma_descq::intr_id

interrupt id associated for this queue

3.25.2.16 struct list_head qdma_descq::work_list

work list for the queue

3.25.2.17 struct qdma_kthread* qdma_descq::cmplthp

write back thread list

3.25.2.18 struct list_head qdma_descq::cmplthp_list

completion status thread list for the queue

3.25.2.19 struct list_head qdma_descq::pend_list

pending qork thread list

3.25.2.20 qdma_wait_queue qdma_descq::pend_list_wq

wait queue for pending list clear

3.25.2.21 unsigned int qdma_descq::pend_list_empty

pending list empty count

3.25.2.22 unsigned int qdma_descq::avail

availed count

3.25.2.23 unsigned int qdma_descq::io_batch_cnt

IO batching cunt

3.25.2.24 unsigned int qdma_descq::pend_req_desc

current req count

3.25.2.25 unsigned int qdma_descq::pidx

current producer index

3.25.2.26 unsigned int qdma_descq::cidx

current consumer index

3.25.2.27 unsigned int qdma_descq::credit

number of descrtors yet to be processed

3.25.2.28 u8* qdma_descq::desc

descrtor to be processed

3.25.2.29 dma_addr_t qdma_descq::desc_bus

descrtor dma address

3.25.2.30 u8* qdma_descq::desc_cmpl_status

descrtor writeback

3.25.2.31 unsigned char qdma_descq::fl_pg_order

programming order of the data in ST c2h mode

3.25.2.32 unsigned char qdma_descq::cmpt_entry_len

cmpt entry length

3.25.2.33 unsigned char qdma_descq::rsvd[2]

2 bits reserved

3.25.2.34 unsigned char qdma_descq::flq[QDMA_FLQ_SIZE]

qdma free list q

3.25.2.35 unsigned int qdma_descq::udd_cnt

total # of udd outstanding

3.25.2.36 unsigned int qdma_descq::pkt_cnt

packet count/number of packets to be processed

3.25.2.37 unsigned int qdma_descq::pkt_dlen

packet data length

3.25.2.38 unsigned int qdma_descq::pidx_cmpt

pidx of the completion entry

3.25.2.39 unsigned int qdma_descq::cidx_cmpt

completion cidx

3.25.2.40 unsigned int qdma_descq::cidx_cmpt_pend

pending writeback cidx

3.25.2.41 unsigned long long qdma_descq::total_cmpl_descs

number of packets processed in q

3.25.2.42 void* qdma_descq::desc_cmpt_cur

descriptor writeback, data type depends on the cmpt_entry_len

3.25.2.43 u8* qdma_descq::desc_cmpt

pointer to completion entry

3.25.2.44 dma_addr_t qdma_descq::desc_cmpt_bus

descriptor dma bus address

3.25.2.45 u8* qdma_descq::desc_cmpt_cmpl_status

descriptor writeback dma bus address

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

- [qdma_descq.h](#)

3.26 qdma_dev Struct Reference

```
#include <qdma_device.h>
```

Data Fields

- u8 [init_qrange](#):1
- u8 [filler](#) [3]
- unsigned short [qmax](#)
- unsigned short [qbase](#)
- spinlock_t [lock](#)
- unsigned short [h2c_qcnt](#)
- unsigned short [c2h_qcnt](#)
- struct [qdma_descq](#) * [h2c_descq](#)
- struct [qdma_descq](#) * [c2h_descq](#)

3.26.1 Detailed Description

qdma device per function

3.26.2 Field Documentation

3.26.2.1 u8 qdma_dev::init_qrange

flag indicates whether the fmap programming is completed or not

3.26.2.2 u8 qdma_dev::filler[3]

filler

3.26.2.3 unsigned short qdma_dev::qmax

max number of queues per function

3.26.2.4 unsigned short qdma_dev::qbase

queue start number for this function

3.26.2.5 spinlock_t qdma_dev::lock

[qdma_dev](#) lock

3.26.2.6 unsigned short qdma_dev::h2c_qcnt

number of h2c queues for this function

3.26.2.7 unsigned short qdma_dev::c2h_qcnt

number of c2h queues for this function

3.26.2.8 struct qdma_descq* qdma_dev::h2c_descq

h2c descq list

3.26.2.9 struct qdma_descq* qdma_dev::c2h_descq

c2h descq list

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

- [qdma_device.h](#)

3.27 qdma_dev_conf Struct Reference

```
#include <libqdma_export.h>
```

Data Fields

- struct pci_dev * [pdev](#)
- unsigned short [qsets_max](#)
- unsigned short [rsvd2](#)
- u8 [zerolen_dma](#):1
- u8 [master_pf](#):1
- u8 [isr_top_q_en](#):1
- u8 [rsvd1](#):5
- u8 [vf_max](#)
- u8 [intr_rngsz](#)
- u8 [msix_qvec_max](#)
- unsigned long [uld](#)
- enum [qdma_drv_mode](#) [qdma_drv_mode](#)
- char [name](#) [QDMA_DEV_NAME_MAXLEN]
- char [bar_num_config](#)
- char [bar_num_user](#)
- char [bar_num_bypass](#)
- char [bar_num_stm](#)
- unsigned int [qsets_base](#)
- u32 [bdf](#)
- u32 [idx](#)
- enum [qdma_dev_qmax_state](#) [cur_cfg_state](#)
- u8 [tm_mode_en](#)
- u8 [tm_one_cdh_en](#)
- void(* [fp_user_isr_handler](#))(unsigned long dev_hdl, unsigned long [uld](#))
- void(* [fp_q_isr_top_dev](#))(unsigned long dev_hdl, unsigned long [uld](#))

3.27.1 Detailed Description

[qdma_dev_conf](#) defines the per-device qdma property.

NOTE: if any of the max requested is less than supported, the value will be updated

3.27.2 Field Documentation

3.27.2.1 struct pci_dev* qdma_dev_conf::pdev

pointer to pci_dev

3.27.2.2 unsigned short qdma_dev_conf::qsets_max

Maximum number of queue pairs per device

3.27.2.3 unsigned short qdma_dev_conf::rsvd2

Reserved

3.27.2.4 u8 qdma_dev_conf::zerolen_dma

Indicates whether zero length DMA is allowed or not

3.27.2.5 u8 qdma_dev_conf::master_pf

Indicates whether the current pf is master_pf or not

3.27.2.6 u8 qdma_dev_conf::isr_top_q_en

extra handling of per descq handling in top half (i.e., qdma_descq.fp_descq_isr_top will be set)

3.27.2.7 u8 qdma_dev_conf::rsvd1

Reserved1

3.27.2.8 u8 qdma_dev_conf::vf_max

Maximum number of virtual functions for current physical function

3.27.2.9 u8 qdma_dev_conf::intr_rngsz

Interrupt ring size

3.27.2.10 u8 qdma_dev_conf::msix_qvec_max

interrupt:

- MSI-X only max of QDMA_DEV_MSIX_VEC_MAX per function, 32 in Everest
- 1 vector is reserved for user interrupt
- 1 vector is reserved mailbox
- 1 vector on pf0 is reserved for error interrupt
- the remaining vectors will be used for queuesmax. of vectors used for queues. libqdma update w/ actual #

3.27.2.11 unsigned long qdma_dev_conf::uld

upper layer data, i.e. callback data

3.27.2.12 enum qdma_drv_mode qdma_dev_conf::qdma_drv_mode

qdma driver mode

3.27.2.13 char qdma_dev_conf::name[QDMA_DEV_NAME_MAXLEN]

an unique string to identify the dev. current format: qdma[pf|vf][idx] filled in by libqdma

3.27.2.14 char qdma_dev_conf::bar_num_config

dma config bar #, < 0 not present

3.27.2.15 char qdma_dev_conf::bar_num_user

user bar

3.27.2.16 char qdma_dev_conf::bar_num_bypass

bypass bar

3.27.2.17 char qdma_dev_conf::bar_num_stm

STM bar, PF only

3.27.2.18 unsigned int qdma_dev_conf::qsets_base

user bar, PF only

3.27.2.19 u32 qdma_dev_conf::bdf

device index

3.27.2.20 u32 qdma_dev_conf::idx

index of device in device list

3.27.2.21 enum qdma_dev_qmax_state qdma_dev_conf::cur_cfg_state

current configuration state of device

3.27.2.22 u8 qdma_dev_conf::tm_mode_en

xmit in traffic manager mode

3.27.2.23 u8 qdma_dev_conf::tm_one_cdh_en

enable 1 CDH for Traffic Manager

3.27.2.24 void(* qdma_dev_conf::fp_user_isr_handler) (unsigned long dev_hndl, unsigned long uld)

user interrupt, if null, default libqdma handler is used

3.27.2.25 void(* qdma_dev_conf::fp_q_isr_top_dev) (unsigned long dev_hndl, unsigned long uld)

example flow of ST C2H: a. interrupt fires b. Hard IRQ: libqdma isr top -> dev->fp_q_isr_top_dev -> isr_top_qproc && Q->fp_descq_isr_top c. Soft IRQ: irq handler qdma_queue_service_bh() -> if rx: Q->fp_descq_rx_packet() called for each packet qdma_queue_cmpl_ctrl(set=true) to update h/w and re-enable interruptQ interrupt top, per-device additional handling code

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

- [libqdma_export.h](#)

3.28 qdma_flq Struct Reference

```
#include <qdma_st_c2h.h>
```

Data Fields

- unsigned int [size](#)
- unsigned char [pg_order](#)
- unsigned char [pg_shift](#)
- struct [qdma_c2h_desc](#) * [desc](#)
- unsigned int [udd_cnt](#)
- unsigned int [pkt_cnt](#)
- unsigned int [pkt_dlen](#)
- unsigned int [avail](#)
- unsigned long [alloc_fail](#)
- unsigned long [mapping_err](#)
- unsigned int [cidx](#)
- unsigned int [pidx](#)
- unsigned int [pidx_pend](#)
- struct [qdma_sw_sg](#) * [sdesc](#)
- struct [qdma_sdesc_info](#) * [sdesc_info](#)

3.28.1 Detailed Description

qdma free list q page allocation book keeping

3.28.2 Field Documentation

3.28.2.1 unsigned int qdma_flq::size

RO: size of the decriptor

3.28.2.2 unsigned char qdma_flq::pg_order

RO: page order

3.28.2.3 unsigned char qdma_flq::pg_shift

RO: page shift

3.28.2.4 struct qdma_c2h_desc* qdma_flq::desc

RO: pointer to qdma c2h decriptor

3.28.2.5 unsigned int qdma_flq::udd_cnt

RW: total # of udd outstanding

3.28.2.6 unsigned int qdma_flq::pkt_cnt

RW: total # of packet outstanding

3.28.2.7 unsigned int qdma_flq::pkt_dlen

RW: total # of pkt payload length outstanding

3.28.2.8 unsigned int qdma_flq::avail

RW: # of available Rx buffers

3.28.2.9 unsigned long qdma_flq::alloc_fail

RW: # of times buffer allocation failed

3.28.2.10 unsigned long qdma_flq::mapping_err

RW: # of RX Buffer DMA Mapping failures

3.28.2.11 unsigned int qdma_flq::cidx

RW: consumer index

3.28.2.12 unsigned int qdma_flq::pidx

RW: producer index

3.28.2.13 unsigned int qdma_flq::pidx_pend

RW: pending pidxes

3.28.2.14 struct qdma_sw_sg* qdma_flq::sdesc

RW: sw scatter gather list

3.28.2.15 struct qdma_sdesc_info* qdma_flq::sdesc_info

RW: sw descriptor info

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

- [qdma_st_c2h.h](#)

3.29 qdma_h2c_desc Struct Reference

```
#include <qdma_regs.h>
```

Data Fields

- [__be16 cdh_flags](#)
- [__be16 pld_len](#)
- [__be16 len](#)
- [__be16 flags](#)
- [__be64 src_addr](#)

3.29.1 Detailed Description

memory mapped descriptor format

3.29.2 Field Documentation

3.29.2.1 `__be16 qdma_h2c_desc::cdh_flags`

cdh flags

3.29.2.2 `__be16 qdma_h2c_desc::pld_len`

current packet length

3.29.2.3 `__be16 qdma_h2c_desc::len`

total packet length

3.29.2.4 `__be16 qdma_h2c_desc::flags`

descriptor flags

3.29.2.5 `__be64 qdma_h2c_desc::src_addr`

source address

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

- `qdma_regs.h`

3.30 `qdma_intr_ring` Struct Reference

```
#include <qdma_intr.h>
```

Data Fields

- `__be64 pidx:16`
- `__be64 cidx:16`
- `__be64 s_color:1`
- `__be64 intr_satus:2`
- `__be64 error:2`
- `__be64 rsvd:1`
- `__be64 intr_type:1`
- `__be64 qid:24`
- `__be64 coal_color:1`

3.30.1 Detailed Description

Interrupt ring entry definition.

3.30.2 Field Documentation

3.30.2.1 __be64 qdma_intr_ring::pidx

producer index. This is from Interrupt source. Cumulative pointer of total interrupt Aggregation Ring entry written

3.30.2.2 __be64 qdma_intr_ring::cidx

consumer index. This is from Interrupt source. Cumulative consumed pointer

3.30.2.3 __be64 qdma_intr_ring::s_color

source color. This is from Interrupt source. This bit inverts every time pidx wraps around and this field gets copied to color field of descriptor.

3.30.2.4 __be64 qdma_intr_ring::intr_satus

This is from Interrupt source. Interrupt state, 0: CMPT_INT_ISR; 1: CMPT_INT_TRIG; 2: CMPT_INT_ARMED

3.30.2.5 __be64 qdma_intr_ring::error

error. This is from interrupt source {C2h_err[1:0], h2c_err[1:0]}

3.30.2.6 __be64 qdma_intr_ring::rsvd

1 reserved bits

3.30.2.7 __be64 qdma_intr_ring::intr_type

interrupt type, 0: H2C; 1: C2H

3.30.2.8 __be64 qdma_intr_ring::qid

This is from Interrupt source. Queue ID

3.30.2.9 __be64 qdma_intr_ring::coal_color

The color bit of the Interrupt Aggregation Ring. This bit inverts every time pidx wraps around on the Interrupt Aggregation Ring.

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

- [qdma_intr.h](#)

3.31 qdma_io_cb Struct Reference

```
#include <cdev.h>
```

Data Fields

- void * **private**
- void __user * [buf](#)
- size_t [len](#)
- unsigned int [pages_nr](#)
- struct [qdma_sw_sg](#) * [sgl](#)
- struct page ** [pages](#)
- struct [qdma_request](#) [req](#)

3.31.1 Detailed Description

QDMA character device io call back book keeping parameters.

3.31.2 Field Documentation

3.31.2.1 void __user* qdma_io_cb::buf

user buffer

3.31.2.2 size_t qdma_io_cb::len

length of the user buffer

3.31.2.3 unsigned int qdma_io_cb::pages_nr

page number

3.31.2.4 struct qdma_sw_sg* qdma_io_cb::sgl

scatter gather list

3.31.2.5 struct page** qdma_io_cb::pages

pages allocated to accommodate the scatter gather list

3.31.2.6 struct qdma_request qdma_io_cb::req

qdma request

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

- [cdev.h](#)

3.32 qdma_kthread Struct Reference

```
#include <thread.h>
```

Data Fields

- `spinlock_t` [lock](#)
- `char` [name](#) [16]
- `unsigned short` [cpu](#)
- `unsigned short` [id](#)
- `unsigned int` [timeout](#)
- `unsigned long` [flag](#)
- `qdma_wait_queue` [waitq](#)
- `unsigned int` [schedule](#)
- `struct task_struct *` [task](#)
- `unsigned int` [work_cnt](#)
- `struct list_head` [work_list](#)
- `int(* finit)(struct qdma_kthread *)`
- `int(* fpending)(struct list_head *)`
- `int(* fproc)(struct list_head *)`
- `int(* fdone)(struct qdma_kthread *)`

3.32.1 Detailed Description

qdma thread book keeping parameters

3.32.2 Field Documentation

3.32.2.1 `spinlock_t` qdma_kthread::lock

thread lock

3.32.2.2 `char qdma_kthread::name[16]`

name of the thread

3.32.2.3 `unsigned short qdma_kthread::cpu`

cpu number for which the thread associated with

3.32.2.4 `unsigned short qdma_kthread::id`

thread id

3.32.2.5 `unsigned int qdma_kthread::timeout`

thread sleep timeout value

3.32.2.6 `unsigned long qdma_kthread::flag`

flags for thread

3.32.2.7 `qdma_wait_queue qdma_kthread::waitq`

thread wait queue

3.32.2.8 `struct task_struct* qdma_kthread::task`

kernel task structure associated with thread

3.32.2.9 `unsigned int qdma_kthread::work_cnt`

thread work list count

3.32.2.10 `struct list_head qdma_kthread::work_list`

thread work list count

3.32.2.11 `int(* qdma_kthread::finit)(struct qdma_kthread *)`

thread initialization handler

3.32.2.12 `int(* qdma_kthread::fpending) (struct list_head *)`

thread pending handler

3.32.2.13 `int(* qdma_kthread::fproc) (struct list_head *)`

thread peocessing handler

3.32.2.14 `int(* qdma_kthread::fdone) (struct qdma_kthread *)`

thread done handler

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

- [thread.h](#)

3.33 qdma_mbox Struct Reference

Data Fields

- `spinlock_t lock`
- `spinlock_t hw_tx_lock`
- `spinlock_t hw_rx_lock`
- `struct workqueue_struct * workq`
- `struct xlnx_dma_dev * xdev`
- `struct work_struct tx_work`
- `struct work_struct rx_work`
- `struct mbox_msg rx`
- `spinlock_t list_lock`
- `struct list_head tx_todo_list`
- `struct list_head rx_pend_list`
- `struct timer_list timer`

3.33.1 Detailed Description

mailbox book keeping structure

3.33.2 Field Documentation

3.33.2.1 `spinlock_t qdma_mbox::lock`

common lock

3.33.2.2 `spinlock_t qdma_mbox::hw_tx_lock`

tx lock

3.33.2.3 `spinlock_t qdma_mbox::hw_rx_lock`

rx lock

3.33.2.4 `struct workqueue_struct* qdma_mbox::workq`

work queue

3.33.2.5 `struct xlnx_dma_dev* qdma_mbox::xdev`

pointer to device data

3.33.2.6 `struct work_struct qdma_mbox::tx_work`

tx work_struct to pass data to tx work queue

3.33.2.7 `struct work_struct qdma_mbox::rx_work`

rx work_struct to pass data to rx work queue

3.33.2.8 `struct mbox_msg qdma_mbox::rx`

mbox rx message

3.33.2.9 `spinlock_t qdma_mbox::list_lock`

list lock

3.33.2.10 `struct list_head qdma_mbox::tx_todo_list`

list of messages waiting to be sent

3.33.2.11 `struct list_head qdma_mbox::rx_pend_list`

list of messages waiting for response

3.33.2.12 struct timer_list qdma_mbox::timer

timer list

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

- [qdma_mbox.h](#)

3.34 qdma_mm_desc Struct Reference

```
#include <qdma_regs.h>
```

Data Fields

- [__be64 src_addr](#)
- [__be32 flag_len](#)
- [__be32 rsvd0](#)
- [__be64 dst_addr](#)
- [__be64 rsvd1](#)

3.34.1 Detailed Description

memory mapped descriptor format

3.34.2 Field Documentation

3.34.2.1 [__be64 qdma_mm_desc::src_addr](#)

source address

3.34.2.2 [__be32 qdma_mm_desc::flag_len](#)

flags

3.34.2.3 [__be32 qdma_mm_desc::rsvd0](#)

reserved 32 bits

3.34.2.4 [__be64 qdma_mm_desc::dst_addr](#)

destination address

3.34.2.5 __be64 qdma_mm_desc::rsvd1

reserved 64 bits

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

- qdma_regs.h

3.35 qdma_queue_conf Struct Reference

```
#include <libqdma_export.h>
```

Data Fields

- u32 [qid](#):24
- u32 [st](#):1
- u32 [c2h](#):1
- u32 [pipe](#):1
- u32 [irq_en](#):1
- u32 [desc_rng_sz_idx](#):4
- u8 [cmpl_status_en](#):1
- u8 [cmpl_status_acc_en](#):1
- u8 [cmpl_status_pend_chk](#):1
- u8 [desc_bypass](#):1
- u8 [pfetch_en](#):1
- u8 [fetch_credit](#):1
- u8 [st_pkt_mode](#):1
- u8 [c2h_use_fl](#):1
- u8 [c2h_buf_sz_idx](#):4
- u8 [cmpl_rng_sz_idx](#):4
- u8 [cmpl_desc_sz](#):2
- u8 [cmpl_stat_en](#):1
- u8 [cmpl_udd_en](#):1
- u8 [cmpl_timer_idx](#):4
- u8 [cmpl_cnt_th_idx](#):4
- u8 [cmpl_trig_mode](#):3
- u8 [cmpl_en_intr](#):1
- u8 [sw_desc_sz](#):2
- u8 [pfetch_bypass](#):1
- u8 [cmpl_ovf_chk_dis](#):1
- u8 [port_id](#):3
- u8 [at](#):1
- u8 [cdh_max](#)
- u8 [pipe_gl_max](#)
- u8 [pipe_flow_id](#)
- u8 [pipe_slr_id](#)
- u16 [pipe_tdest](#)
- unsigned long [quld](#)
- void(* [fp_descq_isr_top](#))(unsigned long qhndl, unsigned long [quld](#))
- int(* [fp_descq_c2h_packet](#))(unsigned long qhndl, unsigned long [quld](#), unsigned int len, unsigned int sgcnt, struct [qdma_sw_sg](#) *sgl, void *udd)
- char [name](#) [QDMA_QUEUE_NAME_MAXLEN]
- unsigned int [rngsz](#)
- unsigned int [rngsz_cmpt](#)
- unsigned int [c2h_bufsz](#)

3.35.1 Detailed Description

struct [qdma_queue_conf](#) - qdma configuration parameters [qdma_queue_conf](#) defines the per-dma Q property. if any of the max requested is less than supported, the value will be updated

3.35.2 Field Documentation

3.35.2.1 u32 qdma_queue_conf::qidx

0xFFFF: libqdma choose the queue idx 0 ~ ([qdma_dev_conf.qsets_max](#) - 1) the calling function choose the queue idx

3.35.2.2 u32 qdma_queue_conf::st

config flags: byte #1 st mode

3.35.2.3 u32 qdma_queue_conf::c2h

c2h direction

3.35.2.4 u32 qdma_queue_conf::pipe

SDx only: inter-kernel communication pipe

3.35.2.5 u32 qdma_queue_conf::irq_en

poll or interrupt

3.35.2.6 u32 qdma_queue_conf::desc_rng_sz_idx

descriptor ring [global_csr_conf.ringsz\[N\]](#)

3.35.2.7 u8 qdma_queue_conf::cmpl_status_en

config flags: byte #2 writeback enable, disabled for ST C2H

3.35.2.8 u8 qdma_queue_conf::cmpl_status_acc_en

sw context.cmpl_status_acc_en

3.35.2.9 u8 qdma_queue_conf::cmpl_status_pend_chk

sw context.cmpl_stauts_pend_chk

3.35.2.10 u8 qdma_queue_conf::desc_bypass

send descriptor to bypass out

3.35.2.11 u8 qdma_queue_conf::pfetch_en

descriptor prefetch enable control

3.35.2.12 u8 qdma_queue_conf::fetch_credit

sw context.frcd_en[32]

3.35.2.13 u8 qdma_queue_conf::st_pkt_mode

SDx only: ST packet mode (i.e., with TLAST to identify the packet boundary)

3.35.2.14 u8 qdma_queue_conf::c2h_use_fl

c2h use pre-alloc free list

3.35.2.15 u8 qdma_queue_conf::c2h_buf_sz_idx

config flags: byte #3 [global_csr_conf.c2h_buf_sz\[N\]](#)

3.35.2.16 u8 qdma_queue_conf::cmpl_rng_sz_idx

ST C2H Completion/Writeback ring [global_csr_conf.ringsz\[N\]](#)

3.35.2.17 u8 qdma_queue_conf::cmpl_desc_sz

config flags: byte #4 C2H ST cmpt + immediate data, desc_sz_t

3.35.2.18 u8 qdma_queue_conf::cmpl_stat_en

enable status desc. for CMPT

3.35.2.19 u8 qdma_queue_conf::cmpl_udd_en

C2H Completion entry user-defined data

3.35.2.20 u8 qdma_queue_conf::cmpl_timer_idx

[global_csr_conf.c2h_timer_cnt\[N\]](#)

3.35.2.21 u8 qdma_queue_conf::cmpl_cnt_th_idx

config flags: byte #5 [global_csr_conf.c2h_cnt_th\[N\]](#)

3.35.2.22 u8 qdma_queue_conf::cmpl_trig_mode

tigger_mode_t

3.35.2.23 u8 qdma_queue_conf::cmpl_en_intr

enable interrupt for CMPT

3.35.2.24 u8 qdma_queue_conf::sw_desc_sz

config flags: byte #6 SW Context desc size, desc_sz_t

3.35.2.25 u8 qdma_queue_conf::pfetch_bypass

prefetch bypass en

3.35.2.26 u8 qdma_queue_conf::cmpl_ovf_chk_dis

OVF_DIS C2H ST over flow disable

3.35.2.27 u8 qdma_queue_conf::port_id

Port ID

3.35.2.28 u8 qdma_queue_conf::at

Address Translation

3.35.2.29 u8 qdma_queue_conf::cdh_max

only if pipe = 1 max 16. CDH length per packet

3.35.2.30 u8 qdma_queue_conf::pipe_gl_max

<= 7, max # gather buf. per packet

3.35.2.31 u8 qdma_queue_conf::pipe_flow_id

pipe flow id

3.35.2.32 u8 qdma_queue_conf::pipe_slr_id

pipe SLR id

3.35.2.33 u16 qdma_queue_conf::pipe_tdest

pipe route id

3.35.2.34 unsigned long qdma_queue_conf::quld

user provided per-Q irq handler

3.35.2.35 void(* qdma_queue_conf::fp_descq_isr_top) (unsigned long qhndl, unsigned long quld)

TBA: Q interrupt top, per-queue additional handling code for example, network rx: napi_schedule(&Q->napi)

3.35.2.36 int(* qdma_queue_conf::fp_descq_c2h_packet) (unsigned long qhndl, unsigned long quld, unsigned int len, unsigned int sgcnt, struct qdma_sw_sg *sgl, void *udd)

optional rx packet handler: called from irq BH (i.e. qdma_queue_service_bh())

- udd: user defined data in the completion entry
- sgcnt / sgl: packet data in scatter-gather list NOTE: a. do NOT modify any field of sgl b. if zero copy, do a get_page() to prevent page freeing c. do loop through the sgl with sg->next and stop at sgcnt. the last sg may not have sg->next = NULL Returns:
 - 0 to allow libqdma free/re-task the sgl
 - < 0, libqdma will keep the packet for processing again

A single packet could contain: in the case of c2h_udd_en = 1:

- udd only (udd valid, sgcnt = 0, sgl = NULL), or
- udd + packet data in the case of c2h_udd_en = 0:
- packet data (udd = NULL, sgcnt > 0 and sgl valid)

3.35.2.37 `char qdma_queue_conf::name[QDMA_QUEUE_NAME_MAXLEN]`

fill in by libqdma name of the qdma device

3.35.2.38 `unsigned int qdma_queue_conf::rngsz`

ring size of the queue

3.35.2.39 `unsigned int qdma_queue_conf::rngsz_cmpt`

completion ring size of the queue

3.35.2.40 `unsigned int qdma_queue_conf::c2h_bufsz`

C2H buffer size

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

- [libqdma_export.h](#)

3.36 qdma_request Struct Reference

```
#include <libqdma_export.h>
```

Data Fields

- unsigned char [opaque](#) [[QDMA_REQ_OPAQUE_SIZE](#)]
- unsigned long [uld_data](#)
- int(* [fp_done](#))(struct [qdma_request](#) *, unsigned int bytes_done, int err)
- unsigned int [timeout_ms](#)
- unsigned int [count](#)
- u64 [ep_addr](#)
- u8 [no_memcpy](#):1
- u8 [write](#):1
- u8 [dma_mapped](#):1
- u8 [h2c_eot](#):1
- u8 [udd_len](#)
- unsigned int [sgcnt](#)
- struct [qdma_sw_sg](#) * [sgl](#)
- u8 [udd](#) [[QDMA_UDD_MAXLEN](#)]

3.36.1 Detailed Description

struct [qdma_request](#) - qdma request for read or write

3.36.2 Field Documentation

3.36.2.1 unsigned char qdma_request::opaque[QDMA_REQ_OPAQUE_SIZE]

private to the dma driver, do NOT touch

3.36.2.2 unsigned long qdma_request::uld_data

filled in by the calling function for the calling function

3.36.2.3 int(* qdma_request::fp_done) (struct qdma_request *, unsigned int bytes_done, int err)

set fp_done for non-blocking mode

3.36.2.4 unsigned int qdma_request::timeout_ms

timeout in mili-seconds, 0 - no timeout

3.36.2.5 unsigned int qdma_request::count

total data size

3.36.2.6 u64 qdma_request::ep_addr

MM only, DDR/BRAM memory addr

3.36.2.7 u8 qdma_request::write

write: if write to the device

3.36.2.8 u8 qdma_request::dma_mapped

if sgt is already dma mapped

3.36.2.9 u8 qdma_request::h2c_eot

user defined data present

3.36.2.10 u8 qdma_request::udd_len

indicates end of transfer towards user kernel

3.36.2.11 unsigned int qdma_request::sgcnt

of scatter-gather entries < 64K

3.36.2.12 struct qdma_sw_sg* qdma_request::sgl

scatter-gather list of data bufs

3.36.2.13 u8 qdma_request::udd[QDMA_UDD_MAXLEN]

udd data

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

- [libqdma_export.h](#)

3.37 qdma_sdesc_info Struct Reference

```
#include <qdma_st_c2h.h>
```

Data Fields

- struct [qdma_sdesc_info](#) * next
- union {
 - u8 fbits
 - struct **flags** {
 - u8 valid:1
 - u8 sop:1
 - u8 eop:1
 - u8 filler:5
 - } f
- };
- u8 rsvd [3]
- unsigned int cidx

3.37.1 Detailed Description

qdma descriptor information

3.37.2 Field Documentation

3.37.2.1 struct qdma_sdesc_info* qdma_sdesc_info::next

pointer to next descriptor

3.37.2.2 u8 qdma_sdesc_info::fbits

8 flag bits

3.37.2.3 u8 qdma_sdesc_info::valid

is descriptor valid

3.37.2.4 u8 qdma_sdesc_info::sop

start of the packet

3.37.2.5 u8 qdma_sdesc_info::eop

end of the packet

3.37.2.6 u8 qdma_sdesc_info::filler

filler for 5 bits

3.37.2.7 u8 qdma_sdesc_info::rsvd[3]

reserved 3 bits

3.37.2.8 unsigned int qdma_sdesc_info::cidx

consumer index

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

- [qdma_st_c2h.h](#)

3.38 qdma_sgt_req_cb Struct Reference

```
#include <qdma_descq.h>
```

Data Fields

- struct list_head [list](#)
- [qdma_wait_queue](#) [wq](#)
- unsigned int [desc_nr](#)
- unsigned int [offset](#)
- unsigned int [left](#)
- unsigned int [sg_offset](#)
- unsigned int [sg_idx](#)
- int [status](#)
- u8 [done](#)
- u8 [unmap_needed](#):1
- enum qdma_req_state [req_state](#)

3.38.1 Detailed Description

[qdma_sgt_req_cb](#) fits in [qdma_request.opaque](#)

3.38.2 Field Documentation

3.38.2.1 struct list_head qdma_sgt_req_cb::list

qdma read/write request list

3.38.2.2 qdma_wait_queue qdma_sgt_req_cb::wq

request wait queue

3.38.2.3 unsigned int qdma_sgt_req_cb::desc_nr

number of descriptors to proccess

3.38.2.4 unsigned int qdma_sgt_req_cb::offset

offset in the page

3.38.2.5 unsigned int qdma_sgt_req_cb::left

number of data byte not yet proccessed

3.38.2.6 unsigned int qdma_sgt_req_cb::sg_offset

offset in the scatter gather list

3.38.2.7 unsigned int qdma_sgt_req_cb::sg_idx

scatter gather ebtry index

3.38.2.8 int qdma_sgt_req_cb::status

status of the request

3.38.2.9 u8 qdma_sgt_req_cb::done

indicates whether request processing is done or not

3.38.2.10 u8 qdma_sgt_req_cb::unmap_needed

indicates whether to unmap the kernel pages

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

- [qdma_descq.h](#)

3.39 qdma_sw_sg Struct Reference

```
#include <libqdma_export.h>
```

Data Fields

- struct [qdma_sw_sg](#) * [next](#)
- struct page * [pg](#)
- unsigned int [offset](#)
- unsigned int [len](#)
- dma_addr_t [dma_addr](#)

3.39.1 Detailed Description

struct [qdma_sw_sg](#) - qdma scatter gather request

3.39.2 Field Documentation

3.39.2.1 struct qdma_sw_sg* qdma_sw_sg::next

pointer to next page

3.39.2.2 struct page* qdma_sw_sg::pg

pointer to current page

3.39.2.3 unsigned int qdma_sw_sg::offset

offset in current page

3.39.2.4 unsigned int qdma_sw_sg::len

length of the page

3.39.2.5 dma_addr_t qdma_sw_sg::dma_addr

dma address of the allocated page

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

- [libqdma_export.h](#)

3.40 qdma_version_info Struct Reference

Data Fields

- u8 **rtl_version**
- char **rtl_version_str** [[DEVICE_VERSION_INFO_STR_LENGTH](#)]
- u8 **vivado_release_id**
- char **vivado_release_str** [[DEVICE_VERSION_INFO_STR_LENGTH](#)]
- u8 **everest_ip**
- char **everest_ip_str** [[DEVICE_VERSION_INFO_STR_LENGTH](#)]

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

- [libqdma_export.h](#)

3.41 stm_descq_context Struct Reference

```
#include <qdma_mbox.h>
```

Data Fields

- u32 **stm** [6]

3.41.1 Detailed Description

queue stm information

3.41.2 Field Documentation

3.41.2.1 u32 stm_descq_context::stm[6]

STM data: 6 data words

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

- [qdma_mbox.h](#)

3.42 xlnx_dma_dev Struct Reference

```
#include <xdev.h>
```

Data Fields

- char [mod_name](#) [QDMA_DEV_NAME_MAXLEN]
- struct [qdma_dev_conf](#) [conf](#)
- struct list_head [list_head](#)
- spinlock_t [lock](#)
- spinlock_t [hw_prg_lock](#)
- unsigned int [flags](#)
- u8 [flr_prsnt](#):1
- u8 [st_mode_en](#):1
- u8 [mm_mode_en](#):1
- u8 [stm_en](#):1
- void * [vf_info](#)
- u8 [vf_count](#)
- u16 [func_id](#)
- u8 [pf_count](#)
- u8 [mm_channel_max](#)
- u8 [stm_rev](#)
- void __iomem * [regs](#)
- void __iomem * [stm_regs](#)
- int [num_vecs](#)
- struct msix_entry * [msix](#)
- struct [intr_info_t](#) * [dev_intr_info_list](#)
- int [dvec_start_idx](#)
- void * [dev_priv](#)
- u32 [pipe_stm_max_pkt_size](#)
- struct [intr_coal_conf](#) * [intr_coal_list](#)
- int [vector_legacy](#)
- struct [qdma_mbox](#) [mbox](#)
- unsigned long long [total_mm_h2c_pkts](#)
- unsigned long long [total_mm_c2h_pkts](#)
- unsigned long long [total_st_h2c_pkts](#)
- unsigned long long [total_st_c2h_pkts](#)
- unsigned int [dev_ulf_extra](#) [0]

3.42.1 Detailed Description

Xilinx DMA device details.

3.42.2 Field Documentation

3.42.2.1 `char xlnx_dma_dev::mod_name[QDMA_DEV_NAME_MAXLEN]`

< Xilinx DMA device name DMA device configuration

3.42.2.2 `struct qdma_dev_conf xlnx_dma_dev::conf`

DMA device list

3.42.2.3 `struct list_head xlnx_dma_dev::list_head`

DMA device lock to protects concurrent access

3.42.2.4 `spinlock_t xlnx_dma_dev::lock`

DMA device hardware program lock

3.42.2.5 `spinlock_t xlnx_dma_dev::hw_prg_lock`

device flags

3.42.2.6 `u8 xlnx_dma_dev::flr_prsnt`

flag to indicate the FLR present status

3.42.2.7 `u8 xlnx_dma_dev::st_mode_en`

flag to indicate the streaming mode enabled status

3.42.2.8 `u8 xlnx_dma_dev::mm_mode_en`

flag to indicate the memory mapped mode enabled status

3.42.2.9 `u8 xlnx_dma_dev::stm_en`

flag to indicate the presence of STM sriov info

3.42.2.10 void* xlnx_dma_dev::vf_info

number of virtual functions

3.42.2.11 u8 xlnx_dma_dev::vf_count

function id

3.42.2.12 u16 xlnx_dma_dev::func_id

number of physical functions

3.42.2.13 u8 xlnx_dma_dev::pf_count

max mm channels

3.42.2.14 u8 xlnx_dma_dev::stm_rev

PCIe config. bar

3.42.2.15 void __iomem* xlnx_dma_dev::stm_regs

PCIe Bar for STM config number of MSI-X interrupt vectors per device

3.42.2.16 int xlnx_dma_dev::num_vecs

msix_entry list for all MSIx vectors associated for device

3.42.2.17 struct msix_entry* xlnx_dma_dev::msix

interrupt info list for all MSIx vectors associated for device

3.42.2.18 struct intr_info_t* xlnx_dma_dev::dev_intr_info_list

data vector start index

3.42.2.19 int xlnx_dma_dev::dvec_start_idx

DMA private device to hold the qdma que details

3.42.2.20 `void* xlnx_dma_dev::dev_priv`

dsa configured max pkt size that STM can support

3.42.2.21 `u32 xlnx_dma_dev::pipe_stm_max_pkt_size`

list of interrupt coalescing configuration for each vector

3.42.2.22 `struct intr_coal_conf* xlnx_dma_dev::intr_coal_list`

legacy interrupt vector

3.42.2.23 `struct qdma_mbox xlnx_dma_dev::mbox`

number of packets processed in pf

3.42.2.24 `unsigned long long xlnx_dma_dev::total_st_c2h_pkts`

for upper layer calling function

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

- [xdev.h](#)

3.43 xlnx_nl_work Struct Reference

Data Fields

- struct work_struct **work**
- struct genl_info **nl_info**
- struct [xlnx_pci_dev](#) * **xpdev**
- wait_queue_head_t **wq**
- spinlock_t **lock**
- unsigned int **q_start_handled**
- union {
 - struct [xlnx_nl_work_q_ctrl](#) **qctrl**

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

- [qdma_mod.h](#)

3.44 xlnx_nl_work_q_ctrl Struct Reference

Data Fields

- unsigned short **qidx**
- unsigned short **qcnt**
- u8 **is_qp**:1
- u8 **is_c2h**:1

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

- [qdma_mod.h](#)

3.45 xlnx_pci_dev Struct Reference

```
#include <qdma_mod.h>
```

Data Fields

- struct list_head [list_head](#)
- struct pci_dev * [pdev](#)
- unsigned long [dev_hndl](#)
- struct workqueue_struct * [nl_task_wq](#)
- struct [qdma_cdev_cb](#) [cdev_cb](#)
- spinlock_t [cdev_lock](#)
- unsigned int [qmax](#)
- unsigned int [idx](#)
- void __iomem * [user_bar_regs](#)
- void __iomem * [bypass_bar_regs](#)
- struct [xlnx_qdata](#) [qdata](#) [0]

3.45.1 Detailed Description

xilinx pcie device data members

3.45.2 Field Documentation

3.45.2.1 struct list_head xlnx_pci_dev::list_head

device list

3.45.2.2 struct pci_dev* xlnx_pci_dev::pdev

pointer to struct pci_dev

3.45.2.3 unsigned long xlnx_pci_dev::dev_hdl

device handle

3.45.2.4 struct workqueue_struct* xlnx_pci_dev::nl_task_wq

netlink request work queue

3.45.2.5 struct qdma_cdev_cb xlnx_pci_dev::cdev_cb

character device call back data

3.45.2.6 spinlock_t xlnx_pci_dev::cdev_lock

character device lock

3.45.2.7 unsigned int xlnx_pci_dev::qmax

max number of queues for device

3.45.2.8 unsigned int xlnx_pci_dev::idx

device index

3.45.2.9 void __iomem* xlnx_pci_dev::user_bar_regs

PCIe user bar

3.45.2.10 void __iomem* xlnx_pci_dev::bypass_bar_regs

PCIe bypass bar

3.45.2.11 struct xlnx_qdata xlnx_pci_dev::qdata[0]

queue data

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

- [qdma_mod.h](#)

3.46 xlnx_qdata Struct Reference

```
#include <qdma_mod.h>
```

Data Fields

- unsigned long [qhndl](#)
- struct [qdma_cdev](#) * [xcdev](#)

3.46.1 Detailed Description

queue data variables send while read/write request

3.46.2 Field Documentation

3.46.2.1 unsigned long xlnx_qdata::qhndl

Queue handle

3.46.2.2 struct qdma_cdev* xlnx_qdata::xcdev

qdma character device details

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

- [qdma_mod.h](#)

3.47 xreg_info Struct Reference

Data Fields

- const char **name** [32]
- uint32_t **addr**
- unsigned int **repeat**
- unsigned int **step**
- unsigned char **shift**
- unsigned char **len**
- unsigned char **filler** [2]

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

- [xdev_regs.h](#)

Chapter 4

File Documentation

4.1 cdev.h File Reference

```
#include <linux/cdev.h>
#include "version.h"
#include <linux/spinlock_types.h>
#include "libqdma/libqdma_export.h"
#include <linux/workqueue.h>
```

Data Structures

- struct [qdma_cdev_cb](#)
- struct [qdma_cdev](#)
- struct [qdma_io_cb](#)

Macros

- #define [QDMA_CDEV_CLASS_NAME](#) DRV_MODULE_NAME
- #define [QDMA_MINOR_MAX](#) (2048)

Functions

- void [qdma_cdev_destroy](#) (struct [qdma_cdev](#) *xcdev)
- int [qdma_cdev_create](#) (struct [qdma_cdev_cb](#) *xcb, struct pci_dev *pdev, struct [qdma_queue_conf](#) *qconf, unsigned int minor, unsigned long qhndl, struct [qdma_cdev](#) **xcdev_pp, char *ebuf, int ebuflen)
- void [qdma_cdev_device_cleanup](#) (struct [qdma_cdev_cb](#) *xcb)
- int [qdma_cdev_device_init](#) (struct [qdma_cdev_cb](#) *xcb)
- void [qdma_cdev_cleanup](#) (void)
- int [qdma_cdev_init](#) (void)

4.1.1 Detailed Description

This file contains the declarations for qdma pcie kernel module.

4.1.2 Macro Definition Documentation

4.1.2.1 #define QDMA_CDEV_CLASS_NAME DRV_MODULE_NAME

QDMA character device class name

4.1.2.2 #define QDMA_MINOR_MAX (2048)

QDMA character device max minor number

4.1.3 Function Documentation

4.1.3.1 void qdma_cdev_destroy (struct qdma_cdev * xcdev)

[qdma_cdev_destroy\(\)](#) - handler to destroy the character device

Parameters

in	<i>xcdev</i>	pointer to character device
----	--------------	-----------------------------

Returns

none

4.1.3.2 int qdma_cdev_create (struct qdma_cdev_cb * xcb, struct pci_dev * pdev, struct qdma_queue_conf * qconf, unsigned int minor, unsigned long qhndl, struct qdma_cdev ** xcdev_pp, char * ebuf, int ebuflen)

[qdma_cdev_create\(\)](#) - handler to create a character device

Parameters

in	<i>xcb</i>	pointer to qdma character device call back data
in	<i>pdev</i>	pointer to struct pci_dev
in	<i>qconf</i>	queue configurations
in	<i>minor</i>	character device minor number
in	<i>ebuflen</i>	buffer length
in	<i>qhndl</i>	queue handle
out	<i>xcdev_pp</i>	pointer to struct qdma_cdev
out	<i>ebuf</i>	error message buffer, can be NULL/0 (i.e., optional)

Returns

0: success
<0: failure

4.1.3.3 void qdma_cdev_device_cleanup (struct qdma_cdev_cb * *xcb*)

[qdma_cdev_device_cleanup\(\)](#) - handler to clean up a character device

Parameters

in	<i>xcb</i>	pointer to qdma character device call back data
----	------------	---

Returns

none

4.1.3.4 int qdma_cdev_device_init (struct qdma_cdev_cb * *xcb*)

[qdma_cdev_device_init\(\)](#) - handler to initialize a character device

Parameters

in	<i>xcb</i>	pointer to qdma character device call back data
----	------------	---

Returns

0: success
<0: failure

4.1.3.5 void qdma_cdev_cleanup (void)

[qdma_cdev_cleanup\(\)](#) - character device cleanup handler

4.1.3.6 int qdma_cdev_init (void)

[qdma_cdev_init\(\)](#) - character device initialization handler

4.2 libqdma_config.h File Reference

```
#include <linux/types.h>
```

Macros

- `#define QDMA_CONFIG_BAR 0`
- `#define STM_BAR 2`
- `#define QDMA_PF_MAX 4 /* 4 PFs */`
- `#define QDMA_VF_MAX 252`
- `#define QDMA_Q_PER_PF_MAX 512`
- `#define MAX_DMA_DEV 32`
- `#define TOTAL_QDMA_QS (QDMA_PF_MAX * QDMA_Q_PER_PF_MAX)`
- `#define QDMA_Q_PER_VF_MAX 1`
- `#define TOTAL_VF_QS 0`
- `#define TOTAL_PF_QS (TOTAL_QDMA_QS - TOTAL_VF_QS)`
- `#define MAX_QS_PER_PF (TOTAL_PF_QS/QDMA_PF_MAX)`
- `#define PCI_SHIFT_BUS 12`
- `#define PCI_SHIFT_DEV 4`
- `#define SHIFT_DEC_PCI_BUS 1000`
- `#define SHIFT_DEC_PCI_DEV 10`
- `#define QDMA_DEV_MSIX_VEC_MAX 8`
- `#define QDMA_INTR_COAL_RING_SIZE INTR_RING_SZ_4KB`
- `#define QDMA_NUM_DATA_VEC_FOR_INTR_CXT 1`

Functions

- `int qdma_set_qmax` (unsigned long dev_hndl, u32 qsets_max, bool forced)
- `unsigned int qdma_get_qmax` (unsigned long dev_hndl)
- `int qdma_set_intr_rngsz` (unsigned long dev_hndl, u32 rngsz)
- `unsigned int qdma_get_intr_rngsz` (unsigned long dev_hndl)
- `int qdma_set_cmpl_status_acc` (unsigned long dev_hndl, u32 cmpl_status_acc)
- `unsigned int qdma_get_cmpl_status_acc` (unsigned long dev_hndl)
- `int qdma_set_buf_sz` (unsigned long dev_hndl, u32 *buf_sz)
- `unsigned int qdma_get_buf_sz` (unsigned long dev_hndl, u32 *buf_sz)
- `int qdma_set_glbl_rng_sz` (unsigned long dev_hndl, u32 *glbl_rng_sz)
- `unsigned int qdma_get_glbl_rng_sz` (unsigned long dev_hndl, u32 *glbl_rng_sz)
- `int qdma_set_timer_cnt` (unsigned long dev_hndl, u32 *tmr_cnt)
- `unsigned int qdma_get_timer_cnt` (unsigned long dev_hndl, u32 *tmr_cnt)
- `int qdma_set_cnt_thresh` (unsigned long dev_hndl, unsigned int *cnt_th)
- `unsigned int qdma_get_cnt_thresh` (unsigned long dev_hndl, u32 *cnt_th)

4.2.1 Detailed Description

This file contains the declarations for qdma configuration apis.

4.2.2 Macro Definition Documentation

4.2.2.1 `#define QDMA_CONFIG_BAR 0`

QDMA config bar number

4.2.2.2 #define STM_BAR 2

STM bar

4.2.2.3 #define QDMA_PF_MAX 4 /* 4 PFs */

Maximum number of physical functions

4.2.2.4 #define QDMA_VF_MAX 252

Maximum number of virtual functions

4.2.2.5 #define QDMA_Q_PER_PF_MAX 512

Maximum number of queues per physical function

4.2.2.6 #define MAX_DMA_DEV 32

Maximum number of QDMA devices in the system

4.2.2.7 #define TOTAL_QDMA_QS (QDMA_PF_MAX * QDMA_Q_PER_PF_MAX)

Total number of qdma qs

4.2.2.8 #define QDMA_Q_PER_VF_MAX 1

Maximum number of queues per virtual function

4.2.2.9 #define TOTAL_VF_QS 0

Total number of qs for all VF

4.2.2.10 #define TOTAL_PF_QS (TOTAL_QDMA_QS - TOTAL_VF_QS)

Total number of qs for all PFs

4.2.2.11 #define MAX_QS_PER_PF (TOTAL_PF_QS/QDMA_PF_MAX)

Maximum number of qs for PF

4.2.2.12 `#define PCI_SHIFT_BUS 12`

Shift for bus 'B' in B:D:F

4.2.2.13 `#define PCI_SHIFT_DEV 4`

Shift for device 'D' in B:D:F

4.2.2.14 `#define SHIFT_DEC_PCI_BUS 1000`

To shift the Bus number for getting BDF

4.2.2.15 `#define SHIFT_DEC_PCI_DEV 10`

To shift the device number for getting BDF

4.2.2.16 `#define QDMA_DEV_MSIX_VEC_MAX 8`

Maximum number of MSI-X vector per function

4.2.2.17 `#define QDMA_INTR_COAL_RING_SIZE INTR_RING_SZ_4KB`

ring size is 4KB, i.e 512 entries

4.2.2.18 `#define QDMA_NUM_DATA_VEC_FOR_INTR_CXT 1`

Maximum data vectors to be used for each function TODO: Please note that for 2018.2 only one vector would be used per pf and only one ring would be created for this vector It is also assumed that all functions have the same number of data vectors and currently different number of vectors per PF is not supported

4.2.3 Function Documentation

4.2.3.1 `int qdma_set_qmax (unsigned long dev_hdl, u32 qsets_max, bool forced)`

[qdma_set_qmax\(\)](#) - Handler function to set the qmax configuration value

Parameters

in	<i>dev_hdl</i>	qdma device handle
in	<i>qsets_max</i>	qmax configuration value
in	<i>forced</i>	flag to force qmax change

Returns

QDMA_OPERATION_SUCCESSFUL on success
<0 on failure

4.2.3.2 unsigned int qdma_get_qmax (unsigned long *dev_hndl*)

[qdma_get_qmax\(\)](#) - Handler function to get the qmax configuration value

Parameters

in	<i>dev_hndl</i>	qdma device handle
----	-----------------	--------------------

Returns

qmax value on success
< 0 on failure

4.2.3.3 int qdma_set_intr_rngsz (unsigned long *dev_hndl*, u32 *rngsz*)

[qdma_set_intr_rngsz\(\)](#) - Handler function to set the intr_ring_size value

Parameters

in	<i>dev_hndl</i>	qdma device handle
in	<i>rngsz</i>	interrupt aggregation ring size

Returns

QDMA_OPERATION_SUCCESSFUL on success
<0 on failure

4.2.3.4 unsigned int qdma_get_intr_rngsz (unsigned long *dev_hndl*)

[qdma_get_intr_rngsz\(\)](#) - Handler function to get the intr_ring_size value

Parameters

in	<i>dev_hndl</i>	qdma device handle
----	-----------------	--------------------

Returns

interrupt ring size on success
<0 on failure

4.2.3.5 `int qdma_set_cmpl_status_acc (unsigned long dev_hndl, u32 cmpl_status_acc)`

[qdma_set_cmpl_status_acc\(\)](#) - Handler function to set the `cmpl_status_acc` configuration value

Parameters

in	<i>dev_hndl</i>	qdma device handle
in	<i>cmpl_status_acc</i>	Writeback Accumulation value

Returns

QDMA_OPERATION_SUCCESSFUL on success
<0 on failure

4.2.3.6 `unsigned int qdma_get_cmpl_status_acc (unsigned long dev_hndl)`

[qdma_get_cmpl_status_acc\(\)](#) - Handler function to get the `cmpl_status_acc` configuration value

Parameters

in	<i>dev_hndl</i>	qdma device handle
----	-----------------	--------------------

Handler function to get the writeback accumulation value

Returns

`cmpl_status_acc` on success
<0 on failure

4.2.3.7 `int qdma_set_buf_sz (unsigned long dev_hndl, u32 * buf_sz)`

[qdma_set_buf_sz\(\)](#) - Handler function to set the `buf_sz` value

Parameters

in	<i>dev_hndl</i>	qdma device handle
in	<i>buf_sz</i>	buf sizes

Returns

QDMA_OPERATION_SUCCESSFUL on success
<0 on failure

4.2.3.8 `unsigned int qdma_get_buf_sz (unsigned long dev_hndl, u32 * buf_sz)`

[qdma_get_buf_sz\(\)](#) - Handler function to get the `buf_sz` value

Parameters

in	<i>dev_hndl</i>	qdma device handle
in	<i>buf_sz</i>	buf sizes

Returns

buf sizes on success
<0 on failure

4.2.3.9 `int qdma_set_glbl_rng_sz (unsigned long dev_hndl, u32 * glbl_rng_sz)`

[qdma_set_glbl_rng_sz\(\)](#) - Handler function to set the *glbl_rng_sz* value

Parameters

in	<i>dev_hndl</i>	qdma device handle
in	<i>glbl_rng_sz</i>	glbl_rng_sizes

Returns

QDMA_OPERATION_SUCCESSFUL on success
<0 on failure

4.2.3.10 `unsigned int qdma_get_glbl_rng_sz (unsigned long dev_hndl, u32 * glbl_rng_sz)`

[qdma_get_glbl_rng_sz\(\)](#) - Handler function to get the *glbl_rng_sz* value

Parameters

in	<i>dev_hndl</i>	qdma device handle
in	<i>glbl_rng_sz</i>	glbl_rng sizes

Returns

glbl_rng_sz on success
<0 on failure

4.2.3.11 `int qdma_set_timer_cnt (unsigned long dev_hndl, u32 * tmr_cnt)`

[qdma_set_timer_cnt\(\)](#) - Handler function to set the *buf_sz* value

Parameters

in	<i>dev_hndl</i>	qdma device handle
in	<i>tmr_cnt</i>	Array of 16 timer count values

Returns

QDMA_OPERATION_SUCCESSFUL on success
<0 on failure

4.2.3.12 `unsigned int qdma_get_timer_cnt (unsigned long dev_hndl, u32 * tmr_cnt)`

[qdma_get_timer_cnt\(\)](#) - Handler function to get the timer_cnt value

Parameters

in	<i>dev_hndl</i>	qdma device handle
----	-----------------	--------------------

Returns

timer_cnt on success
<0 on failure

4.2.3.13 `int qdma_set_cnt_thresh (unsigned long dev_hndl, unsigned int * cnt_th)`

[qdma_set_cnt_thresh\(\)](#) - Handler function to set the counter threshold value

Parameters

in	<i>dev_hndl</i>	qdma device handle
in	<i>cnt_th</i>	Array of 16 timer count values

Returns

QDMA_OPERATION_SUCCESSFUL on success
<0 on failure

4.2.3.14 `unsigned int qdma_get_cnt_thresh (unsigned long dev_hndl, u32 * cnt_th)`

[qdma_get_cnt_thresh\(\)](#) - Handler function to get the counter thresh value

Parameters

in	<i>dev_hndl</i>	qdma device handle
----	-----------------	--------------------

Returns

counter threshold values on success
<0 on failure

4.3 libqdma_export.h File Reference

```
#include <linux/types.h>
#include <linux/interrupt.h>
#include "libqdma_config.h"
```

Data Structures

- struct [drv_mode_name](#)
- struct [qdma_dev_conf](#)
- struct [qdma_version_info](#)
- struct [global_csr_conf](#)
- struct [qdma_sw_sg](#)
- struct [qdma_queue_conf](#)
- struct [qdma_request](#)
- struct [qdma_cmpl_ctrl](#)

Macros

- #define [QDMA_FUNC_ID_INVALID](#) (QDMA_PF_MAX + QDMA_VF_MAX)
- #define [QDMA_DEV_NAME_MAXLEN](#) 32
- #define [DEVICE_VERSION_INFO_STR_LENGTH](#) 10
- #define [QDMA_GLOBAL_CSR_ARRAY_SZ](#) 16
- #define [QDMA_QUEUE_NAME_MAXLEN](#) 32
- #define [QDMA_QUEUE_IDX_INVALID](#) 0xFFFF
- #define [QDMA_QUEUE_VEC_INVALID](#) 0xFF
- #define [QDMA_REQ_OPAQUE_SIZE](#) 72
- #define [QDMA_UDD_MAXLEN](#) 32

Enumerations

- enum [qdma_error_codes](#) {
[QDMA_OPERATION_SUCCESSFUL](#) = 0,
[QDMA_ERR_PCI_DEVICE_NOT_FOUND](#) = -1,
[QDMA_ERR_PCI_DEVICE_ALREADY_ATTACHED](#) = -2,
[QDMA_ERR_PCI_DEVICE_ENABLE_FAILED](#) = -3,
[QDMA_ERR_PCI_DEVICE_INIT_FAILED](#) = -4,
[QDMA_ERR_INVALID_INPUT_PARAM](#) = -5,
[QDMA_ERR_INVALID_PCI_DEV](#) = -6,
[QDMA_ERR_INVALID_QIDX](#) = -7,
[QDMA_ERR_INVALID_DESCQ_STATE](#) = -8,
[QDMA_ERR_INVALID_DIRECTION](#) = -9,
[QDMA_ERR_DESCQ_SETUP_FAILED](#) = -10,
[QDMA_ERR_DESCQ_FULL](#) = -11,
[QDMA_ERR_DESCQ_IDX_ALREADY_ADDED](#) = -12,
[QDMA_ERR_QUEUE_ALREADY_CONFIGURED](#) = -13,
[QDMA_ERR_OUT_OF_MEMORY](#) = -14,
[QDMA_ERR_INVALID_QDMA_DEVICE](#) = -15,
[QDMA_ERR_INTERFACE_NOT_ENABLED_IN_DEVICE](#) = -16 }

- enum `qdma_drv_mode` {
`AUTO_MODE`,
`POLL_MODE`,
`DIRECT_INTR_MODE`,
`INDIRECT_INTR_MODE`,
`LEGACY_INTR_MODE` }
- enum `intr_ring_size_sel` {
`INTR_RING_SZ_4KB` = 0,
`INTR_RING_SZ_8KB`,
`INTR_RING_SZ_12KB`,
`INTR_RING_SZ_16KB`,
`INTR_RING_SZ_20KB`,
`INTR_RING_SZ_24KB`,
`INTR_RING_SZ_28KB`,
`INTR_RING_SZ_32KB` }
- enum `qdma_dev_qmax_state` {
`QMAX_CFG_UNCONFIGURED`,
`QMAX_CFG_INITIAL`,
`QMAX_CFG_USER` }
- enum `cmpt_desc_sz_t` {
`CMPT_DESC_SZ_8B` = 0,
`CMPT_DESC_SZ_16B`,
`CMPT_DESC_SZ_32B`,
`CMPT_DESC_SZ_64B` }
- enum `desc_sz_t` {
`DESC_SZ_8B` = 0,
`DESC_SZ_16B`,
`DESC_SZ_32B`,
`DESC_SZ_64B` }
- enum `tigger_mode_t` {
`TRIG_MODE_DISABLE`,
`TRIG_MODE_ANY`,
`TRIG_MODE_COUNTER`,
`TRIG_MODE_USER`,
`TRIG_MODE_TIMER`,
`TRIG_MODE_COMBO` }

Functions

- int `libqdma_init` (enum `qdma_drv_mode` `qdma_drv_mode`, unsigned int `num_threads`)
- void `libqdma_exit` (void)
- int `qdma_device_open` (const char *`mod_name`, struct `qdma_dev_conf` *`conf`, unsigned long *`dev_hdl`)
- void `qdma_device_close` (struct `pci_dev` *`pdev`, unsigned long `dev_hdl`)
- void `qdma_device_offline` (struct `pci_dev` *`pdev`, unsigned long `dev_hdl`)
- int `qdma_device_online` (struct `pci_dev` *`pdev`, unsigned long `dev_hdl`)
- int `qdma_device_flr_quirk_set` (struct `pci_dev` *`pdev`, unsigned long `dev_hdl`)
- int `qdma_device_flr_quirk_check` (struct `pci_dev` *`pdev`, unsigned long `dev_hdl`)
- int `qdma_device_get_config` (unsigned long `dev_hdl`, struct `qdma_dev_conf` *`conf`, char *`ebuf`, int `ebufen`)
- int `qdma_device_clear_stats` (unsigned long `dev_hdl`)
- int `qdma_device_get_mmh2c_pkts` (unsigned long `dev_hdl`, unsigned long long *`mmh2c_pkts`)
- int `qdma_device_get_mmc2h_pkts` (unsigned long `dev_hdl`, unsigned long long *`mmc2h_pkts`)
- int `qdma_device_get_sth2c_pkts` (unsigned long `dev_hdl`, unsigned long long *`sth2c_pkts`)
- int `qdma_device_get_stc2h_pkts` (unsigned long `dev_hdl`, unsigned long long *`stc2h_pkts`)
- int `qdma_device_set_config` (unsigned long `dev_hdl`, struct `qdma_dev_conf` *`conf`)
- int `qdma_device_set_cfg_state` (unsigned long `dev_hdl`, enum `qdma_dev_qmax_state` `new_cfg_state`)

- int [qdma_device_sriov_config](#) (struct pci_dev *pdev, unsigned long dev_hdl, int num_vfs)
- unsigned int [qdma_device_read_config_register](#) (unsigned long dev_hdl, unsigned int reg_addr)
- void [qdma_device_write_config_register](#) (unsigned long dev_hdl, unsigned int reg_addr, u32 value)
- int [qdma_device_version_info](#) (unsigned long dev_hdl, struct [qdma_version_info](#) *version_info)
- int [qdma_software_version_info](#) (char *software_version)
- int [qdma_global_csr_get](#) (unsigned long dev_hdl, struct [global_csr_conf](#) *csr)
- int [qdma_global_csr_set](#) (unsigned long dev_hdl, struct [global_csr_conf](#) *csr)
- int [qdma_queue_add](#) (unsigned long dev_hdl, struct [qdma_queue_conf](#) *qconf, unsigned long *qhndl, char *buf, int buflen)
- int [qdma_queue_reconfig](#) (unsigned long dev_hdl, unsigned long id, struct [qdma_queue_conf](#) *qconf, char *buf, int buflen)
- int [qdma_queue_start](#) (unsigned long dev_hdl, unsigned long id, char *buf, int buflen)
- int [qdma_queue_stop](#) (unsigned long dev_hdl, unsigned long id, char *buf, int buflen)
- int [qdma_queue_prog_stm](#) (unsigned long dev_hdl, unsigned long id, char *buf, int buflen)
- int [qdma_queue_remove](#) (unsigned long dev_hdl, unsigned long id, char *buf, int buflen)
- struct [qdma_queue_conf](#) * [qdma_queue_get_config](#) (unsigned long dev_hdl, unsigned long id, char *buf, int buflen)
- int [qdma_queue_list](#) (unsigned long dev_hdl, char *buf, int buflen)
- int [qdma_queue_dump](#) (unsigned long dev_hdl, unsigned long id, char *buf, int buflen)
- int [qdma_queue_dump_desc](#) (unsigned long dev_hdl, unsigned long id, unsigned int start, unsigned int end, char *buf, int buflen)
- int [qdma_queue_dump_cmpt](#) (unsigned long dev_hdl, unsigned long id, unsigned int start, unsigned int end, char *buf, int buflen)
- ssize_t [qdma_request_submit](#) (unsigned long dev_hdl, unsigned long id, struct [qdma_request](#) *req)
- ssize_t [qdma_batch_request_submit](#) (unsigned long dev_hdl, unsigned long id, unsigned long count, struct [qdma_request](#) **reqv)
- int [qdma_queue_c2h_peek](#) (unsigned long dev_hdl, unsigned long qhndl, unsigned int *udd_cnt, unsigned int *pkt_cnt, unsigned int *data_len)
- int [qdma_queue_avail_desc](#) (unsigned long dev_hdl, unsigned long qhndl)
- int [qdma_queue_cmpl_ctrl](#) (unsigned long dev_hdl, unsigned long qhndl, struct [qdma_cmpl_ctrl](#) *cctrl, bool set)
- int [qdma_queue_packet_read](#) (unsigned long dev_hdl, unsigned long qhndl, struct [qdma_request](#) *req, struct [qdma_cmpl_ctrl](#) *cctrl)
- int [qdma_queue_packet_write](#) (unsigned long dev_hdl, unsigned long qhndl, struct [qdma_request](#) *req)
- void [qdma_queue_service](#) (unsigned long dev_hdl, unsigned long qhndl, int budget, bool c2h_upd_cmpl)
- int [qdma_intr_ring_dump](#) (unsigned long dev_hdl, unsigned int vector_idx, int start_idx, int end_idx, char *buf, int buflen)
- int [qdma_descq_get_cmpt_udd](#) (unsigned long dev_hdl, unsigned long qhndl, char *buf, int buflen)

Variables

- struct [drv_mode_name](#) **mode_name_list** []

4.3.1 Detailed Description

This file contains the declarations for libqdma interfaces.

4.3.2 Macro Definition Documentation

4.3.2.1 #define QDMA_FUNC_ID_INVALID (QDMA_PF_MAX + QDMA_VF_MAX)

Invalid QDMA function number

4.3.2.2 `#define QDMA_DEV_NAME_MAXLEN 32`

Maximum length of the QDMA device name

4.3.2.3 `#define DEVICE_VERSION_INFO_STR_LENGTH 10`

[qdma_version_info](#) defines the per-device version information

4.3.2.4 `#define QDMA_GLOBAL_CSR_ARRAY_SZ 16`

QDMA Global CSR array size

4.3.2.5 `#define QDMA_QUEUE_NAME_MAXLEN 32`

maximum queue name length

4.3.2.6 `#define QDMA_QUEUE_IDX_INVALID 0xFFFF`

invalid queue index

4.3.2.7 `#define QDMA_QUEUE_VEC_INVALID 0xFF`

invalid MSI-x vector index

4.3.2.8 `#define QDMA_REQ_OPAQUE_SIZE 72`

maximum request length

4.3.2.9 `#define QDMA_UDD_MAXLEN 32`

Max length of the user defined data

4.3.3 Enumeration Type Documentation

4.3.3.1 enum qdma_error_codes

QDMA Error codes

Enumerator

QDMA_OPERATION_SUCCESSFUL QDMA driver API operation successful

QDMA_ERR_PCI_DEVICE_NOT_FOUND QDMA PCI device not found on the PCIe bus

QDMA_ERR_PCI_DEVICE_ALREADY_ATTACHED QDMA PCI device already attached

QDMA_ERR_PCI_DEVICE_ENABLE_FAILED Failed to enable the QDMA PCIe device

QDMA_ERR_PCI_DEVICE_INIT_FAILED Failed to initialize the QDMA PCIe device

QDMA_ERR_INVALID_INPUT_PARAM Invalid input parameter given to QDMA API

QDMA_ERR_INVALID_PCI_DEV Invalid PCIe device

QDMA_ERR_INVALID_QIDX Invalid Queue ID provided as input

QDMA_ERR_INVALID_DESCQ_STATE Invalid descriptor queue state

QDMA_ERR_INVALID_DIRECTION Invalid descriptor direction provided

QDMA_ERR_DESCQ_SETUP_FAILED Failed to setup the descriptor queue

QDMA_ERR_DESCQ_FULL Descriptor queue is full

QDMA_ERR_DESCQ_IDX_ALREADY_ADDED Descriptor queue index is already added

QDMA_ERR_QUEUE_ALREADY_CONFIGURED Queue is already configured

QDMA_ERR_OUT_OF_MEMORY Out of memory

QDMA_ERR_INVALID_QDMA_DEVICE Invalid QDMA device, QDMA device is not yet created

QDMA_ERR_INTERFACE_NOT_ENABLED_IN_DEVICE The ST or MM or Both interface not enabled in the device

4.3.3.2 enum qdma_drv_mode

qdma_drv_state_t - qdma driver state

Enumerator

AUTO_MODE auto mode decided automatically, mix of poll and interrupt mode driver is inserted in poll mode

POLL_MODE driver is inserted in direct interrupt mode

DIRECT_INTR_MODE driver is inserted in indirect interrupt mode

INDIRECT_INTR_MODE driver is inserted in legacy interrupt mode

4.3.3.3 enum intr_ring_size_sel

intr_ring_size_sel - qdma interrupt ring size selection

Enumerator

INTR_RING_SZ_4KB 0 - INTR_RING_SZ_4KB, Accommodates 512 entries

INTR_RING_SZ_8KB 1 - INTR_RING_SZ_8KB, Accommodates 1024 entries

INTR_RING_SZ_12KB 2 - INTR_RING_SZ_12KB, Accommodates 1536 entries

INTR_RING_SZ_16KB 3 - INTR_RING_SZ_16KB, Accommodates 2048 entries

INTR_RING_SZ_20KB 4 - INTR_RING_SZ_20KB, Accommodates 2560 entries

INTR_RING_SZ_24KB 5 - INTR_RING_SZ_24KB, Accommodates 3072 entries

INTR_RING_SZ_28KB 6 - INTR_RING_SZ_24KB, Accommodates 3584 entries

INTR_RING_SZ_32KB 7 - INTR_RING_SZ_24KB, Accommodates 4096 entries

4.3.3.4 enum qdma_dev_qmax_state

Enumerator

QMAX_CFG_UNCONFIGURED device qmax not configured
QMAX_CFG_INITIAL device qmax configured with initial values
QMAX_CFG_USER device qmax configured from sysfs

4.3.3.5 enum cmpt_desc_sz_t

cmpt_desc_sz_t - descriptor sizes

Enumerator

CMPT_DESC_SZ_8B 0 - completion size 8B
CMPT_DESC_SZ_16B 0 - completion size 16B
CMPT_DESC_SZ_32B 0 - completion size 32B
CMPT_DESC_SZ_64B 0 - completion size 64B

4.3.3.6 enum desc_sz_t

desc_sz_t - descriptor sizes

Enumerator

DESC_SZ_8B 0 - size 8B
DESC_SZ_16B 0 - size 16B
DESC_SZ_32B 0 - size 32B
DESC_SZ_64B 0 - size 64B

4.3.3.7 enum tigger_mode_t

tigger_mode_t - trigger modes

Enumerator

TRIG_MODE_DISABLE 0 - disable trigger mode
TRIG_MODE_ANY 1 - any trigger mode
TRIG_MODE_COUNTER 2 - counter trigger mode
TRIG_MODE_USER 3 - trigger mode of user choice
TRIG_MODE_TIMER 4 - timer trigger mode
TRIG_MODE_COMBO 5 - timer and counter combo trigger mode

4.3.4 Function Documentation

4.3.4.1 int libqdma_init (enum qdma_drv_mode qdma_drv_mode, unsigned int num_threads)

[libqdma_init\(\)](#) initialize the QDMA core library

Parameters

in	<i>qdma_drv_mode</i>	- mode in which the driver needs to operate
in	<i>num_threads</i>	- number of threads to be created each for request processing and writeback processing

Returns

0: success
 <0: error

4.3.4.2 void libqdma_exit (void)

[libqdma_exit\(\)](#) cleanup the QDMA core library before exiting

Returns

none

4.3.4.3 int qdma_device_open (const char * *mod_name*, struct qdma_dev_conf * *conf*, unsigned long * *dev_hndl*)

[qdma_device_open\(\)](#) - read the pci bars and configure the fpga This API should be called from probe()

User interrupt will not be enabled until [qdma_user_isr_enable\(\)](#) is called

Parameters

in	<i>mod_name</i>	the module name, used for request_irq
in	<i>conf</i>	device configuration
out	<i>dev_hndl</i>	an opaque handle for libqdma to identify the device

Returns

QDMA_OPERATION_SUCCESSFUL success
 <0 in case of error

4.3.4.4 void qdma_device_close (struct pci_dev * *pdev*, unsigned long *dev_hndl*)

[qdma_device_close\(\)](#) - prepare fpga for removal: disable all interrupts (users and qdma) and release all resources. This API should be called from remove()

Parameters

in	<i>pdev</i>	ptr to struct pci_dev
in	<i>dev_hndl</i>	dev_hndl retured from qdma_device_open()

4.3.4.5 void qdma_device_offline (struct pci_dev * *pdev*, unsigned long *dev_hndl*)

[qdma_device_offline\(\)](#) - Set the device in offline mode

Parameters

in	<i>pdev</i>	ptr to struct pci_dev
in	<i>dev_hndl</i>	dev_hndl retured from qdma_device_open()

Returns

0: success
<0: error

4.3.4.6 int qdma_device_online (struct pci_dev * *pdev*, unsigned long *dev_hndl*)

[qdma_device_online\(\)](#) - Set the device in online mode and re-initialize it

Parameters

in	<i>pdev</i>	ptr to struct pci_dev
in	<i>dev_hndl</i>	dev_hndl retured from qdma_device_open()

Returns

0: success
<0: error

4.3.4.7 int qdma_device_flr_quirk_set (struct pci_dev * *pdev*, unsigned long *dev_hndl*)

[qdma_device_flr_quirk_set\(\)](#) - start pre-flr processing

Parameters

in	<i>pdev</i>	ptr to struct pci_dev
in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()

Returns

0: success
<0: error

4.3.4.8 int qdma_device_flr_quirk_check (struct pci_dev * *pdev*, unsigned long *dev_hndl*)

[qdma_device_flr_quirk_check\(\)](#) - check if pre-flr processing completed

Parameters

in	<i>pdev</i>	ptr to struct pci_dev
in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()

Returns

0: success
 <0: error

4.3.4.9 int qdma_device_get_config (unsigned long *dev_hndl*, struct qdma_dev_conf * *conf*, char * *ebuf*, int *ebuflen*)

[qdma_device_get_config\(\)](#) - retrieve the current device configuration

Parameters

in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
in	<i>conf</i>	device configuration
in	<i>ebuflen</i>	input buffer length
out	<i>ebuf</i>	error message buffer, can be NULL/0 (i.e., optional)

Returns

0: success
 <0: error

4.3.4.10 int qdma_device_clear_stats (unsigned long *dev_hndl*)

[qdma_device_clear_stats\(\)](#) - clear device statistics

Parameters

in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
----	-----------------	---

Returns

0: success
 <0: error

4.3.4.11 int qdma_device_get_mmh2c_pkts (unsigned long *dev_hndl*, unsigned long long * *mmh2c_pkts*)

[qdma_device_get_mmh2c_pkts\(\)](#) - get mm h2c packets processed

Parameters

in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
out	<i>mmh2c_pkts</i>	number of mm h2c packets processed

Returns

0: success
 <0: error

4.3.4.12 `int qdma_device_get_mmc2h_pkts (unsigned long dev_hdl, unsigned long long * mmc2h_pkts)`

[qdma_device_get_mmc2h_pkts\(\)](#) - get mm c2h packets processed

Parameters

in	<i>dev_hdl</i>	dev_hdl returned from qdma_device_open()
out	<i>mmc2h_pkts</i>	number of mm c2h packets processed

Returns

0: success
 <0: error

4.3.4.13 `int qdma_device_get_sth2c_pkts (unsigned long dev_hdl, unsigned long long * sth2c_pkts)`

[qdma_device_get_sth2c_pkts\(\)](#) - get st h2c packets processed

Parameters

in	<i>dev_hdl</i>	dev_hdl returned from qdma_device_open()
out	<i>sth2c_pkts</i>	number of st h2c packets processed

Returns

0: success
 <0: error

4.3.4.14 `int qdma_device_get_stc2h_pkts (unsigned long dev_hdl, unsigned long long * stc2h_pkts)`

[qdma_device_get_stc2h_pkts\(\)](#) - get st c2h packets processed

Parameters

in	<i>dev_hdl</i>	dev_hdl returned from qdma_device_open()
out	<i>stc2h_pkts</i>	number of st c2h packets processed

Returns

0: success
 <0: error

4.3.4.15 `int qdma_device_set_config (unsigned long dev_hndl, struct qdma_dev_conf * conf)`

[qdma_device_set_config\(\)](#) - set the current device configuration

Parameters

in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
in	<i>conf</i>	device configuration to set

Returns

0: success
<0: error

4.3.4.16 `int qdma_device_set_cfg_state (unsigned long dev_hndl, enum qdma_dev_qmax_state new_cfg_state)`

[qdma_device_set_cfg_state](#) - set the device configuration state

Parameters

in	<i>dev_hndl</i>	device handle
in	<i>new_cfg_state</i>	dma device conf state to set

Returns

0 on success , <0 on failure

4.3.4.17 `int qdma_device_sriov_config (struct pci_dev * pdev, unsigned long dev_hndl, int num_vfs)`

[qdma_device_sriov_config\(\)](#) - configure sriov

Parameters

in	<i>pdev</i>	ptr to struct <i>pci_dev</i>
in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
in	<i>num_vfs</i>	# of VFs to be instantiated

Returns

0: success
<0: error

4.3.4.18 `unsigned int qdma_device_read_config_register (unsigned long dev_hndl, unsigned int reg_addr)`

[qdma_device_read_config_register\(\)](#) - read dma config. register

Parameters

in	<i>dev_hdl</i>	dev_hdl returned from qdma_device_open()
in	<i>reg_addr</i>	register address

Returns

value of the config register

4.3.4.19 void qdma_device_write_config_register (unsigned long *dev_hdl*, unsigned int *reg_addr*, u32 *value*)

[qdma_device_write_config_register\(\)](#) - write dma config. register

Parameters

in	<i>dev_hdl</i>	dev_hdl returned from qdma_device_open()
in	<i>reg_addr</i>	register address
in	<i>value</i>	register value to be written

4.3.4.20 int qdma_device_version_info (unsigned long *dev_hdl*, struct qdma_version_info * *version_info*)

[qdma_device_version_info\(\)](#) - retrieve the RTL version , Vivado Release ID and Everest IP info

Parameters

in	<i>dev_hdl</i>	handle returned from qdma_device_open()
out	<i>version_info</i>	pointer to hold all the version details

Returns

0: success
<0: error

4.3.4.21 int qdma_software_version_info (char * *software_version*)

[qdma_software_version_info\(\)](#) - retrieve the software version

Parameters

out	<i>software_version</i>	A pointer to a null-terminated string
-----	-------------------------	---------------------------------------

Returns

0: success
<0: error

4.3.4.22 `int qdma_global_csr_get (unsigned long dev_hdl, struct global_csr_conf * csr)`

`qdma_glbai_csr_get()` - retrieve the global csr settings

Parameters

in	<i>dev_hdl</i>	handle returned from qdma_device_open()
out	<i>csr</i>	data structures to hold the csr values

Returns

0: success
<0: error

4.3.4.23 `int qdma_global_csr_set (unsigned long dev_hdl, struct global_csr_conf * csr)`

`qdma_glbai_csr_set()` - set the global csr values NOTE: for set, libqdma will enforce the access control

Parameters

in	<i>dev_hdl</i>	handle returned from qdma_device_open()
in	<i>csr</i>	data structures to hold the csr values

Returns

0: success
<0: error

4.3.4.24 `int qdma_queue_add (unsigned long dev_hdl, struct qdma_queue_conf * qconf, unsigned long * qhndl, char * buf, int buflen)`

[qdma_queue_add\(\)](#) - add a queue

Parameters

in	<i>dev_hdl</i>	<i>dev_hdl</i> returned from qdma_device_open()
in	<i>qconf</i>	queue configuration parameters
in	<i>qhndl</i>	list of unsigned long values that are the opaque <i>qhndl</i>
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

0: success
<0: error

4.3.4.25 `int qdma_queue_reconfig (unsigned long dev_hdl, unsigned long id, struct qdma_queue_conf * qconf, char * buf, int buflen)`

[qdma_queue_reconfig\(\)](#) - reconfigure the queue

Parameters

in	<i>dev_hdl</i>	qdma device handle
in	<i>id</i>	queue index
in	<i>qconf</i>	queue configuration
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

0: success
<0: failure

4.3.4.26 `int qdma_queue_start (unsigned long dev_hdl, unsigned long id, char * buf, int buflen)`

[qdma_queue_start\(\)](#) - start a queue (i.e, online, ready for dma)

Parameters

in	<i>dev_hdl</i>	<i>dev_hdl</i> returned from qdma_device_open()
in	<i>id</i>	the opaque qhdl
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

0: success
<0: error

4.3.4.27 `int qdma_queue_stop (unsigned long dev_hdl, unsigned long id, char * buf, int buflen)`

[qdma_queue_stop\(\)](#) - stop a queue (i.e., offline, NOT ready for dma)

Parameters

in	<i>dev_hdl</i>	<i>dev_hdl</i> returned from qdma_device_open()
in	<i>id</i>	the opaque qhdl
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

0: success
 <0: error

4.3.4.28 `int qdma_queue_prog_stm (unsigned long dev_hndl, unsigned long id, char * buf, int buflen)`

- [qdma_queue_prog_stm\(\)](#) - Program STM for queue (context, map, etc)

Parameters

in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
in	<i>id</i>	queue index
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

– 0: success
 <0: error

4.3.4.29 `int qdma_queue_remove (unsigned long dev_hndl, unsigned long id, char * buf, int buflen)`

[qdma_queue_remove\(\)](#) - remove a queue

Parameters

in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
in	<i>id</i>	the opaque qhndl
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

0: success
 <0: error

4.3.4.30 `struct qdma_queue_conf* qdma_queue_get_config (unsigned long dev_hndl, unsigned long id, char * buf, int buflen)`

queue helper/debug functions [qdma_queue_get_config\(\)](#) - retrieve the configuration of a queue

Parameters

in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
in	<i>id</i>	an opaque queue handle of type unsigned long
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

success: if optional message buffer used then strlen of buf, otherwise QDMA_OPERATION_SUCCESSFUL
 <0: error

4.3.4.31 `int qdma_queue_list (unsigned long dev_hdl, char * buf, int buflen)`

[qdma_queue_list\(\)](#) - display all configured queues in a string buffer

Parameters

in	<i>dev_hdl</i>	dev_hdl returned from qdma_device_open()
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

success: if optional message buffer used then strlen of buf, otherwise QDMA_OPERATION_SUCCESSFUL
 <0: error

4.3.4.32 `int qdma_queue_dump (unsigned long dev_hdl, unsigned long id, char * buf, int buflen)`

[qdma_queue_dump\(\)](#) - display a queue's state in a string buffer

Parameters

in	<i>dev_hdl</i>	dev_hdl returned from qdma_device_open()
in	<i>id</i>	an opaque queue handle of type unsigned long
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

success: if optional message buffer used then strlen of buf, otherwise QDMA_OPERATION_SUCCESSFUL
 <0: error

4.3.4.33 `int qdma_queue_dump_desc (unsigned long dev_hdl, unsigned long id, unsigned int start, unsigned int end, char * buf, int buflen)`

[qdma_queue_dump_desc\(\)](#) - display a queue's descriptor ring from index start ~ end in a string buffer

Parameters

in	<i>dev_hdl</i>	dev_hdl returned from qdma_device_open()
in	<i>id</i>	an opaque queue handle of type unsigned long
in	<i>start</i>	start index
in	<i>end</i>	end index
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

success: if optional message buffer used then strlen of buf, otherwise QDMA_OPERATION_SUCCESSFUL
 <0: error

4.3.4.34 `int qdma_queue_dump_cmpt (unsigned long dev_hndl, unsigned long id, unsigned int start, unsigned int end, char * buf, int buflen)`

[qdma_queue_dump_cmpt\(\)](#) - display a queue's descriptor ring from index start ~ end in a string buffer

Parameters

in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
in	<i>id</i>	an opaque queue handle of type unsigned long
in	<i>start</i>	start index
in	<i>end</i>	end index
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

success: if optional message buffer used then strlen of buf, otherwise QDMA_OPERATION_SUCCESSFUL
 <0: error

4.3.4.35 `ssize_t qdma_request_submit (unsigned long dev_hndl, unsigned long id, struct qdma_request * req)`

[qdma_request_submit\(\)](#) - submit a scatter-gather list of data for dma operation (for both read and write)

Parameters

in	<i>dev_hndl</i>	hndl returned from qdma_device_open()
in	<i>id</i>	queue index
in	<i>req</i>	qdma request

Returns

of bytes transferred
 <0: error

4.3.4.36 `ssize_t qdma_batch_request_submit (unsigned long dev_hndl, unsigned long id, unsigned long count, struct qdma_request ** reqv)`

[qdma_batch_request_submit\(\)](#) - submit a scatter-gather list of data for dma operation (for both read and write)

Parameters

in	<i>dev_hndl</i>	hndl returned from qdma_device_open()
in	<i>id</i>	queue index
in	<i>count</i>	number of requests
in	<i>reqv</i>	qdma request

Returns

of bytes transferred
 <0: error

4.3.4.37 `int qdma_queue_c2h_peek (unsigned long dev_hndl, unsigned long qhndl, unsigned int * udd_cnt, unsigned int * pkt_cnt, unsigned int * data_len)`

[qdma_queue_c2h_peek\(\)](#) - peek a receive (c2h) queue

Parameters

<i>dev_hndl</i>	hndl returned from qdma_device_open()
<i>qhndl</i>	hndl returned from qdma_queue_add()

filled in by libqdma:

Parameters

in	<i>udd_cnt</i>	# of udd received
in	<i>pkt_cnt</i>	# of packets received
in	<i>data_len</i>	# of bytes of packet data received

Returns

0: success and # of packets received in the Q
 <0: error

4.3.4.38 `int qdma_queue_avail_desc (unsigned long dev_hndl, unsigned long qhndl)`

[qdma_queue_avail_desc\(\)](#) - query of # of free descriptor

Parameters

in	<i>dev_hndl</i>	hndl returned from qdma_device_open()
in	<i>qhndl</i>	hndl returned from qdma_queue_add()

Returns

of available desc in the queue
 <0: error

4.3.4.39 `int qdma_queue_cmpl_ctrl (unsigned long dev_hndl, unsigned long qhndl, struct qdma_cmpl_ctrl * ctrl, bool set)`

[qdma_queue_cmpl_ctrl\(\)](#) - read/set the c2h Q's completion control

Parameters

in	<i>dev_hndl</i>	hndl returned from qdma_device_open()
in	<i>qhndl</i>	hndl returned from qdma_queue_add()
in	<i>cctrl</i>	completion control
in	<i>set</i>	read or set

Returns

0: success
 <0: error

4.3.4.40 `int qdma_queue_packet_read (unsigned long dev_hndl, unsigned long qhndl, struct qdma_request * req, struct qdma_cmpl_ctrl * cctrl)`

[qdma_queue_packet_read\(\)](#) - read rcv'ed data (ST C2H dma operation)

Parameters

in	<i>dev_hndl</i>	hndl returned from qdma_device_open()
in	<i>qhndl</i>	hndl returned from qdma_queue_add()
in	<i>req</i>	pointer to the request data
out	<i>cctrl</i>	completion control, if no change is desired, set it to NULL

Returns

number of packets read
 <0: error

4.3.4.41 `int qdma_queue_packet_write (unsigned long dev_hndl, unsigned long qhndl, struct qdma_request * req)`

[qdma_queue_packet_write\(\)](#) - submit data for h2c dma operation

Parameters

in	<i>dev_hndl</i>	hndl returned from qdma_device_open()
in	<i>qhndl</i>	hndl returned from qdma_queue_add()
in	<i>req</i>	pointer to the list of packet data

Returns

number of desc posted
 <0: error

4.3.4.42 void qdma_queue_service (unsigned long *dev_hndl*, unsigned long *qhndl*, int *budget*, bool *c2h_upd_cmpl*)

[qdma_queue_service\(\)](#) - service the queue in the case of irq handler is registered by the user, the user should call [qdma_queue_service\(\)](#) in its interrupt handler to service the queue

Parameters

in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
in	<i>qhndl</i>	hndl returned from qdma_queue_add()
in	<i>budget</i>	ST C2H only, max # of completions to be processed. 0 - no limit
in	<i>c2h_upd_cmpl</i>	To keep intrrupt disabled, set to false, Set to true to re-enable the interrupt: ST C2H only, max # of completions to be processed. 0 - no limit

4.3.4.43 int qdma_intr_ring_dump (unsigned long *dev_hndl*, unsigned int *vector_idx*, int *start_idx*, int *end_idx*, char * *buf*, int *buflen*)

[qdma_intr_ring_dump\(\)](#) - display the interrupt ring info of a vector

Parameters

in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
in	<i>vector_idx</i>	vector number
in	<i>start_idx</i>	interrupt ring start idx
in	<i>end_idx</i>	interrupt ring end idx
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message bufferuffer

Returns

0: success
<0: error

4.3.4.44 int qdma_descq_get_cmpt_udd (unsigned long *dev_hndl*, unsigned long *qhndl*, char * *buf*, int *buflen*)

[qdma_descq_get_cmpt_udd\(\)](#) - function to receive the user defined data

Parameters

in	<i>dev_hndl</i>	dev_hndl returned from qdma_device_open()
in	<i>qhndl</i>	queue handle
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message bufferuffer

Returns

0: success
<0: error

4.4 nl.h File Reference

```
#include <net/genetlink.h>
```

Macros

- `#define pr_fmt(fmt) KBUILD_MODNAME ":%s: " fmt, __func__`

Functions

- `int xnl_respond_buffer (struct genl_info *info, char *buf, int buflen)`

4.4.1 Detailed Description

This file contains the declarations for qdma netlink helper functions kernel module.

4.4.2 Function Documentation

4.4.2.1 `int xnl_respond_buffer (struct genl_info * info, char * buf, int buflen)`

`xnl_respond_buffer()` - send a netlink string message

Parameters

in	<i>nl_info</i>	pointer to netlink genl_info
in	<i>buf</i>	string buffer
in	<i>buflen</i>	length of the string buffer

Returns

0: success
<0: failure

4.5 pci_ids.h File Reference

Functions

- `MODULE_DEVICE_TABLE (pci, pci_ids)`

4.5.1 Detailed Description

This file contains the list of pcie devices supported for qdma driver.

4.5.2 Function Documentation

4.5.2.1 MODULE_DEVICE_TABLE (pci , pci_ids)

module device table

4.6 qdma_compat.h File Reference

```
#include <linux/version.h>
#include <asm/barrier.h>
#include <linux/swait.h>
```

Macros

- #define [qdma_wait_queue](#) struct swait_queue_head
- #define **qdma_waitq_init** init_swait_queue_head
- #define **qdma_waitq_wakeup** swake_up
- #define **qdma_waitq_wait_event** swait_event_interruptible
- #define **qdma_waitq_wait_event_timeout** swait_event_interruptible_timeout
- #define **qdma_timer_setup**(timer, fp_handler, data) timer_setup(timer, fp_handler, 0)
- #define **qdma_timer_start**(timer, expires) mod_timer(timer, round_jiffies(jiffies + (expires)))

4.6.1 Detailed Description

This file is used to allow the driver to be compiled under multiple versions of Linux with as few obtrusive in-line ifdef's as possible.

4.6.2 Macro Definition Documentation

4.6.2.1 #define qdma_wait_queue struct swait_queue_head

if linux kernel version is < 3.19.0 then define the dma_rmb and dma_wmb

4.7 qdma_context.h File Reference

```
#include "xdev.h"
#include "qdma_mbox.h"
```

Functions

- [int qdma_intr_context_setup](#) (struct [xlrx_dma_dev](#) *xdev)
- [int qdma_prog_intr_context](#) (struct [xlrx_dma_dev](#) *xdev, struct [mbox_msg_intr_ctxt](#) *ictxt)
- [int qdma_descq_context_setup](#) (struct [qdma_descq](#) *descq)
- [int qdma_descq_stm_setup](#) (struct [qdma_descq](#) *descq)
- [int qdma_descq_stm_clear](#) (struct [qdma_descq](#) *descq)
- [int qdma_descq_context_clear](#) (struct [xlrx_dma_dev](#) *xdev, unsigned int qid_hw, bool st, bool c2h, bool clr)
- [int qdma_descq_context_read](#) (struct [xlrx_dma_dev](#) *xdev, unsigned int qid_hw, bool st, bool c2h, struct [hw_descq_context](#) *ctxt)
- [int qdma_intr_context_read](#) (struct [xlrx_dma_dev](#) *xdev, int ring_index, unsigned int ctxt_sz, u32 *context)
- [int qdma_descq_context_program](#) (struct [xlrx_dma_dev](#) *xdev, unsigned int qid_hw, bool st, bool c2h, struct [hw_descq_context](#) *ctxt)
- [int qdma_descq_stm_read](#) (struct [xlrx_dma_dev](#) *xdev, unsigned int qid_hw, u8 pipe_flow_id, bool c2h, bool map, bool ctxt, struct [stm_descq_context](#) *context)
- [int qdma_descq_stm_program](#) (struct [xlrx_dma_dev](#) *xdev, unsigned int qid_hw, uint8_t pipe_flow_id, bool c2h, bool clear, struct [stm_descq_context](#) *context)

4.7.1 Detailed Description

This file contains the declarations for qdma context handlers.

4.7.2 Function Documentation

4.7.2.1 [int qdma_intr_context_setup](#) (struct [xlrx_dma_dev](#) * xdev)

[qdma_intr_context_setup\(\)](#) - handler to set the qdma interrupt context

Parameters

in	<i>xdev</i>	pointer to xdev
----	-------------	-----------------

Returns

0: success
<0: failure

4.7.2.2 [int qdma_prog_intr_context](#) (struct [xlrx_dma_dev](#) * xdev, struct [mbox_msg_intr_ctxt](#) * ictxt)

[qdma_prog_intr_context\(\)](#) - handler to program the qdma interrupt context for VF from PF

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>ictxt</i>	interrupt context

Returns

0: success
<0: failure

4.7.2.3 int qdma_descq_context_setup (struct qdma_descq * descq)

[qdma_descq_context_setup\(\)](#) - handler to set the qdma sw descriptor context

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

0: success
<0: failure

4.7.2.4 int qdma_descq_stm_setup (struct qdma_descq * descq)

[qdma_descq_stm_setup\(\)](#) - handler to set the qdma stm

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

0: success
<0: failure

4.7.2.5 int qdma_descq_stm_clear (struct qdma_descq * descq)

[qdma_descq_stm_clear\(\)](#) - handler to clear the qdma stm

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

0: success
<0: failure

4.7.2.6 `int qdma_descq_context_clear (struct xlnx_dma_dev * xdev, unsigned int qid_hw, bool st, bool c2h, bool clr)`

[qdma_descq_context_clear\(\)](#) - handler to clear the qdma sw descriptor context

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>qid_hw</i>	hw qidx
in	<i>st</i>	indicated whether the mm mode or st mode
in	<i>c2h</i>	indicates whether the h2c or c2h direction
in	<i>clr</i>	flag to indicate whether to clear the context or not

Returns

0: success
 <0: failure

4.7.2.7 `int qdma_descq_context_read (struct xlnx_dma_dev * xdev, unsigned int qid_hw, bool st, bool c2h, struct hw_descq_context * ctxt)`

[qdma_descq_context_read\(\)](#) - handler to read the queue context

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>qid_hw</i>	hw qidx
in	<i>st</i>	indicated whether the mm mode or st mode
in	<i>c2h</i>	indicates whether the h2c or c2h direction
out	<i>ctxt</i>	pointer to context data

Returns

0: success
 <0: failure

4.7.2.8 `int qdma_intr_context_read (struct xlnx_dma_dev * xdev, int ring_index, unsigned int ctxt_sz, u32 * context)`

[qdma_intr_context_read\(\)](#) - handler to read the interrupt context

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>ring_index</i>	interrupt ring index
in	<i>ctxt_sz</i>	context size
out	<i>context</i>	pointer to interrupt context*

Returns

0: success
 <0: failure

4.7.2.9 `int qdma_descq_context_program (struct xlnx_dma_dev * xdev, unsigned int qid_hw, bool st, bool c2h, struct hw_descq_context * ctxt)`

[qdma_descq_context_read\(\)](#) - handler to program the context for vf

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>qid_hw</i>	hw qidx
in	<i>st</i>	indicated whether the mm mode or st mode
in	<i>c2h</i>	indicates whether the h2c or c2h direction
out	<i>ctxt</i>	pointer to context data

Returns

0: success
<0: failure

4.7.2.10 `int qdma_descq_stm_read (struct xlnx_dma_dev * xdev, unsigned int qid_hw, u8 pipe_flow_id, bool c2h, bool map, bool ctxt, struct stm_descq_context * context)`

[qdma_descq_stm_read\(\)](#) - handler to read stm context, can, maps

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>qid_hw</i>	hw qidx
in	<i>pipe_flow_id</i>	pipe_flow_id for queue
in	<i>c2h</i>	indicates whether the h2c or c2h direction
in	<i>map</i>	indicates whether to read map or ctxt/can
in	<i>ctxt</i>	indicates whether to read ctxt or can
out	<i>context</i>	pointer to context data

Returns

0: success
<0: failure

4.7.2.11 `int qdma_descq_stm_program (struct xlnx_dma_dev * xdev, unsigned int qid_hw, uint8_t pipe_flow_id, bool c2h, bool clear, struct stm_descq_context * context)`

[qdma_descq_stm_program\(\)](#) - handler to program the stm

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>qid_hw</i>	hw qidx

Parameters

in	<i>pipe_flow↔ _id</i>	flow id for pipe
in	<i>c2h</i>	indicates whether the h2c or c2h direction
in	<i>clear</i>	flag to prog/clear stm context/maps
out	<i>stm</i>	pointer to stm data

Returns

0: success
<0: failure

4.8 qdma_descq.h File Reference

```
#include <linux/spinlock_types.h>
#include <linux/types.h>
#include "qdma_compat.h"
#include "libqdma_export.h"
#include "qdma_regs.h"
```

Data Structures

- struct [qdma_descq](#)
- struct [qdma_sgt_req_cb](#)

Macros

- #define **QDMA_FLQ_SIZE** 80
- #define [lock_descq](#)(descq) spin_lock_bh(&(descq)->lock)
- #define [unlock_descq](#)(descq) spin_unlock_bh(&(descq)->lock)
- #define [qdma_req_cb_get](#)(req) (struct [qdma_sgt_req_cb](#) *)((req)->opaque)

Enumerations

- enum [q_state_t](#) {
 Q_STATE_DISABLED = 0,
 [Q_STATE_ENABLED](#),
 [Q_STATE_ONLINE](#) }
- enum **qdma_req_state** {
 QDMA_REQ_NOT_SUBMITTED,
 QDMA_REQ_SUBMIT_PARTIAL,
 QDMA_REQ_SUBMITTED,
 QDMA_REQ_COMPLETE }

Functions

- void [qdma_descq_init](#) (struct [qdma_descq](#) *descq, struct [xlNX_dma_dev](#) *xdev, int idx_hw, int idx_sw)
- void [qdma_descq_config](#) (struct [qdma_descq](#) *descq, struct [qdma_queue_conf](#) *qconf, int reconfig)
- int [qdma_descq_config_complete](#) (struct [qdma_descq](#) *descq)
- void [qdma_descq_cleanup](#) (struct [qdma_descq](#) *descq)
- int [qdma_descq_alloc_resource](#) (struct [qdma_descq](#) *descq)
- void [qdma_descq_free_resource](#) (struct [qdma_descq](#) *descq)
- int [qdma_descq_prog_hw](#) (struct [qdma_descq](#) *descq)
- int [qdma_descq_prog_stm](#) (struct [qdma_descq](#) *descq, bool clear)
- int [qdma_descq_context_cleanup](#) (struct [qdma_descq](#) *descq)
- void [qdma_descq_service_cmpl_update](#) (struct [qdma_descq](#) *descq, int budget, bool c2h_upd_cmpl)
- int [qdma_descq_dump](#) (struct [qdma_descq](#) *descq, char *buf, int buflen, int detail)
- int [qdma_descq_dump_desc](#) (struct [qdma_descq](#) *descq, int start, int end, char *buf, int buflen)
- int [qdma_descq_dump_state](#) (struct [qdma_descq](#) *descq, char *buf, int buflen)
- void [intr_cidx_update](#) (struct [qdma_descq](#) *descq, unsigned int sw_cidx, int ring_index)
- void [incr_cmpl_desc_cnt](#) (struct [qdma_descq](#) *descq, unsigned int cnt)
- ssize_t [qdma_descq_proc_sgt_request](#) (struct [qdma_descq](#) *descq)
- void [qdma_sgt_req_done](#) (struct [qdma_descq](#) *descq, struct [qdma_sgt_req_cb](#) *cb, int error)
- int [sgl_map](#) (struct pci_dev *pdev, struct [qdma_sw_sg](#) *sg, unsigned int sgcnt, enum dma_data_direction dir)
- void [sgl_unmap](#) (struct pci_dev *pdev, struct [qdma_sw_sg](#) *sg, unsigned int sgcnt, enum dma_data_direction dir)
- void [descq_flq_free_resource](#) (struct [qdma_descq](#) *descq)

4.8.1 Detailed Description

This file contains the declarations for qdma descq processing.

4.8.2 Macro Definition Documentation

4.8.2.1 `#define lock_descq(descq) spin_lock_bh(&(descq)->lock)`

macro to lock descq

4.8.2.2 `#define unlock_descq(descq) spin_unlock_bh(&(descq)->lock)`

macro to un lock descq

4.8.2.3 `#define qdma_req_cb_get(req) (struct qdma_sgt_req_cb *)((req)->opaque)`

macro to get the request call back data

4.8.3 Enumeration Type Documentation

4.8.3.1 `enum q_state_t`

Enumerator

Q_STATE_ENABLED Queue is not taken

Q_STATE_ONLINE Assigned/taken. Partial config is done

4.8.4 Function Documentation

4.8.4.1 void qdma_descq_init (struct qdma_descq * *descq*, struct xlnx_dma_dev * *xdev*, int *idx_hw*, int *idx_sw*)

[qdma_descq_init\(\)](#) - initialize the sw descq entry

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>xdev</i>	pointer to xdev
in	<i>idx_hw</i>	hw queue index
in	<i>idx_sw</i>	sw queue index

Returns

none

4.8.4.2 void [qdma_descq_config](#) (struct [qdma_descq](#) * *descq*, struct [qdma_queue_conf](#) * *qconf*, int *reconfig*)

[qdma_descq_config\(\)](#) - configure the sw descq entry

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>qconf</i>	queue configuration
in	<i>reconfig</i>	flag to indicate whether to refig the sw descq

Returns

none

4.8.4.3 int [qdma_descq_config_complete](#) (struct [qdma_descq](#) * *descq*)

[qdma_descq_config_complete\(\)](#) - initialize the descq with default values

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

0: success
<0: failure

4.8.4.4 void [qdma_descq_cleanup](#) (struct [qdma_descq](#) * *descq*)

[qdma_descq_cleanup\(\)](#) - clean up the resources assigned to a request

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

none

4.8.4.5 `int qdma_descq_alloc_resource (struct qdma_descq * descq)`

[qdma_descq_alloc_resource\(\)](#) - allocate the resources for a request

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

0: success
<0: failure

4.8.4.6 `void qdma_descq_free_resource (struct qdma_descq * descq)`

[qdma_descq_free_resource\(\)](#) - free up the resources assigned to a request

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

none

4.8.4.7 `int qdma_descq_prog_hw (struct qdma_descq * descq)`

[qdma_descq_prog_hw\(\)](#) - program the hw descriptors

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

0: success
<0: failure

4.8.4.8 `int qdma_descq_prog_stm (struct qdma_descq * descq, bool clear)`

[qdma_descq_prog_stm\(\)](#) - program the STM

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>clear</i>	flag to program/clear stm context

Returns

0: success
 <0: failure

4.8.4.9 `int qdma_descq_context_cleanup (struct qdma_descq * descq)`

[qdma_descq_context_cleanup\(\)](#) - clean up the queue context

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

0: success
 <0: failure

4.8.4.10 `void qdma_descq_service_cmpl_update (struct qdma_descq * descq, int budget, bool c2h_upd_cmpl)`

[qdma_descq_service_cmpl_update\(\)](#) - process completion data for the request

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>budget</i>	number of descriptors to process
in	<i>c2h_upd_cmpl</i>	C2H only: if update completion needed

Returns

none

4.8.4.11 `int qdma_descq_dump (struct qdma_descq * descq, char * buf, int buflen, int detail)`

[qdma_descq_dump\(\)](#) - dump the queue sw descriptor data

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>detail</i>	indicate whether full details or abstract details
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

0: success
 <0: failure

4.8.4.12 `int qdma_descq_dump_desc (struct qdma_descq * descq, int start, int end, char * buf, int buflen)`

[qdma_descq_dump_desc\(\)](#) - dump the queue hw descriptors

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>start</i>	start descriptor index
in	<i>end</i>	end descriptor index
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

0: success
 <0: failure

4.8.4.13 `int qdma_descq_dump_state (struct qdma_descq * descq, char * buf, int buflen)`

[qdma_descq_dump_state\(\)](#) - dump the queue descriptor state

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

0: success
 <0: failure

4.8.4.14 `void intr_cidx_update (struct qdma_descq * descq, unsigned int sw_cidx, int ring_index)`

[intr_cidx_update\(\)](#) - update the interrupt cidx

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>sw_cidx</i>	sw cidx
in	<i>ring_index</i>	ring index

4.8.4.15 void `incr_cmpl_desc_cnt` (struct `qdma_descq` * *descq*, unsigned int *cnt*)

[`incr_cmpl_desc_cnt\(\)`](#) - update the interrupt cidx

Parameters

in	<i>descq</i>	pointer to <code>qdma_descq</code>
----	--------------	--

4.8.4.16 ssize_t `qdma_descq_proc_sgt_request` (struct `qdma_descq` * *descq*)

[`qdma_descq_proc_sgt_request\(\)`](#) - handler to process the qdma read/write request

Parameters

in	<i>descq</i>	pointer to <code>qdma_descq</code>
in	<i>cb</i>	scatter gather list call back data

Returns

size of the request

4.8.4.17 void `qdma_sgt_req_done` (struct `qdma_descq` * *descq*, struct `qdma_sgt_req_cb` * *cb*, int *error*)

[`qdma_sgt_req_done\(\)`](#) - handler to track the progress of the request

Parameters

in	<i>descq</i>	pointer to <code>qdma_descq</code>
in	<i>cb</i>	scatter gather list call back data
out	<i>error</i>	indicates the error status

Returns

none

4.8.4.18 int `sgl_map` (struct `pci_dev` * *pdev*, struct `qdma_sw_sg` * *sg*, unsigned int *sgcnt*, enum `dma_data_direction` *dir*)

[`sgl_map\(\)`](#) - handler to map the scatter gather list to kernel pages

Parameters

in	<i>pdev</i>	pointer to struct <code>pci_dev</code>
in	<i>sg</i>	scatter gather list
in	<i>sgcnt</i>	number of entries in scatter gather list
in	<i>dir</i>	direction of the request

Returns

none

4.8.4.19 `void sgl_unmap (struct pci_dev * pdev, struct qdma_sw_sg * sg, unsigned int sgcnt, enum dma_data_direction dir)`

[sgl_unmap\(\)](#) - handler to unmap the scatter gather list and free the kernel pages

Parameters

in	<i>pdev</i>	pointer to struct pci_dev
in	<i>sg</i>	scatter gather list
in	<i>sgcnt</i>	number of entries in scatter gather list
in	<i>dir</i>	direction of the request

Returns

none

4.8.4.20 `void descq_flq_free_resource (struct qdma_descq * descq)`

[descq_flq_free_resource\(\)](#) - handler to free the pages for the request

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

none

4.9 qdma_device.h File Reference

```
#include <linux/spinlock_types.h>
#include "libqdma_export.h"
```

Data Structures

- struct [qdma_dev](#)
- struct [qdma_csr_info](#)

Macros

- #define [xdev_2_qdev](#)(xdev) (struct [qdma_dev](#) *)((xdev)->dev_priv)

Enumerations

- enum [csr_type](#) {
QDMA_CSR_TYPE_NONE,
QDMA_CSR_TYPE_RNGSZ,
QDMA_CSR_TYPE_BUFSZ,
QDMA_CSR_TYPE_TIMER_CNT,
QDMA_CSR_TYPE_CNT_TH,
QDMA_CSR_TYPE_MAX }

Functions

- int [qdma_device_init](#) (struct [xlrx_dma_dev](#) *xdev)
- void [qdma_device_cleanup](#) (struct [xlrx_dma_dev](#) *xdev)
- long [qdma_device_get_id_from_descq](#) (struct [xlrx_dma_dev](#) *xdev, struct [qdma_descq](#) *descq)
- struct [qdma_descq](#) * [qdma_device_get_descq_by_id](#) (struct [xlrx_dma_dev](#) *xdev, unsigned long idx, char *buf, int buflen, int init)
- struct [qdma_descq](#) * [qdma_device_get_descq_by_hw_qid](#) (struct [xlrx_dma_dev](#) *xdev, unsigned long qidx_hw, u8 c2h)
- int [qdma_device_prep_q_resource](#) (struct [xlrx_dma_dev](#) *xdev)
- void [qdma_csr_read_cmpl_status_acc](#) (struct [xlrx_dma_dev](#) *xdev, unsigned int *cs_acc)
- void [qdma_csr_read_rngsz](#) (struct [xlrx_dma_dev](#) *xdev, unsigned int *rngsz)
- void [qdma_csr_read_bufsz](#) (struct [xlrx_dma_dev](#) *xdev, unsigned int *bufsz)
- void [qdma_csr_read_timer_cnt](#) (struct [xlrx_dma_dev](#) *xdev, unsigned int *cnt)
- void [qdma_csr_read_cnt_thresh](#) (struct [xlrx_dma_dev](#) *xdev, unsigned int *th)
- int [qdma_csr_read](#) (struct [xlrx_dma_dev](#) *xdev, struct [qdma_csr_info](#) *csr_info, unsigned int timeout_ms)

4.9.1 Detailed Description

This file contains the declarations for QDMA device.

4.9.2 Macro Definition Documentation

4.9.2.1 `#define xdev_2_qdev(xdev) (struct qdma_dev *)((xdev)->dev_priv)`

macro to convert the given xdev to qdev

4.9.3 Enumeration Type Documentation

4.9.3.1 `enum csr_type`

Enumerator

QDMA_CSR_TYPE_BUFSZ all global csr ring size settings
QDMA_CSR_TYPE_TIMER_CNT all global csr buffer size settings
QDMA_CSR_TYPE_CNT_TH all global csr timer count settings
QDMA_CSR_TYPE_MAX all global csr counter thresh settings

4.9.4 Function Documentation

4.9.4.1 `int qdma_device_init(struct xlrx_dma_dev * xdev)`

[qdma_device_init\(\)](#) - initializes the qdma device

Parameters

in	<i>xdev</i>	pointer to xdev
----	-------------	-----------------

Returns

0: success
-1: failure

4.9.4.2 void qdma_device_cleanup (struct xlnx_dma_dev * *xdev*)

[qdma_device_cleanup\(\)](#) - clean up the qdma device

Parameters

in	<i>xdev</i>	pointer to xdev
----	-------------	-----------------

Returns

none

4.9.4.3 long qdma_device_get_id_from_descq (struct xlnx_dma_dev * *xdev*, struct qdma_descq * *descq*)

[qdma_device_get_descq_by_id\(\)](#) - get the qhndl for descq

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>descq</i>	pointer to the descq

Returns

qhndl for descq on success
<0 on failure

4.9.4.4 struct qdma_descq* qdma_device_get_descq_by_id (struct xlnx_dma_dev * *xdev*, unsigned long *idx*, char * *buf*, int *buflen*, int *init*)

[qdma_device_get_descq_by_id\(\)](#) - get the descq using the qid

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>idx</i>	sw qidx
in	<i>init</i>	indicates whether to initialize the device or not
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

pointer to descq on success
NULL on failure

4.9.4.5 `struct qdma_descq* qdma_device_get_descq_by_hw_qid (struct xlnx_dma_dev * xdev, unsigned long qidx_hw, u8 c2h)`

[qdma_device_get_descq_by_hw_qid\(\)](#) - get the descq using the hw qid

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>qidx_hw</i>	hw qidx
in	<i>c2h</i>	indicates whether hw qidx belongs to c2h or h2c

Returns

pointer to descq on success
NULL on failure

4.9.4.6 `int qdma_device_prep_q_resource (struct xlnx_dma_dev * xdev)`

[qdma_device_prep_q_resource\(\)](#) - Prepare queue resources

Parameters

in	<i>xdev</i>	pointer to xdev
----	-------------	-----------------

Returns

0: success
<0: failure

4.9.4.7 `void qdma_csr_read_cmpl_status_acc (struct xlnx_dma_dev * xdev, unsigned int * cs_acc)`

[qdma_csr_read_cmpl_status_acc\(\)](#) - Read the completion status accumulation value

Parameters

in	<i>xdev</i>	pointer to xdev
out	<i>cs_acc</i>	cs_acc value

Returns

none

4.9.4.8 void qdma_csr_read_rngsz (struct xlnx_dma_dev * *xdev*, unsigned int * *rngsz*)

[qdma_csr_read_rngsz\(\)](#) - Read the queue ring size

Parameters

in	<i>xdev</i>	pointer to xdev
out	<i>rngsz</i>	queue ring size

Returns

none

4.9.4.9 void qdma_csr_read_bufsz (struct xlnx_dma_dev * *xdev*, unsigned int * *bufsz*)

[qdma_csr_read_bufsz\(\)](#) - Read the buffer size

Parameters

in	<i>xdev</i>	pointer to xdev
out	<i>bufsz</i>	buffer size

Returns

none

4.9.4.10 void qdma_csr_read_timer_cnt (struct xlnx_dma_dev * *xdev*, unsigned int * *cnt*)

[qdma_csr_read_timer_cnt\(\)](#) - Read the timer count

Parameters

in	<i>xdev</i>	pointer to xdev
out	<i>cnt</i>	timer count

Returns

none

4.9.4.11 void qdma_csr_read_cnt_thresh (struct xlnx_dma_dev * *xdev*, unsigned int * *th*)

[qdma_csr_read_timer_cnt\(\)](#) - Read the timer threshold

Parameters

in	<i>xdev</i>	pointer to xdev
out	<i>th</i>	timer threshold

Returns

none

4.9.4.12 `int qdma_csr_read (struct xlnx_dma_dev * xdev, struct qdma_csr_info * csr_info, unsigned int timeout_ms)`

[qdma_csr_read\(\)](#) - Read specific global csr registers

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>csr</i>	csr type & index
out	<i>csr</i>	csr value

Returns

0: success
<0: failure

4.10 qdma_intr.h File Reference

```
#include <linux/types.h>
#include <linux/workqueue.h>
#include "qdma_descq.h"
```

Data Structures

- struct [qdma_intr_ring](#)

Functions

- void [intr_teardown](#) (struct [xlnx_dma_dev](#) *xdev)
- int [intr_setup](#) (struct [xlnx_dma_dev](#) *xdev)
- void [intr_ring_teardown](#) (struct [xlnx_dma_dev](#) *xdev)
- int [intr_context_setup](#) (struct [xlnx_dma_dev](#) *xdev)
- int [intr_ring_setup](#) (struct [xlnx_dma_dev](#) *xdev)
- void [intr_legacy_init](#) (void)
- int [intr_legacy_setup](#) (struct [qdma_descq](#) *descq)
- void [intr_legacy_clear](#) (struct [qdma_descq](#) *descq)
- void [intr_work](#) (struct work_struct *work)
- void [qdma_err_intr_setup](#) (struct [xlnx_dma_dev](#) *xdev, u8 rearm)
- void [qdma_enable_hw_err](#) (struct [xlnx_dma_dev](#) *xdev, u8 hw_err_type)
- int [get_intr_ring_index](#) (struct [xlnx_dma_dev](#) *xdev, u32 vector_index)

4.10.1 Detailed Description

This file contains the declarations for qdma dev interrupt handlers.

4.10.2 Function Documentation

4.10.2.1 void intr_teardown (struct xlnx_dma_dev * xdev)

[intr_teardown\(\)](#) - un register the interrupts for the device

Parameters

in	<i>xdev</i>	pointer to xdev
----	-------------	-----------------

Returns

none

4.10.2.2 int intr_setup (struct xlnx_dma_dev * xdev)

[intr_setup\(\)](#) - register the interrupts for the device

Parameters

in	<i>xdev</i>	pointer to xdev
----	-------------	-----------------

Returns

0: success

<0: failure

4.10.2.3 void intr_ring_teardown (struct xlnx_dma_dev * xdev)

[intr_ring_teardown\(\)](#) - delete the interrupt ring

Parameters

in	<i>xdev</i>	pointer to xdev
----	-------------	-----------------

Returns

none

4.10.2.4 int intr_context_setup (struct xlnx_dma_dev * xdev)

[intr_context_setup\(\)](#) - set up the interrupt context

Parameters

in	<i>xdev</i>	pointer to xdev
----	-------------	-----------------

Returns

0: success
<0: failure

4.10.2.5 int intr_ring_setup (struct xlnx_dma_dev * xdev)

[intr_ring_setup\(\)](#) - create the interrupt ring

Parameters

in	<i>xdev</i>	pointer to xdev
----	-------------	-----------------

Returns

0: success
<0: failure

4.10.2.6 void intr_legacy_init (void)

[intr_legacy_init\(\)](#) - legacy interrupt init

4.10.2.7 int intr_legacy_setup (struct qdma_descq * descq)

[intr_legacy_setup\(\)](#) - setup the legacy interrupt handler

Parameters

in	<i>descq</i>	descq on which the interrupt needs to be setup
----	--------------	--

Returns

0: success
<0: failure

4.10.2.8 void intr_legacy_clear (struct qdma_descq * descq)

[intr_legacy_clear\(\)](#) - clear the legacy interrupt handler

Parameters

in	<i>descq</i>	descq on which the interrupt needs to be cleared
----	--------------	--

Returns

0: success
<0: failure

4.10.2.9 void `intr_work` (struct `work_struct` * *work*)

[`intr_work\(\)`](#) - attach the top half for the interrupt

Parameters

in	<i>work</i>	pointer to struct <code>work_struct</code>
----	-------------	--

Returns

none

4.10.2.10 void `qdma_err_intr_setup` (struct `xlnx_dma_dev` * *xdev*, u8 *rearm*)

[`qdma_err_intr_setup\(\)`](#) - set up the error interrupt

Parameters

in	<i>xdev</i>	pointer to <code>xdev</code>
in	<i>rearm</i>	flag to control the error interrupt arming

Returns

none

4.10.2.11 void `qdma_enable_hw_err` (struct `xlnx_dma_dev` * *xdev*, u8 *hw_err_type*)

[`qdma_enable_hw_err\(\)`](#) - enable the hw errors

Parameters

in	<i>xdev</i>	pointer to <code>xdev</code>
in	<i>hw_err_type</i>	hw error type

Returns

none

4.10.2.12 `int get_intr_ring_index (struct xlnx_dma_dev * xdev, u32 vector_index)`

[get_intr_ring_index\(\)](#) - get the interrupt ring index based on vector index

Parameters

in	<i>xdev</i>	pointer to xdev
in	<i>vector_index</i>	vector index

Returns

0: success
<0: failure

4.11 qdma_mbox.h File Reference

```
#include "qdma_compat.h"
#include "qdma_device.h"
```

Data Structures

- struct [hw_descq_context](#)
- struct [stm_descq_context](#)
- struct [mbox_msg_hdr](#)
- struct [mbox_msg_fmap](#)
- struct [mbox_msg_csr](#)
- struct [mbox_msg_intr_ctxt](#)
- struct [mbox_msg_qctxt](#)
- struct [mbox_msg](#)
- struct [qdma_mbox](#)

Macros

- #define [MBOX_BASE](#) 0x2400
- #define [MBOX_FN_STATUS](#) 0x0
- #define [S_MBOX_FN_STATUS_IN_MSG](#) 0
- #define [M_MBOX_FN_STATUS_IN_MSG](#) 0x1
- #define [F_MBOX_FN_STATUS_IN_MSG](#) 0x1
- #define [S_MBOX_FN_STATUS_OUT_MSG](#) 1
- #define [M_MBOX_FN_STATUS_OUT_MSG](#) 0x1
- #define [F_MBOX_FN_STATUS_OUT_MSG](#) (1 << [S_MBOX_FN_STATUS_OUT_MSG](#))
- #define [S_MBOX_FN_STATUS_ACK](#) 2 /* PF only, ack status */
- #define [M_MBOX_FN_STATUS_ACK](#) 0x1

- `#define F_MBOX_FN_STATUS_ACK (1 << S_MBOX_FN_STATUS_ACK)`
- `#define S_MBOX_FN_STATUS_SRC 4 /* PF only, source func.*/`
- `#define M_MBOX_FN_STATUS_SRC 0xFFFF`
- `#define G_MBOX_FN_STATUS_SRC(x) (((x) >> S_MBOX_FN_STATUS_SRC) & M_MBOX_FN_STAT←
US_SRC)`
- `#define MBOX_FN_STATUS_MASK`
- `#define MBOX_FN_CMD 0x4`
- `#define S_MBOX_FN_CMD_SND 0`
- `#define M_MBOX_FN_CMD_SND 0x1`
- `#define F_MBOX_FN_CMD_SND (1 << S_MBOX_FN_CMD_SND)`
- `#define S_MBOX_FN_CMD_RCV 1`
- `#define M_MBOX_FN_CMD_RCV 0x1`
- `#define F_MBOX_FN_CMD_RCV (1 << S_MBOX_FN_CMD_RCV)`
- `#define S_MBOX_FN_CMD_VF_RESET 3 /* TBD PF only: reset VF */`
- `#define M_MBOX_FN_CMD_VF_RESET 0x1`
- `#define MBOX_ISR_VEC 0x8`
- `#define S_MBOX_ISR_VEC 0`
- `#define M_MBOX_ISR_VEC 0x1F`
- `#define V_MBOX_ISR_VEC(x) ((x) & M_MBOX_ISR_VEC)`
- `#define MBOX_FN_TARGET 0xC`
- `#define S_MBOX_FN_TARGET_ID 0`
- `#define M_MBOX_FN_TARGET_ID 0xFFFF`
- `#define V_MBOX_FN_TARGET_ID(x) ((x) & M_MBOX_FN_TARGET_ID)`
- `#define MBOX_ISR_EN 0x10`
- `#define S_MBOX_ISR_EN 0`
- `#define M_MBOX_ISR_EN 0x1`
- `#define F_MBOX_ISR_EN 0x1`
- `#define MBOX_PF_ACK_BASE 0x20`
- `#define MBOX_PF_ACK_STEP 4`
- `#define MBOX_PF_ACK_COUNT 8`
- `#define MBOX_IN_MSG_BASE 0x800`
- `#define MBOX_OUT_MSG_BASE 0xc00`
- `#define MBOX_MSG_STEP 4`
- `#define MBOX_MSG_REG_MAX 32`
- `#define MBOX_MSG_OP_PF_MASK 0x10`
- `#define mbox_invalidate_msg(m) { (m)->hdr.op = MBOX_OP_NOOP; }`
- `#define QDMA_MBOX_MSG_TIMEOUT_MS 10000 /* 10 sec*/`
- `#define qdma_mbox_msg_free(m) __qdma_mbox_msg_free(__func__, m)`

Enumerations

- enum [mbox_msg_op](#) {
MBOX_OP_NOOP,
[MBOX_OP_BYE](#),
[MBOX_OP_HELLO](#),
[MBOX_OP_FMAP](#),
[MBOX_OP_CSR](#),
[MBOX_OP_INTR_CTXT](#),
[MBOX_OP_QCTXT_WRT](#),
[MBOX_OP_QCTXT_RD](#),
[MBOX_OP_QCTXT_CLR](#),
[MBOX_OP_QCTXT_INV](#),
[MBOX_OP_QCONF](#),
[MBOX_OP_HELLO_RESP](#) = 0x12,
[MBOX_OP_FMAP_RESP](#),
[MBOX_OP_CSR_RESP](#),
[MBOX_OP_INTR_CTXT_RESP](#),
[MBOX_OP_QCTXT_WRT_RESP](#),
[MBOX_OP_QCTXT_RD_RESP](#),
[MBOX_OP_QCTXT_CLR_RESP](#),
[MBOX_OP_QCTXT_INV_RESP](#),
[MBOX_OP_QCONF_RESP](#),
MBOX_OP_MAX }

Functions

- int [qdma_mbox_init](#) (struct [xlrx_dma_dev](#) *xdev)
- void [qdma_mbox_cleanup](#) (struct [xlrx_dma_dev](#) *xdev)
- void [qdma_mbox_stop](#) (struct [xlrx_dma_dev](#) *xdev)
- void [qdma_mbox_start](#) (struct [xlrx_dma_dev](#) *xdev)
- int [qdma_mbox_msg_send](#) (struct [xlrx_dma_dev](#) *xdev, struct [mbox_msg](#) *m, bool wait_resp, u8 resp_op, unsigned int timeout_ms)
- struct [mbox_msg](#) * [qdma_mbox_msg_alloc](#) (struct [xlrx_dma_dev](#) *xdev, enum [mbox_msg_op](#) op)
- void [__qdma_mbox_msg_free](#) (const char *fname, struct [mbox_msg](#) *m)

4.11.1 Detailed Description

This file contains the declarations for qdma mailbox apis.

4.11.2 Macro Definition Documentation

4.11.2.1 #define MBOX_BASE 0x2400

mailbox registers

4.11.2.2 #define MBOX_FN_STATUS 0x0

mailbox function status

4.11.2.3 **#define S_MBOX_FN_STATUS_IN_MSG 0**

shift value for mailbox function status in msg

4.11.2.4 **#define M_MBOX_FN_STATUS_IN_MSG 0x1**

mask value for mailbox function status in msg

4.11.2.5 **#define F_MBOX_FN_STATUS_IN_MSG 0x1**

face value for mailbox function status in msg

4.11.2.6 **#define S_MBOX_FN_STATUS_OUT_MSG 1**

shift value for out msg

4.11.2.7 **#define M_MBOX_FN_STATUS_OUT_MSG 0x1**

mask value for out msg

4.11.2.8 **#define F_MBOX_FN_STATUS_OUT_MSG (1 << S_MBOX_FN_STATUS_OUT_MSG)**

face value for out msg

4.11.2.9 **#define S_MBOX_FN_STATUS_ACK 2 /* PF only, ack status */**

shift value for status ack

4.11.2.10 **#define M_MBOX_FN_STATUS_ACK 0x1**

mask value for status ack

4.11.2.11 **#define F_MBOX_FN_STATUS_ACK (1 << S_MBOX_FN_STATUS_ACK)**

face value for status ack

4.11.2.12 **#define S_MBOX_FN_STATUS_SRC 4 /* PF only, source func.*/**

shift value for status src

4.11.2.13 `#define M_MBOX_FN_STATUS_SRC 0xFF`

mask value for status src

4.11.2.14 `#define G_MBOX_FN_STATUS_SRC(x) (((x) >> S_MBOX_FN_STATUS_SRC) & M_MBOX_FN_STATUS_SRC)`

face value for status src

4.11.2.15 `#define MBOX_FN_STATUS_MASK`

Value:

```
(F_MBOX_FN_STATUS_IN_MSG | \
 F_MBOX_FN_STATUS_OUT_MSG | \
 F_MBOX_FN_STATUS_ACK)
```

face value for mailbox function status

4.11.2.16 `#define MBOX_FN_CMD 0x4`

mailbox function commands register

4.11.2.17 `#define S_MBOX_FN_CMD_SND 0`

shift value for send command

4.11.2.18 `#define M_MBOX_FN_CMD_SND 0x1`

mask value for send command

4.11.2.19 `#define F_MBOX_FN_CMD_SND (1 << S_MBOX_FN_CMD_SND)`

face value for send command

4.11.2.20 `#define S_MBOX_FN_CMD_RCV 1`

shift value for receive command

4.11.2.21 `#define M_MBOX_FN_CMD_RCV 0x1`

mask value for receive command

4.11.2.22 `#define F_MBOX_FN_CMD_RCV (1 << S_MBOX_FN_CMD_RCV)`

face value for receive command

4.11.2.23 `#define S_MBOX_FN_CMD_VF_RESET 3 /* TBD PF only: reset VF */`

shift value for vf reset

4.11.2.24 `#define M_MBOX_FN_CMD_VF_RESET 0x1`

mask value for vf reset

4.11.2.25 `#define MBOX_ISR_VEC 0x8`

mailbox isr vector register

4.11.2.26 `#define S_MBOX_ISR_VEC 0`

shift value for isr vector

4.11.2.27 `#define M_MBOX_ISR_VEC 0x1F`

mask value for isr vector

4.11.2.28 `#define V_MBOX_ISR_VEC(x) ((x) & M_MBOX_ISR_VEC)`

face value for isr vector

4.11.2.29 `#define MBOX_FN_TARGET 0xC`

mailbox FN target register

4.11.2.30 `#define S_MBOX_FN_TARGET_ID 0`

shift value for FN target id

4.11.2.31 `#define M_MBOX_FN_TARGET_ID 0xFFFF`

mask value for FN target id

4.11.2.32 `#define V_MBOX_FN_TARGET_ID(x) ((x) & M_MBOX_FN_TARGET_ID)`

face value for FN target id

4.11.2.33 `#define MBOX_ISR_EN 0x10`

mailbox isr enable register

4.11.2.34 `#define S_MBOX_ISR_EN 0`

shift value for isr enable

4.11.2.35 `#define M_MBOX_ISR_EN 0x1`

mask value for isr enable

4.11.2.36 `#define F_MBOX_ISR_EN 0x1`

face value for isr enable

4.11.2.37 `#define MBOX_PF_ACK_BASE 0x20`

pf acknowledge base

4.11.2.38 `#define MBOX_PF_ACK_STEP 4`

pf acknowledge step

4.11.2.39 `#define MBOX_PF_ACK_COUNT 8`

pf acknowledge count

4.11.2.40 `#define MBOX_IN_MSG_BASE 0x800`

mailbox incoming msg base

4.11.2.41 `#define MBOX_OUT_MSG_BASE 0xc00`

mailbox outgoing msg base

4.11.2.42 #define MBOX_MSG_STEP 4

mailbox msg step

4.11.2.43 #define MBOX_MSG_REG_MAX 32

mailbox register max

4.11.2.44 #define MBOX_MSG_OP_PF_MASK 0x10

mailbox messages

NOTE: make sure the total message length is ≤ 128 bytes: `mbox_msg_hdr`: 4 bytes body: $\leq (128 - \text{hdr})$ bytes
`mbox_msg_op` - mailbox messages opcode: 1 ~ 0x1F

4.11.3 Enumeration Type Documentation

4.11.3.1 enum mbox_msg_op

Enumerator

MBOX_OP_BYE VF -> PF, request
MBOX_OP_HELLO vf offline
MBOX_OP_FMAP vf online
MBOX_OP_CSR FMAP programming request
MBOX_OP_INTR_CTXT global CSR registers request
MBOX_OP_QCTXT_WRT interrupt context programming
MBOX_OP_QCTXT_RD queue context programming
MBOX_OP_QCTXT_CLR queue context read
MBOX_OP_QCTXT_INV queue context clear
MBOX_OP_QCONF queue context invalidate
MBOX_OP_HELLO_RESP queue context invalidate PF->VF: response
MBOX_OP_FMAP_RESP vf online
MBOX_OP_CSR_RESP FMAP programming
MBOX_OP_INTR_CTXT_RESP global CSR read
MBOX_OP_QCTXT_WRT_RESP interrupt context programming
MBOX_OP_QCTXT_RD_RESP queue context programming
MBOX_OP_QCTXT_CLR_RESP queue context read
MBOX_OP_QCTXT_INV_RESP queue context clear
MBOX_OP_QCONF_RESP queue context invalidate
MBOX_OP_MAX queue context invalidate

4.11.4 Function Documentation

4.11.4.1 int qdma_mbox_init (struct xlnx_dma_dev * xdev)

`qdma_mbox_init()` - initialize qdma mailbox

Parameters

<i>xdev</i>	pointer to xlnx_dma_dev
-------------	---

Returns

0: success
<0: failure

4.11.4.2 void qdma_mbox_cleanup (struct xlnx_dma_dev * *xdev*)

[qdma_mbox_cleanup\(\)](#) - cleanup resources of qdma mailbox [qdma_mbox_stop\(\)](#) - stop mailbox processing [qdma_mbox_start\(\)](#) - start mailbox processing

Parameters

<i>xdev</i>	pointer to xlnx_dma_dev
-------------	---

Returns

none

4.11.4.3 int qdma_mbox_msg_send (struct xlnx_dma_dev * *xdev*, struct mbox_msg * *m*, bool *wait_resp*, u8 *resp_op*, unsigned int *timeout_ms*)

[qdma_mbox_msg_send\(\)](#) - handler to send a mailbox message

Parameters

<i>xdev</i>	pointer to xlnx_dma_dev
<i>m</i>	mailbox message

Returns

0: success
<0: failure

4.11.4.4 struct mbox_msg* qdma_mbox_msg_alloc (struct xlnx_dma_dev * *xdev*, enum mbox_msg_op *op*)

[qdma_mbox_msg_alloc\(\)](#) - allocate a mailbox message

Parameters

<i>xdev</i>	pointer to xlnx_dma_dev
-------------	---

Returns

0: success
 NULL: failure

4.11.4.5 void __qdma_mbox_msg_free (const char * *fname*, struct mbox_msg * *m*)

[__qdma_mbox_msg_free\(\)](#) - free the mailbox message

Parameters

<i>fname</i>	function name
<i>m</i>	mailbox message

Returns

none

4.12 qdma_mod.h File Reference

```
#include <linux/types.h>
#include <linux/pci.h>
#include <linux/workqueue.h>
#include <net/genetlink.h>
#include "libqdma/libqdma_export.h"
#include "libqdma/xdev.h"
#include "cdev.h"
```

Data Structures

- struct [xlnx_qdata](#)
- struct [xlnx_nl_work_q_ctrl](#)
- struct [xlnx_nl_work](#)
- struct [xlnx_pci_dev](#)

Macros

- #define **XNL_EBUFLEN** 256

Functions

- int [xpdev_list_dump](#) (char *buf, int buflen)
- struct [xlnx_pci_dev](#) * [xpdev_find_by_idx](#) (unsigned int idx, char *ebuf, int ebuflen)
- struct [xlnx_qdata](#) * [xpdev_queue_get](#) (struct [xlnx_pci_dev](#) *xpdev, unsigned int qidx, bool c2h, bool check↵_qhndl, char *ebuf, int ebuflen)
- int [xpdev_queue_add](#) (struct [xlnx_pci_dev](#) *xpdev, struct [qdma_queue_conf](#) *qconf, char *ebuf, int ebuflen)
- int [xpdev_queue_delete](#) (struct [xlnx_pci_dev](#) *xpdev, unsigned int qidx, bool c2h, char *ebuf, int ebuflen)
- int [xpdev_nl_queue_start](#) (struct [xlnx_pci_dev](#) *xpdev, void *nl_info, u8 is_qp, u8 is_c2h, unsigned short qidx, unsigned short qcnt)
- unsigned int [qdma_device_read_user_register](#) (struct [xlnx_pci_dev](#) *xpdev, unsigned int reg_addr)
- void [qdma_device_write_user_register](#) (struct [xlnx_pci_dev](#) *xpdev, unsigned int reg_addr, u32 value)
- unsigned int [qdma_device_read_bypass_register](#) (struct [xlnx_pci_dev](#) *xpdev, unsigned int reg_addr)
- void [qdma_device_write_bypass_register](#) (struct [xlnx_pci_dev](#) *xpdev, unsigned int reg_addr, u32 value)

4.12.1 Detailed Description

This file contains the declarations for qdma pcie kernel module.

4.12.2 Function Documentation

4.12.2.1 `int xpdev_list_dump (char * buf, int buflen)`

[xpdev_list_dump\(\)](#) - list the qdma devices

Parameters

in	<i>buflen</i>	buffer length
out	<i>buf</i>	error message buffer, can be NULL/0 (i.e., optional)

Returns

0: success
<0: failure

4.12.2.2 `struct xlnx_pci_dev* xpdev_find_by_idx (unsigned int idx, char * ebuf, int ebuflen)`

[xpdev_find_by_idx\(\)](#) - qdma pcie kernel module api to find the qdma device by index

Parameters

in	<i>idx</i>	qdma device index
in	<i>ebuflen</i>	buffer length
out	<i>ebuf</i>	error message buffer, can be NULL/0 (i.e., optional)

Returns

0: pointer to [xlnx_pci_dev](#)
NULL: failure

4.12.2.3 `struct xlnx_qdata* xpdev_queue_get (struct xlnx_pci_dev * xpdev, unsigned int qidx, bool c2h, bool check_qhndl, char * ebuf, int ebuflen)`

[xpdev_queue_get\(\)](#) - qdma pcie kernel module api to get a queue information

Parameters

in	<i>xpdev</i>	pointer to xlnx_pci_dev
in	<i>qidx</i>	queue index
in	<i>c2h</i>	flag to indicate the queue direction (c2h/h2c)
in	<i>check_qhndl</i>	flag for validating the data
in	<i>ebuflen</i>	buffer length
out	<i>ebuf</i>	error message buffer, can be NULL/0 (i.e., optional)

Returns

0: queue information
 NULL: failure

4.12.2.4 `int xpdev_queue_add (struct xlnx_pci_dev * xpdev, struct qdma_queue_conf * qconf, char * ebuf, int ebuflen)`

[xpdev_queue_add\(\)](#) - qdma pcie kernel module api to add a queue

Parameters

in	<i>xpdev</i>	pointer to xlnx_pci_dev
in	<i>qconf</i>	queue configuration
in	<i>ebuflen</i>	buffer length
out	<i>ebuf</i>	error message buffer, can be NULL/0 (i.e., optional)

Returns

0: success
 <0: failure

4.12.2.5 `int xpdev_queue_delete (struct xlnx_pci_dev * xpdev, unsigned int qidx, bool c2h, char * ebuf, int ebuflen)`

[xpdev_queue_delete\(\)](#) - qdma pcie kernel module api to delete a queue

Parameters

in	<i>xpdev</i>	pointer to xlnx_pci_dev
in	<i>qidx</i>	queue index
in	<i>c2h</i>	flag to indicate the queue direction (c2h/h2c)
in	<i>ebuflen</i>	buffer length
out	<i>ebuf</i>	error message buffer, can be NULL/0 (i.e., optional)

Returns

0: success
 <0: failure

4.12.2.6 `unsigned int qdma_device_read_user_register (struct xlnx_pci_dev * xpdev, unsigned int reg_addr)`

[qdma_device_read_user_register\(\)](#) - read user bar register

Parameters

in	<i>xpdev</i>	pointer to xlnx_pci_dev
in	<i>reg_addr</i>	register address

Returns

value of the user bar register

4.12.2.7 void qdma_device_write_user_register (struct xlnx_pci_dev * *xpdev*, unsigned int *reg_addr*, u32 *value*)

[qdma_device_write_user_register\(\)](#) - write user bar register

Parameters

in	<i>xpdev</i>	pointer to xlnx_pci_dev
in	<i>reg_addr</i>	register address
in	<i>value</i>	register value to be written

4.12.2.8 unsigned int qdma_device_read_bypass_register (struct xlnx_pci_dev * *xpdev*, unsigned int *reg_addr*)

[qdma_device_read_bypass_register\(\)](#) - read bypass bar register

Parameters

in	<i>xpdev</i>	pointer to xlnx_pci_dev
in	<i>reg_addr</i>	register address

Returns

value of the bypass bar register

4.12.2.9 void qdma_device_write_bypass_register (struct xlnx_pci_dev * *xpdev*, unsigned int *reg_addr*, u32 *value*)

[qdma_device_write_bypass_register\(\)](#) - write bypass bar register

Parameters

in	<i>xpdev</i>	pointer to xlnx_pci_dev
in	<i>reg_addr</i>	register address
in	<i>value</i>	register value to be written

4.13 qdma_nl.h File Reference

Macros

- #define [XNL_NAME_PF](#) "xnl_pf"
- #define [XNL_NAME_VF](#) "xnl_vf"

- `#define XNL_VERSION 0x1`
- `#define XNL_RESP_BUFLLEN_MIN 256`
- `#define XNL_RESP_BUFLLEN_MAX (2048 * 6)`
- `#define XNL_ERR_BUFLLEN 64`
- `#define XNL_STR_LEN_MAX 20`
- `#define XNL_QIDX_INVALID 0xFFFF`
- `#define XNL_F_QMODE_ST 0x00000001`
- `#define XNL_F_QMODE_MM 0x00000002`
- `#define XNL_F_QDIR_H2C 0x00000004`
- `#define XNL_F_QDIR_C2H 0x00000008`
- `#define XNL_F_QDIR_BOTH (XNL_F_QDIR_H2C | XNL_F_QDIR_C2H)`
- `#define XNL_F_PFETCH_EN 0x00000010`
- `#define XNL_F_DESC_BYPASS_EN 0x00000020`
- `#define XNL_F_FETCH_CREDIT 0x00000040`
- `#define XNL_F_CMPL_STATUS_ACC_EN 0x00000080`
- `#define XNL_F_CMPL_STATUS_EN 0x00000100`
- `#define XNL_F_CMPL_STATUS_PEND_CHK 0x00000200`
- `#define XNL_F_CMPL_STATUS_DESC_EN 0x00000400`
- `#define XNL_F_C2H_CMPL_INTR_EN 0x00000800`
- `#define XNL_F_CMPL_UDD_EN 0x00001000`
- `#define XNL_F_PFETCH_BYPASS_EN 0x00002000`
- `#define XNL_F_CMPT_OVF_CHK_DIS 0x00004000`
- `#define MAX_QFLAGS 15`
- `#define QDMA_MAX_INT_RING_ENTRIES 512`

Enumerations

- enum `xnl_attr_t` {
 - `XNL_ATTR_GENMSG,`
 - `XNL_ATTR_DRV_INFO,`
 - `XNL_ATTR_DEV_IDX,`
 - `XNL_ATTR_PCI_BUS,`
 - `XNL_ATTR_PCI_DEV,`
 - `XNL_ATTR_PCI_FUNC,`
 - `XNL_ATTR_DEV_STAT_MMH2C_PKTS1,`
 - `XNL_ATTR_DEV_STAT_MMH2C_PKTS2,`
 - `XNL_ATTR_DEV_STAT_MMC2H_PKTS1,`
 - `XNL_ATTR_DEV_STAT_MMC2H_PKTS2,`
 - `XNL_ATTR_DEV_STAT_STH2C_PKTS1,`
 - `XNL_ATTR_DEV_STAT_STH2C_PKTS2,`
 - `XNL_ATTR_DEV_STAT_STC2H_PKTS1,`
 - `XNL_ATTR_DEV_STAT_STC2H_PKTS2,`
 - `XNL_ATTR_DEV_CFG_BAR,`
 - `XNL_ATTR_DEV_USR_BAR,`
 - `XNL_ATTR_DEV_QSET_MAX,`
 - `XNL_ATTR_DEV_QSET_QBASE,`
 - `XNL_ATTR_REG_BAR_NUM,`
 - `XNL_ATTR_REG_ADDR,`
 - `XNL_ATTR_REG_VAL,`
 - `XNL_ATTR_QIDX,`
 - `XNL_ATTR_NUM_Q,`
 - `XNL_ATTR_QFLAG,`
 - `XNL_ATTR_CMPT_DESC_SIZE,`
 - `XNL_ATTR_SW_DESC_SIZE,`
 - `XNL_ATTR_QRNGSZ_IDX,`
 - `XNL_ATTR_C2H_BUFSZ_IDX,`
 - `XNL_ATTR_CMPT_TIMER_IDX,`
 - `XNL_ATTR_CMPT_CNTR_IDX,`
 - `XNL_ATTR_CMPT_TRIG_MODE,`
 - `XNL_ATTR_RANGE_START,`
 - `XNL_ATTR_RANGE_END,`
 - `XNL_ATTR_INTR_VECTOR_IDX,`
 - `XNL_ATTR_INTR_VECTOR_START_IDX,`
 - `XNL_ATTR_INTR_VECTOR_END_IDX,`
 - `XNL_ATTR_RSP_BUF_LEN,`
 - `XNL_ATTR_GLOBAL_CSR,`
 - `XNL_ATTR_PIPE_GL_MAX,`
 - `XNL_ATTR_PIPE_FLOW_ID,`
 - `XNL_ATTR_PIPE_SLR_ID,`
 - `XNL_ATTR_PIPE_TDEST,`
 - `XNL_ATTR_DEV_STM_BAR,`
 - `XNL_ATTR_MAX`** }
- enum `xnl_st_c2h_cmpt_desc_size` {
 - `XNL_ST_C2H_CMPT_DESC_SIZE_8B,`
 - `XNL_ST_C2H_CMPT_DESC_SIZE_16B,`
 - `XNL_ST_C2H_CMPT_DESC_SIZE_32B,`
 - `XNL_ST_C2H_CMPT_DESC_SIZE_64B,`
 - `XNL_ST_C2H_NUM_CMPT_DESC_SIZES` }
- enum `xnl_qdma_rngsz_idx` {

```
XNL_QDMA_RNGSZ_64_IDX,  
XNL_QDMA_RNGSZ_128_IDX,  
XNL_QDMA_RNGSZ_256_IDX,  
XNL_QDMA_RNGSZ_512_IDX,  
XNL_QDMA_RNGSZ_1024_IDX,  
XNL_QDMA_RNGSZ_2048_IDX,  
XNL_QDMA_RNGSZ_4096_IDX,  
XNL_QDMA_RNGSZ_8192_IDX,  
XNL_QDMA_RNGSZ_16384_IDX,  
XNL_QDMA_RNGSZ_32768_IDX,  
XNL_QDMA_RNGSZ_65536_IDX,  
XNL_QDMA_RNGSZ_131072_IDX,  
XNL_QDMA_RNGSZ_262177_IDX,  
XNL_QDMA_RNGSZ_524288_IDX,  
XNL_QDMA_RNGSZ_1048576_IDX,  
XNL_QDMA_RNGSZ_2097152_IDX,  
XNL_QDMA_RNGSZ_IDXS }
```

- enum [xnl_op_t](#) {
 [XNL_CMD_DEV_LIST](#),
 [XNL_CMD_DEV_INFO](#),
 [XNL_CMD_DEV_STAT](#),
 [XNL_CMD_DEV_STAT_CLEAR](#),
 [XNL_CMD_REG_DUMP](#),
 [XNL_CMD_REG_RD](#),
 [XNL_CMD_REG_WRT](#),
 [XNL_CMD_Q_LIST](#),
 [XNL_CMD_Q_ADD](#),
 [XNL_CMD_Q_START](#),
 [XNL_CMD_Q_STOP](#),
 [XNL_CMD_Q_DEL](#),
 [XNL_CMD_Q_DUMP](#),
 [XNL_CMD_Q_DESC](#),
 [XNL_CMD_Q_CMPT](#),
 [XNL_CMD_Q_RX_PKT](#),
 [XNL_CMD_INTR_RING_DUMP](#),
 [XNL_CMD_Q_UDD](#),
 [XNL_CMD_GLOBAL_CSR](#),
 [XNL_CMD_VERSION](#),
 [XNL_CMD_MAX](#) }

- enum [qdma_err_idx](#) {

```

err_ram_sbe,
err_ram_dbe,
err_dsc,
err_trq,
err_h2c_mm_0,
err_h2c_mm_1,
err_c2h_mm_0,
err_c2h_mm_1,
err_c2h_st,
ind_ctxt_cmd_err,
err_bdg,
err_h2c_st,
poison,
ur_ca,
param,
addr,
tag,
flr,
timeout,
dat_poison,
flr_cancel,
dma,
dsc,
rq_cancel,
dbe,
sbe,
unmapped,
qid_range,
vf_access_err,
tcp_timeout,
mty_mismatch,
len_mismatch,
qid_mismatch,
desc_rsp_err,
eng_wpl_data_par_err,
msi_int_fail,
err_desc_cnt,
portid_ctxt_mismatch,
portid_byp_in_mismatch,
cmpt_inv_q_err,
cmpt_qfull_err,
cmpt_cidx_err,
cmpt_ppty_err,
fatal_mty_mismatch,
fatal_len_mismatch,
fatal_qid_mismatch,
timer_fifo_ram_rdbe,
fatal_eng_wpl_data_par_err,
pfch_ll_ram_rdbe,
cmpt_ctxt_ram_rdbe,
pfch_ctxt_ram_rdbe,
desc_req_fifo_ram_rdbe,
int_ctxt_ram_rdbe,
cmpt_coal_data_ram_rdbe,
tuser_fifo_ram_rdbe,
qid_fifo_ram_rdbe,
payload_fifo_ram_rdbe,
wpl_data_par_err,
zero_len_desc_err,
csi_mop_err,
no_err,
err_desc_err,
sb_mi_h2c0_dat,
sb_mi_c2h0_dat,
sb_h2c_rd_brg_dat,
sb_h2c_wrr_brg_dat,

```

```
qdma_errs }
```

4.13.1 Detailed Description

This file contains the declarations for qdma netlink interfaces.

4.13.2 Macro Definition Documentation

4.13.2.1 `#define XNL_NAME_PF "xnl_pf"`

physical function name (no more than 15 characters)

4.13.2.2 `#define XNL_NAME_VF "xnl_vf"`

virtual function name

4.13.2.3 `#define XNL_VERSION 0x1`

qdma netlink interface version number

4.13.2.4 `#define XNL_RESP_BUFLLEN_MIN 256`

qdma nl interface minimum response buffer length

4.13.2.5 `#define XNL_RESP_BUFLLEN_MAX (2048 * 6)`

qdma nl interface maximum response buffer length

4.13.2.6 `#define XNL_ERR_BUFLLEN 64`

qdma nl interface error buffer length

4.13.2.7 `#define XNL_STR_LEN_MAX 20`

qdma nl command parameter length

4.13.2.8 `#define XNL_QIDX_INVALID 0xFFFF`

Q parameter: value to indicate invalid qid

4.13.2.9 `#define XNL_F_QMODE_ST 0x00000001`

Q parameter: streaming mode

4.13.2.10 `#define XNL_F_QMODE_MM 0x00000002`

Q parameter: memory management mode

4.13.2.11 `#define XNL_F_QDIR_H2C 0x00000004`

Q parameter: queue in h2c direction

4.13.2.12 `#define XNL_F_QDIR_C2H 0x00000008`

Q parameter: queue in c2h direction

4.13.2.13 `#define XNL_F_QDIR_BOTH (XNL_F_QDIR_H2C | XNL_F_QDIR_C2H)`

Q parameter: queue in both directions

4.13.2.14 `#define XNL_F_PFETCH_EN 0x00000010`

Q parameter: queue in prefetch mode

4.13.2.15 `#define XNL_F_DESC_BYPASS_EN 0x00000020`

Q parameter: enable the bypass for the queue

4.13.2.16 `#define XNL_F_FETCH_CREDIT 0x00000040`

Q parameter: fetch credits

4.13.2.17 `#define XNL_F_CMPL_STATUS_ACC_EN 0x00000080`

Q parameter: enable writeback accumulation

4.13.2.18 `#define XNL_F_CMPL_STATUS_EN 0x00000100`

Q parameter: enable writeback

4.13.2.19 `#define XNL_F_CMPL_STATUS_PEND_CHK 0x00000200`

Q parameter: enable writeback pending check

4.13.2.20 `#define XNL_F_CMPL_STATUS_DESC_EN 0x00000400`

Q parameter: enable writeback status descriptor

4.13.2.21 `#define XNL_F_C2H_CMPL_INTR_EN 0x00000800`

Q parameter: enable queue completion interrupt

4.13.2.22 `#define XNL_F_CMPL_UDD_EN 0x00001000`

Q parameter: enable user defined data

4.13.2.23 `#define XNL_F_PFETCH_BYPASS_EN 0x00002000`

Q parameter: enable the pfetch bypass for the queue

4.13.2.24 `#define XNL_F_CMPT_OVF_CHK_DIS 0x00004000`

Q parameter: disable CMPT overflow check

4.13.2.25 `#define MAX_QFLAGS 15`

maximum number of queue flags to control queue configuration

4.13.2.26 `#define QDMA_MAX_INT_RING_ENTRIES 512`

maximum number of interrupt ring entries

4.13.3 Enumeration Type Documentation

4.13.3.1 enum xnl_attr_t

xnl_attr_t netlink attributes for qdma(variables): the index in this enum is used as a reference for the type, userspace application has to indicate the corresponding type the policy is used for security considerations

Enumerator

XNL_ATTR_GENMSG generatl message
XNL_ATTR_DRV_INFO device info
XNL_ATTR_DEV_IDX device index
XNL_ATTR_PCI_BUS pci bus number
XNL_ATTR_PCI_DEV pci device number
XNL_ATTR_PCI_FUNC pci function id
XNL_ATTR_DEV_STAT_MMH2C_PKTS1 number of MM H2C packkts
XNL_ATTR_DEV_STAT_MMH2C_PKTS2 number of MM H2C packkts
XNL_ATTR_DEV_STAT_MMC2H_PKTS1 number of MM C2H packkts
XNL_ATTR_DEV_STAT_MMC2H_PKTS2 number of MM C2H packkts
XNL_ATTR_DEV_STAT_STH2C_PKTS1 number of ST H2C packkts
XNL_ATTR_DEV_STAT_STH2C_PKTS2 number of ST H2C packkts
XNL_ATTR_DEV_STAT_STC2H_PKTS1 number of ST C2H packkts
XNL_ATTR_DEV_STAT_STC2H_PKTS2 number of ST C2H packkts
XNL_ATTR_DEV_CFG_BAR device config bar number
XNL_ATTR_DEV_USR_BAR device user bar number
XNL_ATTR_DEV_QSET_MAX max queue sets
XNL_ATTR_DEV_QSET_QBASE queue base start
XNL_ATTR_REG_BAR_NUM register bar number
XNL_ATTR_REG_ADDR register address
XNL_ATTR_REG_VAL register value
XNL_ATTR_QIDX queue index
XNL_ATTR_NUM_Q number of queues
XNL_ATTR_QFLAG queue config flags
XNL_ATTR_CMPT_DESC_SIZE completion descriptor size
XNL_ATTR_SW_DESC_SIZE software descriptor size
XNL_ATTR_QRNGSZ_IDX queue ring index
XNL_ATTR_C2H_BUFSZ_IDX c2h buffer index
XNL_ATTR_CMPT_TIMER_IDX completion timer index
XNL_ATTR_CMPT_CNTR_IDX completion counter index
XNL_ATTR_CMPT_TRIG_MODE completion trigger mode
XNL_ATTR_RANGE_START range start
XNL_ATTR_RANGE_END range end
XNL_ATTR_INTR_VECTOR_IDX interrupt vector index
XNL_ATTR_INTR_VECTOR_START_IDX interrupt vector start index
XNL_ATTR_INTR_VECTOR_END_IDX interrupt vector end index
XNL_ATTR_RSP_BUF_LEN response buffer length
XNL_ATTR_GLOBAL_CSR global csr data
XNL_ATTR_PIPE_GL_MAX max no. of gl for pipe
XNL_ATTR_PIPE_FLOW_ID pipe flow id
XNL_ATTR_PIPE_SLR_ID pipe slr id
XNL_ATTR_PIPE_TDEST pipe tdest
XNL_ATTR_DEV_STM_BAR device STM bar number

4.13.3.2 enum xnl_st_c2h_cmpt_desc_size

xnl_st_c2h_cmpt_desc_size c2h writeback descriptor sizes

Enumerator

XNL_ST_C2H_CMPT_DESC_SIZE_8B 8B descriptor
XNL_ST_C2H_CMPT_DESC_SIZE_16B 16B descriptor
XNL_ST_C2H_CMPT_DESC_SIZE_32B 32B descriptor
XNL_ST_C2H_CMPT_DESC_SIZE_64B 64B descriptor
XNL_ST_C2H_NUM_CMPT_DESC_SIZES Num of desc sizes

4.13.3.3 enum xnl_op_t

xnl_op_t - XNL command types

Enumerator

XNL_CMD_DEV_LIST list all the qdma devices
XNL_CMD_DEV_INFO dump the device information
XNL_CMD_DEV_STAT dump the device statistics
XNL_CMD_DEV_STAT_CLEAR reset the device statistics
XNL_CMD_REG_DUMP dump the register information
XNL_CMD_REG_RD read a register value
XNL_CMD_REG_WRT write value to a register
XNL_CMD_Q_LIST list all the queue present in the system
XNL_CMD_Q_ADD add a queue
XNL_CMD_Q_START start a queue
XNL_CMD_Q_STOP stop a queue
XNL_CMD_Q_DEL delete a queue
XNL_CMD_Q_DUMP dump queue information
XNL_CMD_Q_DESC dump descriptor information
XNL_CMD_Q_CMPT dump writeback descriptor information
XNL_CMD_Q_RX_PKT dump packet information
XNL_CMD_INTR_RING_DUMP dump interrupt ring information
XNL_CMD_Q_UDD dump the user defined data
XNL_CMD_GLOBAL_CSR get all global csr register values
XNL_CMD_VERSION list RTL version and Vivado release ID
XNL_CMD_MAX max number of XNL commands

4.13.3.4 enum qdma_err_idx

qdma_err_idx - Induce error

4.14 qdma_st_c2h.h File Reference

```
#include <linux/spinlock_types.h>
#include <linux/types.h>
#include "qdma_descq.h"
```

Data Structures

- struct [qdma_sdesc_info](#)
- struct [qdma_flq](#)

Functions

- int [qdma_descq_rxq_read](#) (struct [qdma_descq](#) *descq, struct [qdma_request](#) *req)
- int [qdma_descq_dump_cmpt](#) (struct [qdma_descq](#) *descq, int start, int end, char *buf, int buflen)
- void [incr_cmpl_desc_cnt](#) (struct [qdma_descq](#) *descq, unsigned int cnt)
- void [descq_flq_free_resource](#) (struct [qdma_descq](#) *descq)
- int [descq_flq_alloc_resource](#) (struct [qdma_descq](#) *descq)
- int [descq_process_completion_st_c2h](#) (struct [qdma_descq](#) *descq, int budget, bool upd_cmpl)
- int [descq_st_c2h_read](#) (struct [qdma_descq](#) *descq, struct [qdma_request](#) *req, bool update, bool refill)

4.14.1 Detailed Description

This file contains the declarations for qdma st c2h processing.

4.14.2 Function Documentation

4.14.2.1 int [qdma_descq_rxq_read](#) (struct [qdma_descq](#) * *descq*, struct [qdma_request](#) * *req*)

[qdma_descq_rxq_read\(\)](#) - read from the rx queue

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>req</i>	queue request

Returns

0: success
<0: failure

4.14.2.2 int [qdma_descq_dump_cmpt](#) (struct [qdma_descq](#) * *descq*, int *start*, int *end*, char * *buf*, int *buflen*)

[qdma_descq_dump_cmpt\(\)](#) - dump the completion queue descriptors

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>start</i>	start completion descriptor index
in	<i>end</i>	end completion descriptor index
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

0: success
 <0: failure

4.14.2.3 void [incr_cmpl_desc_cnt](#) (struct [qdma_descq](#) * *descq*, unsigned int *cnt*)

[incr_cmpl_desc_cnt\(\)](#) - update the interrupt cidx

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>cnt</i>	increment value

4.14.2.4 void [descq_flq_free_resource](#) (struct [qdma_descq](#) * *descq*)

[descq_flq_free_resource\(\)](#) - handler to free the pages for the request

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

none

4.14.2.5 int [descq_flq_alloc_resource](#) (struct [qdma_descq](#) * *descq*)

[descq_flq_alloc_resource\(\)](#) - handler to allocate the pages for the request

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

0: success
 <0: failure

4.14.2.6 `int descq_process_completion_st_c2h (struct qdma_descq * descq, int budget, bool upd_cmpl)`

[descq_process_completion_st_c2h\(\)](#) - handler to process the st c2h completion request

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>budget</i>	number of descriptors to process
in	<i>upd_cmpl</i>	if update completion required

Returns

0: success
 <0: failure

4.14.2.7 `int descq_st_c2h_read (struct qdma_descq * descq, struct qdma_request * req, bool update, bool refill)`

[descq_st_c2h_read\(\)](#) - handler to process the st c2h read request

Parameters

in	<i>descq</i>	pointer to qdma_descq
in	<i>req</i>	pointer to read request
in	<i>update</i>	flag to update the request
in	<i>refill</i>	flag to indicate whether to refill the flq

Returns

0: success
 <0: failure

4.15 qdma_thread.h File Reference

Functions

- `int qdma_threads_create` (unsigned int num_threads)
- `void qdma_threads_destroy` (void)
- `void qdma_thread_remove_work` (struct [qdma_descq](#) *descq)
- `void qdma_thread_add_work` (struct [qdma_descq](#) *descq)

4.15.1 Detailed Description

This file contains the declarations for qdma thread handlers.

4.15.2 Function Documentation

4.15.2.1 `int qdma_threads_create (unsigned int num_threads)`

[qdma_threads_create\(\)](#) - create qdma threads This functions creates two threads for each cpu in the system or number of number of thread requested by param *num_threads* and assigns the thread handlers 1: queue processing thread 2: queue completion handler thread

Parameters

in	<i>num_threads</i>	- number of threads to be created
----	--------------------	-----------------------------------

Returns

0: success
<0: failure

4.15.2.2 `void qdma_threads_destroy (void)`

[qdma_threads_destroy\(\)](#) - destroy all the qdma threads created during system initialization

Returns

none

4.15.2.3 `void qdma_thread_remove_work (struct qdma_descq * descq)`

[qdma_thread_remove_work\(\)](#) - handler to remove the attached work thread

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

none

4.15.2.4 `void qdma_thread_add_work (struct qdma_descq * descq)`

[qdma_thread_add_work\(\)](#) - handler to add a work thread

Parameters

in	<i>descq</i>	pointer to qdma_descq
----	--------------	---------------------------------------

Returns

none

4.16 thread.h File Reference

```
#include <linux/version.h>
#include <linux/spinlock.h>
#include <linux/kthread.h>
#include <linux/cpuset.h>
#include <linux/signal.h>
#include "qdma_compat.h"
```

Data Structures

- struct [qdma_kthread](#)

Macros

- #define [lock_thread](#)(thp) spin_lock(&(thp)->lock)
- #define [unlock_thread](#)(thp) spin_unlock(&(thp)->lock)
- #define [qdma_kthread_wakeup](#)(thp)
- #define [pr_debug_thread](#)(fmt, ...)

Functions

- int [qdma_kthread_dump](#) (struct [qdma_kthread](#) *thp, char *buf, int buflen, int detail)
- int [qdma_kthread_start](#) (struct [qdma_kthread](#) *thp, char *name, int id)
- int [qdma_kthread_stop](#) (struct [qdma_kthread](#) *thp)

4.16.1 Detailed Description

This file contains the declarations for qdma kernel threads.

4.16.2 Macro Definition Documentation

4.16.2.1 #define lock_thread(thp) spin_lock(&(thp)->lock)

lock thread macro

4.16.2.2 `#define unlock_thread(thp) spin_unlock(&(thp)->lock)`

un lock thread macro

4.16.2.3 `#define qdma_kthread_wakeup(thp)`

Value:

```
do { \
    thp->schedule = 1; \
    qdma_waitq_wakeup(&thp->waitq); \
} while (0)
```

macro to wake up the qdma k thread

4.16.2.4 `#define pr_debug_thread(fmt, ...)`

pr_debug_thread

4.16.3 Function Documentation

4.16.3.1 `int qdma_kthread_dump (struct qdma_kthread * thp, char * buf, int buflen, int detail)`

[qdma_kthread_dump\(\)](#) - handler to dump the thread information

Parameters

in	<i>thp</i>	pointer to qdma_kthread
in	<i>detail</i>	flag to indicate whether details required or not
in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

length of the buffer

4.16.3.2 `int qdma_kthread_start (struct qdma_kthread * thp, char * name, int id)`

[qdma_kthread_start\(\)](#) - handler to start the kernel thread

Parameters

in	<i>thp</i>	pointer to qdma_kthread
in	<i>name</i>	name for the thread
in	<i>id</i>	thread id

Returns

0: success
 <0: failure

4.16.3.3 `int qdma_kthread_stop (struct qdma_kthread * thp)`

[qdma_kthread_stop\(\)](#) - handler to stop the kernel thread

Parameters

in	<i>thp</i>	pointer to qdma_kthread
----	------------	---

Returns

0: success
 <0: failure

4.17 xdev.h File Reference

```
#include <linux/types.h>
#include <linux/dma-mapping.h>
#include <linux/interrupt.h>
#include <linux/pci.h>
#include "libqdma_export.h"
#include "qdma_mbox.h"
#include "qdma_qconf_mgr.h"
```

Data Structures

- struct [intr_coal_conf](#)
- struct [intr_vec_map_type](#)
- struct [intr_info_t](#)
- struct [xlrx_dma_dev](#)

Macros

- #define [QDMA_BAR_NUM](#) 6
- #define [QDMA_MAX_BAR_LEN_MAPPED](#) 0x4000000
- #define [BUS_NUM_MASK](#) 0xFFFF0000
- #define [BUS_NUM_SHIFT](#) 16
- #define [PF_DEV_0_MASK](#) 0x0000F000
- #define [PF_DEV_0_SHIFT](#) 12
- #define [PF_DEV_1_MASK](#) 0x00000F00
- #define [PF_DEV_1_SHIFT](#) 8
- #define [PF_DEV_2_MASK](#) 0x000000F0
- #define [PF_DEV_2_SHIFT](#) 4
- #define [PF_DEV_3_MASK](#) 0x0000000F

- `#define PF_DEV_3_SHIFT 0`
- `#define VF_PF_IDENTIFIER_MASK 0xF`
- `#define VF_PF_IDENTIFIER_SHIFT 8`
- `#define QDMA_DESC_BLEN_BITS 28`
- `#define QDMA_DESC_BLEN_MAX ((1 << (QDMA_DESC_BLEN_BITS)) - 1)`
- `#define PCI_DMA_H(addr) ((addr >> 16) >> 16)`
- `#define PCI_DMA_L(addr) (addr & 0xffffffffUL)`
- `#define XDEV_FLAG_OFFLINE 0x1`
- `#define XDEV_FLAG_IRQ 0x2`
- `#define XDEV_NUM_IRQ_MAX 8`
- `#define RTL1_VERSION 0`
- `#define RTL2_VERSION 1`
- `#define VIVADO_RELEASE_2018_3 0`
- `#define VIVADO_RELEASE_2018_2 1`
- `#define EVEREST_SOFT_IP 0`
- `#define EVEREST_HARD_IP 1`
- `#define xdev_sriov_disable(xdev)`
- `#define xdev_sriov_enable(xdev, num_vfs)`
- `#define xdev_sriov_vf_offline(xdev, func_id)`
- `#define xdev_sriov_vf_online(xdev, func_id)`

Typedefs

- `typedef irqreturn_t(* f_intr_handler) (int irq_index, int irq, void *dev_id)`

Enumerations

- `enum qdma_pf_devices {`
`PF_DEVICE_0 = 0,`
`PF_DEVICE_1,`
`PF_DEVICE_2,`
`PF_DEVICE_3 }`
- `enum intr_type_list {`
`INTR_TYPE_ERROR,`
`INTR_TYPE_USER,`
`INTR_TYPE_DATA,`
`INTR_TYPE_MAX }`

Functions

- `struct xlnx_dma_dev * xdev_find_by_pdev (struct pci_dev *pdev)`
- `struct xlnx_dma_dev * xdev_find_by_idx (int idx)`
- `struct xlnx_dma_dev * xdev_list_first (void)`
- `struct xlnx_dma_dev * xdev_list_next (struct xlnx_dma_dev *xdev)`
- `int xdev_list_dump (char *buf, int buflen)`
- `int xdev_check_hdl (const char *f, struct pci_dev *pdev, unsigned long hndl)`

4.17.1 Detailed Description

This file contains the declarations for QDMA PCIe device.

4.17.2 Macro Definition Documentation

4.17.2.1 `#define QDMA_BAR_NUM 6`

QDMA bars

4.17.2.2 `#define QDMA_MAX_BAR_LEN_MAPPED 0x4000000`

QDMA config bar size - 64MB

4.17.2.3 `#define QDMA_DESC_BLEN_BITS 28`

number of bits to describe the DMA transfer descriptor

4.17.2.4 `#define QDMA_DESC_BLEN_MAX ((1 << (QDMA_DESC_BLEN_BITS)) - 1)`

maximum size of a single DMA transfer descriptor

4.17.2.5 `#define PCI_DMA_H(addr) ((addr >> 16) >> 16)`

obtain the 32 most significant (high) bits of a 32-bit or 64-bit address

4.17.2.6 `#define PCI_DMA_L(addr) (addr & 0xffffffffUL)`

obtain the 32 least significant (low) bits of a 32-bit or 64-bit address

4.17.2.7 `#define XDEV_FLAG_OFFLINE 0x1`

Flag for device offline

4.17.2.8 `#define XDEV_FLAG_IRQ 0x2`

Flag for IRQ

4.17.2.9 `#define XDEV_NUM_IRQ_MAX 8`

Maximum number of interrupts supported per device

4.17.2.10 `#define RTL1_VERSION 0`

Macros for Hardware Version info

4.17.2.11 `#define xdev_sriov_disable(xdev)`

dummy declaration for [xdev_sriov_disable\(\)](#) When virtual function is not enabled

4.17.2.12 `#define xdev_sriov_enable(xdev, num_vfs)`

dummy declaration for [xdev_sriov_enable\(\)](#) When virtual function is not enabled

4.17.2.13 `#define xdev_sriov_vf_offline(xdev, func_id)`

dummy declaration for [xdev_sriov_vf_offline\(\)](#) When virtual function is not enabled

4.17.2.14 `#define xdev_sriov_vf_online(xdev, func_id)`

dummy declaration for [xdev_sriov_vf_online\(\)](#) When virtual function is not enabled

4.17.3 Typedef Documentation

4.17.3.1 `typedef irqreturn_t(* f_intr_handler)(int irq_index, int irq, void *dev_id)`

interrupt call back function handlers

4.17.4 Enumeration Type Documentation

4.17.4.1 `enum intr_type_list`

`intr_type_list` - interrupt types

Enumerator

`INTR_TYPE_ERROR` error interrupt
`INTR_TYPE_USER` user interrupt
`INTR_TYPE_DATA` data interrupt
`INTR_TYPE_MAX` max interrupt

4.17.5 Function Documentation

4.17.5.1 `struct xlnx_dma_dev* xdev_find_by_pdev (struct pci_dev *pdev)`

[xdev_find_by_pdev\(\)](#) - find the xdev using struct pci_dev

Parameters

in	<i>pdev</i>	pointer to struct pci_dev
----	-------------	---------------------------

Returns

pointer to [xlnx_dma_dev](#) on success
NULL on failure

4.17.5.2 struct xlnx_dma_dev* xdev_find_by_idx (int idx)

[xdev_find_by_idx\(\)](#) - find the xdev using the index value

Parameters

in	<i>idx</i>	index value in the xdev list
----	------------	------------------------------

Returns

pointer to [xlnx_dma_dev](#) on success
NULL on failure

4.17.5.3 struct xlnx_dma_dev* xdev_list_first (void)

[xdev_list_first\(\)](#) - handler to return the first xdev entry from the list

Returns

pointer to first [xlnx_dma_dev](#) on success
NULL on failure

4.17.5.4 struct xlnx_dma_dev* xdev_list_next (struct xlnx_dma_dev * xdev)

[xdev_list_next\(\)](#) - handler to return the next xdev entry from the list

Parameters

in	<i>xdev</i>	pointer to current xdev
----	-------------	-------------------------

Returns

pointer to next [xlnx_dma_dev](#) on success
NULL on failure

4.17.5.5 `int xdev_list_dump (char * buf, int buflen)`

[xdev_list_dump\(\)](#) - list the dma device details

Parameters

in	<i>buflen</i>	length of the input buffer
out	<i>buf</i>	message buffer

Returns

pointer to next [xlnx_dma_dev](#) on success
NULL on failure

4.17.5.6 `int xdev_check_hndl (const char * f, struct pci_dev * pdev, unsigned long hndl)`

[xdev_check_hndl\(\)](#) - helper function to validate the device handle

Parameters

in	<i>f</i>	device name
in	<i>pdev</i>	pointer to struct pci_dev
in	<i>hndl</i>	device handle

Returns

0: success
EINVAL: on failure

Index

- [__qdma_mbox_msg_free](#)
 - [qdma_mbox.h](#), [130](#)
- AUTO_MODE
 - [libqdma_export.h](#), [81](#)
- [alloc_fail](#)
 - [qdma_flq](#), [38](#)
- [array](#)
 - [qdma_csr_info](#), [25](#)
- [at](#)
 - [qdma_queue_conf](#), [51](#)
- [avail](#)
 - [qdma_descq](#), [30](#)
 - [qdma_flq](#), [38](#)
- [bar_num_bypass](#)
 - [qdma_dev_conf](#), [36](#)
- [bar_num_config](#)
 - [qdma_dev_conf](#), [36](#)
- [bar_num_stm](#)
 - [qdma_dev_conf](#), [36](#)
- [bar_num_user](#)
 - [qdma_dev_conf](#), [36](#)
- [bdf](#)
 - [qdma_dev_conf](#), [36](#)
- [buf](#)
 - [qdma_io_cb](#), [42](#)
- [bypass_bar_regs](#)
 - [xlNX_pci_dev](#), [65](#)
- [c2h](#)
 - [mbox_msg_qctx](#), [16](#)
 - [qdma_queue_conf](#), [49](#)
- [c2h_buf_sz](#)
 - [global_csr_conf](#), [7](#)
- [c2h_buf_sz_idx](#)
 - [qdma_queue_conf](#), [50](#)
- [c2h_bufsz](#)
 - [qdma_queue_conf](#), [53](#)
- [c2h_cnt_th](#)
 - [global_csr_conf](#), [7](#)
- [c2h_descq](#)
 - [qdma_dev](#), [33](#)
- [c2h_qcnt](#)
 - [qdma_dev](#), [33](#)
- [c2h_qhndl](#)
 - [qdma_cdev](#), [22](#)
- [c2h_timer_cnt](#)
 - [global_csr_conf](#), [7](#)
- [c2h_use_fl](#)
 - [qdma_queue_conf](#), [50](#)
- [CMPT_DESC_SZ_16B](#)
 - [libqdma_export.h](#), [82](#)
- [CMPT_DESC_SZ_32B](#)
 - [libqdma_export.h](#), [82](#)
- [CMPT_DESC_SZ_64B](#)
 - [libqdma_export.h](#), [82](#)
- [CMPT_DESC_SZ_8B](#)
 - [libqdma_export.h](#), [82](#)
- [cdev](#)
 - [qdma_cdev](#), [22](#)
- [cdev.h](#), [67](#)
 - [QDMA_CDEV_CLASS_NAME](#), [68](#)
 - [QDMA_MINOR_MAX](#), [68](#)
 - [qdma_cdev_cleanup](#), [69](#)
 - [qdma_cdev_create](#), [68](#)
 - [qdma_cdev_destroy](#), [68](#)
 - [qdma_cdev_device_cleanup](#), [68](#)
 - [qdma_cdev_device_init](#), [69](#)
 - [qdma_cdev_init](#), [69](#)
- [cdev_cb](#)
 - [xlNX_pci_dev](#), [65](#)
- [cdev_lock](#)
 - [xlNX_pci_dev](#), [65](#)
- [cdev_major](#)
 - [qdma_cdev_cb](#), [24](#)
- [cdev_minor_cnt](#)
 - [qdma_cdev_cb](#), [24](#)
- [cdevno](#)
 - [qdma_cdev](#), [22](#)
- [cdh_flags](#)
 - [qdma_h2c_desc](#), [40](#)
- [cdh_max](#)
 - [qdma_queue_conf](#), [51](#)
- [cfg_state](#)
 - [qconf_entry](#), [18](#)
- [channel](#)
 - [qdma_descq](#), [28](#)
- [cidX](#)
 - [qdma_c2h_cmpt_cmpl_status](#), [20](#)
 - [qdma_desc_cmpl_status](#), [26](#)
 - [qdma_descq](#), [30](#)
 - [qdma_flq](#), [39](#)
 - [qdma_intr_ring](#), [41](#)
 - [qdma_sdesc_info](#), [56](#)
- [cidX_cmpt](#)
 - [qdma_descq](#), [31](#)
- [cidX_cmpt_pend](#)
 - [qdma_descq](#), [31](#)

- clear
 - mbox_msg_intr_ctxt, [15](#)
 - mbox_msg_qctxt, [16](#)
- cmpl_cnt_th_idx
 - qdma_queue_conf, [51](#)
- cmpl_desc_sz
 - qdma_queue_conf, [50](#)
- cmpl_en_intr
 - qdma_cmpl_ctrl, [25](#)
 - qdma_queue_conf, [51](#)
- cmpl_ovf_chk_dis
 - qdma_queue_conf, [51](#)
- cmpl_rng_sz_idx
 - qdma_queue_conf, [50](#)
- cmpl_stat_en
 - qdma_queue_conf, [50](#)
- cmpl_status_acc
 - global_csr_conf, [7](#)
- cmpl_status_acc_en
 - qdma_queue_conf, [49](#)
- cmpl_status_en
 - qdma_queue_conf, [49](#)
- cmpl_status_pend_chk
 - qdma_queue_conf, [49](#)
- cmpl_timer_idx
 - qdma_queue_conf, [51](#)
- cmpl_trig_mode
 - qdma_queue_conf, [51](#)
- cmpl_udd_en
 - qdma_queue_conf, [50](#)
- cmplthp
 - qdma_descq, [29](#)
- cmplthp_list
 - qdma_descq, [29](#)
- cmpt
 - hw_descq_context, [8](#)
- cmpt_desc_sz_t
 - libqdma_export.h, [82](#)
- cmpt_entry_len
 - qdma_descq, [31](#)
- cnt_th_idx
 - qdma_cmpl_ctrl, [24](#)
- coal_color
 - qdma_intr_ring, [41](#)
- color
 - intr_coal_conf, [9](#)
 - qdma_descq, [28](#)
- color_isr_status
 - qdma_c2h_cmpt_cmpl_status, [20](#)
- conf
 - qdma_descq, [28](#)
 - xlrx_dma_dev, [61](#)
- context
 - mbox_msg_qctxt, [17](#)
- count
 - qdma_request, [54](#)
- cpu
 - qdma_kthread, [44](#)
- cpu_assigned
 - qdma_descq, [28](#)
- cr
 - hw_descq_context, [8](#)
- credit
 - qdma_descq, [30](#)
- csr
 - mbox_msg, [12](#)
- csr_type
 - qdma_device.h, [113](#)
- cur_cfg_state
 - qdma_dev_conf, [36](#)
- DESC_SZ_16B
 - libqdma_export.h, [82](#)
- DESC_SZ_32B
 - libqdma_export.h, [82](#)
- DESC_SZ_64B
 - libqdma_export.h, [82](#)
- DESC_SZ_8B
 - libqdma_export.h, [82](#)
- DEVICE_VERSION_INFO_STR_LENGTH
 - libqdma_export.h, [80](#)
- DIRECT_INTR_MODE
 - libqdma_export.h, [81](#)
- desc
 - qdma_descq, [30](#)
 - qdma_flq, [38](#)
- desc_bus
 - qdma_descq, [30](#)
- desc_bypass
 - qdma_queue_conf, [50](#)
- desc_cmpl_status
 - qdma_descq, [30](#)
- desc_cmpt
 - qdma_descq, [32](#)
- desc_cmpt_bus
 - qdma_descq, [32](#)
- desc_cmpt_cmpl_status
 - qdma_descq, [32](#)
- desc_cmpt_cur
 - qdma_descq, [32](#)
- desc_nr
 - qdma_sgt_req_cb, [57](#)
- desc_rng_sz_idx
 - qdma_queue_conf, [49](#)
- desc_sz_t
 - libqdma_export.h, [82](#)
- descriptor, [5](#)
- descq_csr_info, [5](#)
- descq_flq_alloc_resource
 - qdma_st_c2h.h, [144](#)
- descq_flq_free_resource
 - qdma_descq.h, [112](#)
 - qdma_st_c2h.h, [144](#)
- descq_process_completion_st_c2h
 - qdma_st_c2h.h, [145](#)
- descq_st_c2h_read
 - qdma_st_c2h.h, [145](#)

- dev_hndl
 - xlnx_pci_dev, [64](#)
- dev_intr_info_list
 - xlnx_dma_dev, [62](#)
- dev_priv
 - xlnx_dma_dev, [62](#)
- dir_init
 - qdma_cdev, [22](#)
- dma_addr
 - qdma_sw_sg, [59](#)
- dma_mapped
 - qdma_request, [54](#)
- done
 - qdma_sgt_req_cb, [58](#)
- drv_mode_name, [5](#)
- dst
 - mbox_msg_hdr, [14](#)
- dst_addr
 - qdma_c2h_desc, [21](#)
 - qdma_mm_desc, [47](#)
- dvec_start_idx
 - xlnx_dma_dev, [62](#)
- en_stat_desc
 - qdma_cmpl_ctrl, [25](#)
- eop
 - qdma_sdesc_info, [56](#)
- ep_addr
 - qdma_request, [54](#)
- err
 - qdma_descq, [28](#)
- error
 - qdma_intr_ring, [41](#)
- F_MBOX_FN_CMD_RCV
 - qdma_mbox.h, [125](#)
- F_MBOX_FN_CMD_SND
 - qdma_mbox.h, [125](#)
- F_MBOX_FN_STATUS_ACK
 - qdma_mbox.h, [124](#)
- F_MBOX_FN_STATUS_IN_MSG
 - qdma_mbox.h, [124](#)
- F_MBOX_FN_STATUS_OUT_MSG
 - qdma_mbox.h, [124](#)
- F_MBOX_ISR_EN
 - qdma_mbox.h, [127](#)
- f_intr_handler
 - xdev.h, [152](#)
- fbits
 - qdma_sdesc_info, [55](#)
- fdone
 - qdma_kthread, [45](#)
- fetch_credit
 - qdma_queue_conf, [50](#)
- filler
 - mbox_msg_intr_ctxt, [15](#)
 - qdma_dev, [33](#)
 - qdma_sdesc_info, [56](#)
- finit
 - qdma_kthread, [44](#)
- fl_pg_order
 - qdma_descq, [30](#)
- flag
 - qdma_kthread, [44](#)
- flag_len
 - qdma_mm_desc, [47](#)
- flags
 - qdma_h2c_desc, [40](#)
- flq
 - qdma_descq, [31](#)
- flr_prsnt
 - xlnx_dma_dev, [61](#)
- fmap
 - hw_descq_context, [8](#)
 - mbox_msg, [12](#)
- fp_close_extra
 - qdma_cdev, [22](#)
- fp_descq_c2h_packet
 - qdma_queue_conf, [52](#)
- fp_descq_isr_top
 - qdma_queue_conf, [52](#)
- fp_done
 - qdma_request, [54](#)
- fp_ioctl_extra
 - qdma_cdev, [23](#)
- fp_open_extra
 - qdma_cdev, [22](#)
- fp_q_isr_top_dev
 - qdma_dev_conf, [37](#)
- fp_rw
 - qdma_cdev, [23](#)
- fp_user_isr_handler
 - qdma_dev_conf, [37](#)
- fpending
 - qdma_kthread, [44](#)
- fproc
 - qdma_kthread, [45](#)
- free
 - free_entry, [6](#)
- free_entry, [6](#)
 - free, [6](#)
 - index, [6](#)
 - list_head, [6](#)
 - next_qbase, [6](#)
- func_id
 - qconf_entry, [18](#)
 - xlnx_dma_dev, [62](#)
- G_MBOX_FN_STATUS_SRC
 - qdma_mbox.h, [125](#)
- get_intr_ring_index
 - qdma_intr.h, [121](#)
- global_csr_conf, [6](#)
 - c2h_buf_sz, [7](#)
 - c2h_cnt_th, [7](#)
 - c2h_timer_cnt, [7](#)
 - cmpl_status_acc, [7](#)
 - ring_sz, [7](#)

- h2c_descq
 - qdma_dev, [33](#)
- h2c_eot
 - qdma_request, [54](#)
- h2c_qcnt
 - qdma_dev, [33](#)
- h2c_qhndl
 - qdma_cdev, [22](#)
- hdr
 - mbox_msg, [11](#)
 - mbox_msg_csr, [13](#)
 - mbox_msg_fmap, [13](#)
 - mbox_msg_intr_ctxt, [15](#)
 - mbox_msg_qctxt, [16](#)
- hw
 - hw_descq_context, [8](#)
- hw_descq_context, [7](#)
 - cmpt, [8](#)
 - cr, [8](#)
 - fmap, [8](#)
 - hw, [8](#)
 - prefetch, [8](#)
 - sw, [8](#)
- hw_prg_lock
 - xlnx_dma_dev, [61](#)
- hw_rx_lock
 - qdma_mbox, [46](#)
- hw_tx_lock
 - qdma_mbox, [45](#)
- INDIRECT_INTR_MODE
 - libqdma_export.h, [81](#)
- INTR_RING_SZ_12KB
 - libqdma_export.h, [81](#)
- INTR_RING_SZ_16KB
 - libqdma_export.h, [81](#)
- INTR_RING_SZ_20KB
 - libqdma_export.h, [81](#)
- INTR_RING_SZ_24KB
 - libqdma_export.h, [81](#)
- INTR_RING_SZ_28KB
 - libqdma_export.h, [81](#)
- INTR_RING_SZ_32KB
 - libqdma_export.h, [81](#)
- INTR_RING_SZ_4KB
 - libqdma_export.h, [81](#)
- INTR_RING_SZ_8KB
 - libqdma_export.h, [81](#)
- INTR_TYPE_DATA
 - xdev.h, [152](#)
- INTR_TYPE_ERROR
 - xdev.h, [152](#)
- INTR_TYPE_MAX
 - xdev.h, [152](#)
- INTR_TYPE_USER
 - xdev.h, [152](#)
- id
 - qdma_kthread, [44](#)
- idx
 - qconf_entry, [17](#)
 - qdma_dev_conf, [36](#)
 - xlnx_pci_dev, [65](#)
- idx_bufsz
 - qdma_csr_info, [25](#)
- incr_cmpl_desc_cnt
 - qdma_descq.h, [111](#)
 - qdma_st_c2h.h, [144](#)
- index
 - free_entry, [6](#)
- init_qrange
 - qdma_dev, [33](#)
- intr_cidx_update
 - qdma_descq.h, [110](#)
- intr_coal_conf, [8](#)
 - color, [9](#)
 - intr_ring_base, [9](#)
 - intr_rng_num_entries, [9](#)
 - vec_id, [9](#)
- intr_coal_list
 - xlnx_dma_dev, [63](#)
- intr_context_setup
 - qdma_intr.h, [118](#)
- intr_ctxt
 - mbox_msg, [12](#)
- intr_en
 - mbox_msg_qctxt, [16](#)
- intr_handler
 - intr_vec_map_type, [10](#)
- intr_id
 - mbox_msg_qctxt, [16](#)
 - qdma_descq, [29](#)
- intr_info_t, [9](#)
 - intr_list, [10](#)
 - intr_list_cnt, [10](#)
 - msix_name, [10](#)
- intr_legacy_clear
 - qdma_intr.h, [119](#)
- intr_legacy_init
 - qdma_intr.h, [119](#)
- intr_legacy_setup
 - qdma_intr.h, [119](#)
- intr_list
 - intr_info_t, [10](#)
 - qdma_descq, [29](#)
- intr_list_cnt
 - intr_info_t, [10](#)
- intr_ring_base
 - intr_coal_conf, [9](#)
- intr_ring_setup
 - qdma_intr.h, [119](#)
- intr_ring_size_sel
 - libqdma_export.h, [81](#)
- intr_ring_teardown
 - qdma_intr.h, [118](#)
- intr_rng_num_entries
 - intr_coal_conf, [9](#)
- intr_rngsz

- qdma_dev_conf, 35
- intr_satus
 - qdma_intr_ring, 41
- intr_setup
 - qdma_intr.h, 118
- intr_teardown
 - qdma_intr.h, 118
- intr_type
 - intr_vec_map_type, 10
 - qdma_intr_ring, 41
- intr_type_list
 - xdev.h, 152
- intr_vec_index
 - intr_vec_map_type, 10
- intr_vec_map_type, 10
 - intr_handler, 10
 - intr_type, 10
 - intr_vec_index, 10
- intr_work
 - qdma_intr.h, 120
- intr_work_cpu
 - qdma_descq, 28
- io_batch_cnt
 - qdma_descq, 30
- irq_en
 - qdma_queue_conf, 49
- isr_top_q_en
 - qdma_dev_conf, 35
- left
 - qdma_sgt_req_cb, 57
- legacy_intr_q_list
 - qdma_descq, 29
- len
 - qdma_h2c_desc, 40
 - qdma_io_cb, 42
 - qdma_sw_sg, 59
- libqdma_config.h, 69
 - MAX_DMA_DEV, 71
 - MAX_QS_PER_PF, 71
 - PCI_SHIFT_BUS, 71
 - PCI_SHIFT_DEV, 72
 - QDMA_CONFIG_BAR, 70
 - QDMA_DEV_MSIX_VEC_MAX, 72
 - QDMA_INTR_COAL_RING_SIZE, 72
 - QDMA_NUM_DATA_VEC_FOR_INTR_CXT, 72
 - QDMA_PF_MAX, 71
 - QDMA_Q_PER_PF_MAX, 71
 - QDMA_Q_PER_VF_MAX, 71
 - QDMA_VF_MAX, 71
 - qdma_get_buf_sz, 74
 - qdma_get_cmpl_status_acc, 74
 - qdma_get_cnt_thresh, 76
 - qdma_get_glbl_rng_sz, 75
 - qdma_get_intr_rngsz, 73
 - qdma_get_qmax, 73
 - qdma_get_timer_cnt, 76
 - qdma_set_buf_sz, 74
 - qdma_set_cmpl_status_acc, 73
 - qdma_set_cnt_thresh, 76
 - qdma_set_glbl_rng_sz, 75
 - qdma_set_intr_rngsz, 73
 - qdma_set_qmax, 72
 - qdma_set_timer_cnt, 75
 - SHIFT_DEC_PCI_BUS, 72
 - SHIFT_DEC_PCI_DEV, 72
 - STM_BAR, 70
 - TOTAL_PF_QS, 71
 - TOTAL_QDMA_QS, 71
 - TOTAL_VF_QS, 71
- libqdma_exit
 - libqdma_export.h, 83
- libqdma_export.h, 77
 - AUTO_MODE, 81
 - CMPT_DESC_SZ_16B, 82
 - CMPT_DESC_SZ_32B, 82
 - CMPT_DESC_SZ_64B, 82
 - CMPT_DESC_SZ_8B, 82
 - cmpt_desc_sz_t, 82
 - DESC_SZ_16B, 82
 - DESC_SZ_32B, 82
 - DESC_SZ_64B, 82
 - DESC_SZ_8B, 82
 - DEVICE_VERSION_INFO_STR_LENGTH, 80
 - DIRECT_INTR_MODE, 81
 - desc_sz_t, 82
 - INDIRECT_INTR_MODE, 81
 - INTR_RING_SZ_12KB, 81
 - INTR_RING_SZ_16KB, 81
 - INTR_RING_SZ_20KB, 81
 - INTR_RING_SZ_24KB, 81
 - INTR_RING_SZ_28KB, 81
 - INTR_RING_SZ_32KB, 81
 - INTR_RING_SZ_4KB, 81
 - INTR_RING_SZ_8KB, 81
 - intr_ring_size_sel, 81
 - libqdma_exit, 83
 - libqdma_init, 82
 - POLL_MODE, 81
 - QDMA_DEV_NAME_MAXLEN, 79
 - QDMA_ERR_DESCQ_FULL, 81
 - QDMA_ERR_DESCQ_IDX_ALREADY_ADDED, 81
 - QDMA_ERR_DESCQ_SETUP_FAILED, 81
 - QDMA_ERR_INTERFACE_NOT_ENABLED_IN↵_DEVICE, 81
 - QDMA_ERR_INVALID_DESCQ_STATE, 81
 - QDMA_ERR_INVALID_DIRECTION, 81
 - QDMA_ERR_INVALID_INPUT_PARAM, 81
 - QDMA_ERR_INVALID_PCI_DEV, 81
 - QDMA_ERR_INVALID_QDMA_DEVICE, 81
 - QDMA_ERR_INVALID_QIDX, 81
 - QDMA_ERR_OUT_OF_MEMORY, 81
 - QDMA_ERR_PCI_DEVICE_ALREADY_ATTAC↵HED, 81
 - QDMA_ERR_PCI_DEVICE_ENABLE_FAILED, 81
 - QDMA_ERR_PCI_DEVICE_INIT_FAILED, 81

- QDMA_ERR_PCI_DEVICE_NOT_FOUND, 81
- QDMA_ERR_QUEUE_ALREADY_CONFIGURED, 81
- QDMA_FUNC_ID_INVALID, 79
- QDMA_GLOBAL_CSR_ARRAY_SZ, 80
- QDMA_OPERATION_SUCCESSFUL, 81
- QDMA_QUEUE_IDX_INVALID, 80
- QDMA_QUEUE_NAME_MAXLEN, 80
- QDMA_QUEUE_VEC_INVALID, 80
- QDMA_REQ_OPAQUE_SIZE, 80
- QDMA_UDD_MAXLEN, 80
- QMAX_CFG_INITIAL, 82
- QMAX_CFG_UNCONFIGURED, 82
- QMAX_CFG_USER, 82
- qdma_batch_request_submit, 93
- qdma_descq_get_cmpt_udd, 96
- qdma_dev_qmax_state, 81
- qdma_device_clear_stats, 85
- qdma_device_close, 83
- qdma_device_flr_quirk_check, 84
- qdma_device_flr_quirk_set, 84
- qdma_device_get_config, 85
- qdma_device_get_mmc2h_pkts, 86
- qdma_device_get_mmh2c_pkts, 85
- qdma_device_get_stc2h_pkts, 86
- qdma_device_get_sth2c_pkts, 86
- qdma_device_offline, 83
- qdma_device_online, 84
- qdma_device_open, 83
- qdma_device_read_config_register, 87
- qdma_device_set_cfg_state, 87
- qdma_device_set_config, 86
- qdma_device_sriov_config, 87
- qdma_device_version_info, 88
- qdma_device_write_config_register, 88
- qdma_drv_mode, 81
- qdma_error_codes, 81
- qdma_global_csr_get, 88
- qdma_global_csr_set, 89
- qdma_intr_ring_dump, 96
- qdma_queue_add, 89
- qdma_queue_avail_desc, 94
- qdma_queue_c2h_peek, 94
- qdma_queue_cmpl_ctrl, 94
- qdma_queue_dump, 92
- qdma_queue_dump_cmpt, 93
- qdma_queue_dump_desc, 92
- qdma_queue_get_config, 91
- qdma_queue_list, 92
- qdma_queue_packet_read, 95
- qdma_queue_packet_write, 95
- qdma_queue_prog_stm, 91
- qdma_queue_reconfig, 89
- qdma_queue_remove, 91
- qdma_queue_service, 95
- qdma_queue_start, 90
- qdma_queue_stop, 90
- qdma_request_submit, 93
- qdma_software_version_info, 88
- TRIG_MODE_ANY, 82
- TRIG_MODE_COMBO, 82
- TRIG_MODE_COUNTER, 82
- TRIG_MODE_DISABLE, 82
- TRIG_MODE_TIMER, 82
- TRIG_MODE_USER, 82
- tigger_mode_t, 82
- libqdma_init
 - libqdma_export.h, 82
- list
 - mbox_msg, 11
 - qdma_sgt_req_cb, 57
- list_head
 - free_entry, 6
 - qconf_entry, 17
 - qdma_cdev, 22
 - xlrx_dma_dev, 61
 - xlrx_pci_dev, 64
- list_lock
 - qdma_mbox, 46
- lock
 - qdma_cdev_cb, 23
 - qdma_descq, 28
 - qdma_dev, 33
 - qdma_kthread, 43
 - qdma_mbox, 45
 - xlrx_dma_dev, 61
- lock_descq
 - qdma_descq.h, 105
- lock_thread
 - thread.h, 147
- M_MBOX_FN_CMD_RCV
 - qdma_mbox.h, 125
- M_MBOX_FN_CMD_SND
 - qdma_mbox.h, 125
- M_MBOX_FN_CMD_VF_RESET
 - qdma_mbox.h, 126
- M_MBOX_FN_STATUS_ACK
 - qdma_mbox.h, 124
- M_MBOX_FN_STATUS_IN_MSG
 - qdma_mbox.h, 124
- M_MBOX_FN_STATUS_OUT_MSG
 - qdma_mbox.h, 124
- M_MBOX_FN_STATUS_SRC
 - qdma_mbox.h, 124
- M_MBOX_FN_TARGET_ID
 - qdma_mbox.h, 126
- M_MBOX_ISR_EN
 - qdma_mbox.h, 127
- M_MBOX_ISR_VEC
 - qdma_mbox.h, 126
- MAX_DMA_DEV
 - libqdma_config.h, 71
- MAX_QFLAGS
 - qdma_nl.h, 140
- MAX_QS_PER_PF
 - libqdma_config.h, 71

MBOX_BASE
 qdma_mbox.h, [123](#)
 MBOX_FN_CMD
 qdma_mbox.h, [125](#)
 MBOX_FN_STATUS_MASK
 qdma_mbox.h, [125](#)
 MBOX_FN_STATUS
 qdma_mbox.h, [123](#)
 MBOX_FN_TARGET
 qdma_mbox.h, [126](#)
 MBOX_IN_MSG_BASE
 qdma_mbox.h, [127](#)
 MBOX_ISR_EN
 qdma_mbox.h, [127](#)
 MBOX_ISR_VEC
 qdma_mbox.h, [126](#)
 MBOX_MSG_OP_PF_MASK
 qdma_mbox.h, [128](#)
 MBOX_MSG_REG_MAX
 qdma_mbox.h, [128](#)
 MBOX_MSG_STEP
 qdma_mbox.h, [127](#)
 MBOX_OP_BYE
 qdma_mbox.h, [128](#)
 MBOX_OP_CSR_RESP
 qdma_mbox.h, [128](#)
 MBOX_OP_CSR
 qdma_mbox.h, [128](#)
 MBOX_OP_FMAP_RESP
 qdma_mbox.h, [128](#)
 MBOX_OP_FMAP
 qdma_mbox.h, [128](#)
 MBOX_OP_HELLO_RESP
 qdma_mbox.h, [128](#)
 MBOX_OP_HELLO
 qdma_mbox.h, [128](#)
 MBOX_OP_INTR_CTXT_RESP
 qdma_mbox.h, [128](#)
 MBOX_OP_INTR_CTXT
 qdma_mbox.h, [128](#)
 MBOX_OP_MAX
 qdma_mbox.h, [128](#)
 MBOX_OP_QCONF_RESP
 qdma_mbox.h, [128](#)
 MBOX_OP_QCONF
 qdma_mbox.h, [128](#)
 MBOX_OP_QCTXT_CLR_RESP
 qdma_mbox.h, [128](#)
 MBOX_OP_QCTXT_CLR
 qdma_mbox.h, [128](#)
 MBOX_OP_QCTXT_INV_RESP
 qdma_mbox.h, [128](#)
 MBOX_OP_QCTXT_INV
 qdma_mbox.h, [128](#)
 MBOX_OP_QCTXT_RD_RESP
 qdma_mbox.h, [128](#)
 MBOX_OP_QCTXT_RD
 qdma_mbox.h, [128](#)
 MBOX_OP_QCTXT_WRT_RESP
 qdma_mbox.h, [128](#)
 MBOX_OP_QCTXT_WRT
 qdma_mbox.h, [128](#)
 MBOX_OUT_MSG_BASE
 qdma_mbox.h, [127](#)
 MBOX_PF_ACK_BASE
 qdma_mbox.h, [127](#)
 MBOX_PF_ACK_COUNT
 qdma_mbox.h, [127](#)
 MBOX_PF_ACK_STEP
 qdma_mbox.h, [127](#)
 MODULE_DEVICE_TABLE
 pci_ids.h, [98](#)
 mapping_err
 qdma_flq, [38](#)
 master_pf
 qdma_dev_conf, [35](#)
 mbox
 xlnx_dma_dev, [63](#)
 mbox_msg, [11](#)
 csr, [12](#)
 fmap, [12](#)
 hdr, [11](#)
 intr_ctxt, [12](#)
 list, [11](#)
 qctxt, [12](#)
 raw, [12](#)
 waitq, [11](#)
 mbox_msg_csr, [12](#)
 hdr, [13](#)
 mbox_msg_fmap, [13](#)
 hdr, [13](#)
 qbase, [13](#)
 qmax, [13](#)
 mbox_msg_hdr, [14](#)
 dst, [14](#)
 src, [14](#)
 status, [14](#)
 mbox_msg_intr_ctxt, [14](#)
 clear, [15](#)
 filler, [15](#)
 hdr, [15](#)
 num_rings, [15](#)
 ring_index_list, [15](#)
 vec_base, [15](#)
 w, [15](#)
 mbox_msg_op
 qdma_mbox.h, [128](#)
 mbox_msg_qctxt, [15](#)
 c2h, [16](#)
 clear, [16](#)
 context, [17](#)
 hdr, [16](#)
 intr_en, [16](#)
 intr_id, [16](#)
 qid, [17](#)
 st, [16](#)

- verify, 16
- minor
 - qdma_cdev, 22
- mm_mode_en
 - xlnx_dma_dev, 61
- mod_name
 - xlnx_dma_dev, 61
- msix
 - xlnx_dma_dev, 62
- msix_name
 - intr_info_t, 10
- msix_qvec_max
 - qdma_dev_conf, 35
- name
 - qdma_cdev, 23
 - qdma_dev_conf, 36
 - qdma_kthread, 43
 - qdma_queue_conf, 52
- next
 - qdma_sdesc_info, 55
 - qdma_sw_sg, 58
- next_qbase
 - free_entry, 6
- nl.h, 97
 - xnl_respond_buffer, 97
- nl_task_wq
 - xlnx_pci_dev, 65
- num_rings
 - mbox_msg_intr_ctxt, 15
- num_vecs
 - xlnx_dma_dev, 62
- offset
 - qdma_sgt_req_cb, 57
 - qdma_sw_sg, 59
- opaque
 - qdma_request, 54
- PCI_DMA_H
 - xdev.h, 151
- PCI_DMA_L
 - xdev.h, 151
- PCI_SHIFT_BUS
 - libqdma_config.h, 71
- PCI_SHIFT_DEV
 - libqdma_config.h, 72
- POLL_MODE
 - libqdma_export.h, 81
- pages
 - qdma_io_cb, 42
- pages_nr
 - qdma_io_cb, 42
- pci_ids.h, 97
 - MODULE_DEVICE_TABLE, 98
- pdev
 - qdma_dev_conf, 34
 - xlnx_pci_dev, 64
- pend_list
 - qdma_descq, 29
- pend_list_empty
 - qdma_descq, 29
- pend_list_wq
 - qdma_descq, 29
- pend_req_desc
 - qdma_descq, 30
- pf_count
 - xlnx_dma_dev, 62
- pf_list
 - qconf_entry_head, 19
- pf_qmax
 - qconf_entry_head, 19
- pfetch_bypass
 - qdma_queue_conf, 51
- pfetch_en
 - qdma_queue_conf, 50
- pg
 - qdma_sw_sg, 58
- pg_order
 - qdma_flq, 38
- pg_shift
 - qdma_flq, 38
- pidx
 - qdma_c2h_cmpt_cmpl_status, 20
 - qdma_desc_cmpl_status, 26
 - qdma_descq, 30
 - qdma_flq, 39
 - qdma_intr_ring, 41
- pidx_cmpt
 - qdma_descq, 31
- pidx_pend
 - qdma_flq, 39
- pipe
 - qdma_queue_conf, 49
- pipe_flow_id
 - qdma_queue_conf, 52
- pipe_gl_max
 - qdma_queue_conf, 52
- pipe_slr_id
 - qdma_queue_conf, 52
- pipe_stm_max_pkt_size
 - xlnx_dma_dev, 63
- pipe_tdest
 - qdma_queue_conf, 52
- pkt_cnt
 - qdma_descq, 31
 - qdma_flq, 38
- pkt_dlen
 - qdma_descq, 31
 - qdma_flq, 38
- pld_len
 - qdma_h2c_desc, 40
- port_id
 - qdma_queue_conf, 51
- pr_debug_thread
 - thread.h, 148
- prefetch

- hw_descq_context, [8](#)
- proc_req_running
 - qdma_descq, [28](#)
- Q_STATE_ENABLED
 - qdma_descq.h, [105](#)
- Q_STATE_ONLINE
 - qdma_descq.h, [105](#)
- q_state
 - qdma_descq, [28](#)
- q_state_t
 - qdma_descq.h, [105](#)
- QDMA_BAR_NUM
 - xdev.h, [151](#)
- QDMA_CDEV_CLASS_NAME
 - cdev.h, [68](#)
- QDMA_CONFIG_BAR
 - libqdma_config.h, [70](#)
- QDMA_CSR_TYPE_BUFSZ
 - qdma_device.h, [113](#)
- QDMA_CSR_TYPE_CNT_TH
 - qdma_device.h, [113](#)
- QDMA_CSR_TYPE_MAX
 - qdma_device.h, [113](#)
- QDMA_CSR_TYPE_TIMER_CNT
 - qdma_device.h, [113](#)
- QDMA_DESC_BLEN_BITS
 - xdev.h, [151](#)
- QDMA_DESC_BLEN_MAX
 - xdev.h, [151](#)
- QDMA_DEV_MSIX_VEC_MAX
 - libqdma_config.h, [72](#)
- QDMA_DEV_NAME_MAXLEN
 - libqdma_export.h, [79](#)
- QDMA_ERR_DESCQ_FULL
 - libqdma_export.h, [81](#)
- QDMA_ERR_DESCQ_IDX_ALREADY_ADDED
 - libqdma_export.h, [81](#)
- QDMA_ERR_DESCQ_SETUP_FAILED
 - libqdma_export.h, [81](#)
- QDMA_ERR_INTERFACE_NOT_ENABLED_IN_DEVICE
 - libqdma_export.h, [81](#)
- QDMA_ERR_INVALID_DESCQ_STATE
 - libqdma_export.h, [81](#)
- QDMA_ERR_INVALID_DIRECTION
 - libqdma_export.h, [81](#)
- QDMA_ERR_INVALID_INPUT_PARAM
 - libqdma_export.h, [81](#)
- QDMA_ERR_INVALID_PCI_DEV
 - libqdma_export.h, [81](#)
- QDMA_ERR_INVALID_QDMA_DEVICE
 - libqdma_export.h, [81](#)
- QDMA_ERR_INVALID_QIDX
 - libqdma_export.h, [81](#)
- QDMA_ERR_OUT_OF_MEMORY
 - libqdma_export.h, [81](#)
- QDMA_ERR_PCI_DEVICE_ALREADY_ATTACHED
 - libqdma_export.h, [81](#)
- QDMA_ERR_PCI_DEVICE_ENABLE_FAILED
 - libqdma_export.h, [81](#)
- QDMA_ERR_PCI_DEVICE_INIT_FAILED
 - libqdma_export.h, [81](#)
- QDMA_ERR_PCI_DEVICE_NOT_FOUND
 - libqdma_export.h, [81](#)
- QDMA_ERR_QUEUE_ALREADY_CONFIGURED
 - libqdma_export.h, [81](#)
- QDMA_FUNC_ID_INVALID
 - libqdma_export.h, [79](#)
- QDMA_GLOBAL_CSR_ARRAY_SZ
 - libqdma_export.h, [80](#)
- QDMA_INTR_COAL_RING_SIZE
 - libqdma_config.h, [72](#)
- QDMA_MAX_BAR_LEN_MAPPED
 - xdev.h, [151](#)
- QDMA_MAX_INT_RING_ENTRIES
 - qdma_nl.h, [140](#)
- QDMA_MINOR_MAX
 - cdev.h, [68](#)
- QDMA_NUM_DATA_VEC_FOR_INTR_CXT
 - libqdma_config.h, [72](#)
- QDMA_OPERATION_SUCCESSFUL
 - libqdma_export.h, [81](#)
- QDMA_PF_MAX
 - libqdma_config.h, [71](#)
- QDMA_Q_PER_PF_MAX
 - libqdma_config.h, [71](#)
- QDMA_Q_PER_VF_MAX
 - libqdma_config.h, [71](#)
- QDMA_QUEUE_IDX_INVALID
 - libqdma_export.h, [80](#)
- QDMA_QUEUE_NAME_MAXLEN
 - libqdma_export.h, [80](#)
- QDMA_QUEUE_VEC_INVALID
 - libqdma_export.h, [80](#)
- QDMA_REQ_OPAQUE_SIZE
 - libqdma_export.h, [80](#)
- QDMA_UDD_MAXLEN
 - libqdma_export.h, [80](#)
- QDMA_VF_MAX
 - libqdma_config.h, [71](#)
- QMAX_CFG_INITIAL
 - libqdma_export.h, [82](#)
- QMAX_CFG_UNCONFIGURED
 - libqdma_export.h, [82](#)
- QMAX_CFG_USER
 - libqdma_export.h, [82](#)
- qbase
 - mbox_msg_fmap, [13](#)
 - qconf_entry, [17](#)
 - qdma_dev, [33](#)
- qcnt_cfgd_free
 - qconf_entry_head, [19](#)
- qcnt_init_free
 - qconf_entry_head, [19](#)
- qcnt_init_used
 - qconf_entry_head, [19](#)

- qconf_entry, 17
 - cfg_state, 18
 - func_id, 18
 - idx, 17
 - list_head, 17
 - qbase, 17
 - qmax, 18
 - type, 18
- qconf_entry_head, 18
 - pf_list, 19
 - pf_qmax, 19
 - qcnt_cfgd_free, 19
 - qcnt_init_free, 19
 - qcnt_init_used, 19
 - vf_cnt, 19
 - vf_free_list, 19
 - vf_list, 19
 - vf_qbase, 19
 - vf_qmax, 19
- qctx
 - mbox_msg, 12
- qdata
 - xlnx_pci_dev, 65
- qdma_batch_request_submit
 - libqdma_export.h, 93
- qdma_c2h_cmpt_cmpl_status, 20
 - cidx, 20
 - color_isr_status, 20
 - pidx, 20
- qdma_c2h_desc, 20
 - dst_addr, 21
- qdma_cdev, 21
 - c2h_qhndl, 22
 - cdev, 22
 - cdevno, 22
 - dir_init, 22
 - fp_close_extra, 22
 - fp_ioctl_extra, 23
 - fp_open_extra, 22
 - fp_rw, 23
 - h2c_qhndl, 22
 - list_head, 22
 - minor, 22
 - name, 23
 - sys_device, 22
 - xcb, 22
- qdma_cdev_cb, 23
 - cdev_major, 24
 - cdev_minor_cnt, 24
 - lock, 23
 - xpdev, 23
- qdma_cdev_cleanup
 - cdev.h, 69
- qdma_cdev_create
 - cdev.h, 68
- qdma_cdev_destroy
 - cdev.h, 68
- qdma_cdev_device_cleanup
 - cdev.h, 68
- qdma_cdev_device_init
 - cdev.h, 69
- qdma_cdev_init
 - cdev.h, 69
- qdma_cmpl_ctrl, 24
 - cmpl_en_intr, 25
 - cnt_th_idx, 24
 - en_stat_desc, 25
 - timer_idx, 24
 - trigger_mode, 24
- qdma_compat.h, 98
 - qdma_wait_queue, 98
- qdma_context.h, 98
 - qdma_descq_context_clear, 100
 - qdma_descq_context_program, 102
 - qdma_descq_context_read, 102
 - qdma_descq_context_setup, 100
 - qdma_descq_stm_clear, 100
 - qdma_descq_stm_program, 103
 - qdma_descq_stm_read, 103
 - qdma_descq_stm_setup, 100
 - qdma_intr_context_read, 102
 - qdma_intr_context_setup, 99
 - qdma_prog_intr_context, 99
- qdma_csr_info, 25
 - array, 25
 - idx_bufsz, 25
- qdma_csr_read
 - qdma_device.h, 117
- qdma_csr_read_bufsz
 - qdma_device.h, 116
- qdma_csr_read_cmpl_status_acc
 - qdma_device.h, 115
- qdma_csr_read_cnt_thresh
 - qdma_device.h, 116
- qdma_csr_read_rngsz
 - qdma_device.h, 115
- qdma_csr_read_timer_cnt
 - qdma_device.h, 116
- qdma_desc_cmpl_status, 26
 - cidx, 26
 - pidx, 26
 - rsvd, 26
- qdma_descq, 26
 - avail, 30
 - channel, 28
 - cidx, 30
 - cidx_cmpt, 31
 - cidx_cmpt_pend, 31
 - cmplthp, 29
 - cmplthp_list, 29
 - cmpt_entry_len, 31
 - color, 28
 - conf, 28
 - cpu_assigned, 28
 - credit, 30
 - desc, 30

- desc_bus, 30
- desc_cmpl_status, 30
- desc_cmpt, 32
- desc_cmpt_bus, 32
- desc_cmpt_cmpl_status, 32
- desc_cmpt_cur, 32
- err, 28
- fl_pg_order, 30
- flq, 31
- intr_id, 29
- intr_list, 29
- intr_work_cpu, 28
- io_batch_cnt, 30
- legacy_intr_q_list, 29
- lock, 28
- pend_list, 29
- pend_list_empty, 29
- pend_list_wq, 29
- pend_req_desc, 30
- pidx, 30
- pidx_cmpt, 31
- pkt_cnt, 31
- pkt_dlen, 31
- proc_req_running, 28
- q_state, 28
- qidx_hw, 28
- rsvd, 31
- total_cmpl_descs, 31
- udd_cnt, 31
- work, 29
- work_list, 29
- xdev, 28
- qdma_descq.h, 104
 - descq_flq_free_resource, 112
 - incr_cmpl_desc_cnt, 111
 - intr_cidx_update, 110
 - lock_descq, 105
 - Q_STATE_ENABLED, 105
 - Q_STATE_ONLINE, 105
 - q_state_t, 105
 - qdma_descq_alloc_resource, 108
 - qdma_descq_cleanup, 107
 - qdma_descq_config, 107
 - qdma_descq_config_complete, 107
 - qdma_descq_context_cleanup, 109
 - qdma_descq_dump, 109
 - qdma_descq_dump_desc, 110
 - qdma_descq_dump_state, 110
 - qdma_descq_free_resource, 108
 - qdma_descq_init, 106
 - qdma_descq_proc_sgt_request, 111
 - qdma_descq_prog_hw, 108
 - qdma_descq_prog_stm, 108
 - qdma_descq_service_cmpl_update, 109
 - qdma_req_cb_get, 105
 - qdma_sgt_req_done, 111
 - sgl_map, 111
 - sgl_unmap, 112
 - unlock_descq, 105
- qdma_descq_alloc_resource
 - qdma_descq.h, 108
- qdma_descq_cleanup
 - qdma_descq.h, 107
- qdma_descq_config
 - qdma_descq.h, 107
- qdma_descq_config_complete
 - qdma_descq.h, 107
- qdma_descq_context_cleanup
 - qdma_descq.h, 109
- qdma_descq_context_clear
 - qdma_context.h, 100
- qdma_descq_context_program
 - qdma_context.h, 102
- qdma_descq_context_read
 - qdma_context.h, 102
- qdma_descq_context_setup
 - qdma_context.h, 100
- qdma_descq_dump
 - qdma_descq.h, 109
- qdma_descq_dump_cmpt
 - qdma_st_c2h.h, 143
- qdma_descq_dump_desc
 - qdma_descq.h, 110
- qdma_descq_dump_state
 - qdma_descq.h, 110
- qdma_descq_free_resource
 - qdma_descq.h, 108
- qdma_descq_get_cmpt_udd
 - libqdma_export.h, 96
- qdma_descq_init
 - qdma_descq.h, 106
- qdma_descq_proc_sgt_request
 - qdma_descq.h, 111
- qdma_descq_prog_hw
 - qdma_descq.h, 108
- qdma_descq_prog_stm
 - qdma_descq.h, 108
- qdma_descq_rxq_read
 - qdma_st_c2h.h, 143
- qdma_descq_service_cmpl_update
 - qdma_descq.h, 109
- qdma_descq_stm_clear
 - qdma_context.h, 100
- qdma_descq_stm_program
 - qdma_context.h, 103
- qdma_descq_stm_read
 - qdma_context.h, 103
- qdma_descq_stm_setup
 - qdma_context.h, 100
- qdma_dev, 32
 - c2h_descq, 33
 - c2h_qcnt, 33
 - filler, 33
 - h2c_descq, 33
 - h2c_qcnt, 33
 - init_qrange, 33

- lock, 33
- qbase, 33
- qmax, 33
- qdma_dev_conf, 34
 - bar_num_bypass, 36
 - bar_num_config, 36
 - bar_num_stm, 36
 - bar_num_user, 36
 - bdf, 36
 - cur_cfg_state, 36
 - fp_q_isr_top_dev, 37
 - fp_user_isr_handler, 37
 - idx, 36
 - intr_rngsz, 35
 - isr_top_q_en, 35
 - master_pf, 35
 - msix_qvec_max, 35
 - name, 36
 - pdev, 34
 - qdma_drv_mode, 36
 - qsets_base, 36
 - qsets_max, 34
 - rsvd1, 35
 - rsvd2, 34
 - tm_mode_en, 37
 - tm_one_cdh_en, 37
 - uld, 35
 - vf_max, 35
 - zerolen_dma, 35
- qdma_dev_qmax_state
 - libqdma_export.h, 81
- qdma_device.h, 112
 - csr_type, 113
 - QDMA_CSR_TYPE_BUFSZ, 113
 - QDMA_CSR_TYPE_CNT_TH, 113
 - QDMA_CSR_TYPE_MAX, 113
 - QDMA_CSR_TYPE_TIMER_CNT, 113
 - qdma_csr_read, 117
 - qdma_csr_read_bufsz, 116
 - qdma_csr_read_cmpl_status_acc, 115
 - qdma_csr_read_cnt_thresh, 116
 - qdma_csr_read_rngsz, 115
 - qdma_csr_read_timer_cnt, 116
 - qdma_device_cleanup, 114
 - qdma_device_get_descq_by_hw_qid, 115
 - qdma_device_get_descq_by_id, 114
 - qdma_device_get_id_from_descq, 114
 - qdma_device_init, 113
 - qdma_device_prep_q_resource, 115
 - xdev_2_qdev, 113
- qdma_device_cleanup
 - qdma_device.h, 114
- qdma_device_clear_stats
 - libqdma_export.h, 85
- qdma_device_close
 - libqdma_export.h, 83
- qdma_device_flr_quirk_check
 - libqdma_export.h, 84
- qdma_device_flr_quirk_set
 - libqdma_export.h, 84
- qdma_device_get_config
 - libqdma_export.h, 85
- qdma_device_get_descq_by_hw_qid
 - qdma_device.h, 115
- qdma_device_get_descq_by_id
 - qdma_device.h, 114
- qdma_device_get_id_from_descq
 - qdma_device.h, 114
- qdma_device_get_mmc2h_pkts
 - libqdma_export.h, 86
- qdma_device_get_mmh2c_pkts
 - libqdma_export.h, 85
- qdma_device_get_stc2h_pkts
 - libqdma_export.h, 86
- qdma_device_get_sth2c_pkts
 - libqdma_export.h, 86
- qdma_device_init
 - qdma_device.h, 113
- qdma_device_offline
 - libqdma_export.h, 83
- qdma_device_online
 - libqdma_export.h, 84
- qdma_device_open
 - libqdma_export.h, 83
- qdma_device_prep_q_resource
 - qdma_device.h, 115
- qdma_device_read_bypass_register
 - qdma_mod.h, 133
- qdma_device_read_config_register
 - libqdma_export.h, 87
- qdma_device_read_user_register
 - qdma_mod.h, 132
- qdma_device_set_cfg_state
 - libqdma_export.h, 87
- qdma_device_set_config
 - libqdma_export.h, 86
- qdma_device_sriov_config
 - libqdma_export.h, 87
- qdma_device_version_info
 - libqdma_export.h, 88
- qdma_device_write_bypass_register
 - qdma_mod.h, 133
- qdma_device_write_config_register
 - libqdma_export.h, 88
- qdma_device_write_user_register
 - qdma_mod.h, 133
- qdma_drv_mode
 - libqdma_export.h, 81
 - qdma_dev_conf, 36
- qdma_enable_hw_err
 - qdma_intr.h, 120
- qdma_err_idx
 - qdma_nl.h, 142
- qdma_err_intr_setup
 - qdma_intr.h, 120
- qdma_error_codes

- libqdma_export.h, 81
- qdma_flq, 37
 - alloc_fail, 38
 - avail, 38
 - cidx, 39
 - desc, 38
 - mapping_err, 38
 - pg_order, 38
 - pg_shift, 38
 - pidx, 39
 - pidx_pend, 39
 - pkt_cnt, 38
 - pkt_dlen, 38
 - sdesc, 39
 - sdesc_info, 39
 - size, 38
 - udd_cnt, 38
- qdma_get_buf_sz
 - libqdma_config.h, 74
- qdma_get_cmpl_status_acc
 - libqdma_config.h, 74
- qdma_get_cnt_thresh
 - libqdma_config.h, 76
- qdma_get_glbl_rng_sz
 - libqdma_config.h, 75
- qdma_get_intr_rngsz
 - libqdma_config.h, 73
- qdma_get_qmax
 - libqdma_config.h, 73
- qdma_get_timer_cnt
 - libqdma_config.h, 76
- qdma_global_csr_get
 - libqdma_export.h, 88
- qdma_global_csr_set
 - libqdma_export.h, 89
- qdma_h2c_desc, 39
 - cdh_flags, 40
 - flags, 40
 - len, 40
 - pld_len, 40
 - src_addr, 40
- qdma_intr.h, 117
 - get_intr_ring_index, 121
 - intr_context_setup, 118
 - intr_legacy_clear, 119
 - intr_legacy_init, 119
 - intr_legacy_setup, 119
 - intr_ring_setup, 119
 - intr_ring_teardown, 118
 - intr_setup, 118
 - intr_teardown, 118
 - intr_work, 120
 - qdma_enable_hw_err, 120
 - qdma_err_intr_setup, 120
- qdma_intr_context_read
 - qdma_context.h, 102
- qdma_intr_context_setup
 - qdma_context.h, 99
- qdma_intr_ring, 40
 - cidx, 41
 - coal_color, 41
 - error, 41
 - intr_satus, 41
 - intr_type, 41
 - pidx, 41
 - qid, 41
 - rsvd, 41
 - s_color, 41
- qdma_intr_ring_dump
 - libqdma_export.h, 96
- qdma_io_cb, 42
 - buf, 42
 - len, 42
 - pages, 42
 - pages_nr, 42
 - req, 43
 - sgl, 42
- qdma_kthread, 43
 - cpu, 44
 - fdone, 45
 - finit, 44
 - flag, 44
 - fpending, 44
 - fproc, 45
 - id, 44
 - lock, 43
 - name, 43
 - task, 44
 - timeout, 44
 - waitq, 44
 - work_cnt, 44
 - work_list, 44
- qdma_kthread_dump
 - thread.h, 148
- qdma_kthread_start
 - thread.h, 148
- qdma_kthread_stop
 - thread.h, 149
- qdma_kthread_wakeup
 - thread.h, 148
- qdma_mbox, 45
 - hw_rx_lock, 46
 - hw_tx_lock, 45
 - list_lock, 46
 - lock, 45
 - rx, 46
 - rx_pend_list, 46
 - rx_work, 46
 - timer, 46
 - tx_todo_list, 46
 - tx_work, 46
 - workq, 46
 - xdev, 46
- qdma_mbox.h, 121
 - __qdma_mbox_msg_free, 130
 - F_MBOX_FN_CMD_RCV, 125

- F_MBOX_FN_CMD_SND, 125
- F_MBOX_FN_STATUS_ACK, 124
- F_MBOX_FN_STATUS_IN_MSG, 124
- F_MBOX_FN_STATUS_OUT_MSG, 124
- F_MBOX_ISR_EN, 127
- G_MBOX_FN_STATUS_SRC, 125
- M_MBOX_FN_CMD_RCV, 125
- M_MBOX_FN_CMD_SND, 125
- M_MBOX_FN_CMD_VF_RESET, 126
- M_MBOX_FN_STATUS_ACK, 124
- M_MBOX_FN_STATUS_IN_MSG, 124
- M_MBOX_FN_STATUS_OUT_MSG, 124
- M_MBOX_FN_STATUS_SRC, 124
- M_MBOX_FN_TARGET_ID, 126
- M_MBOX_ISR_EN, 127
- M_MBOX_ISR_VEC, 126
- MBOX_BASE, 123
- MBOX_FN_CMD, 125
- MBOX_FN_STATUS_MASK, 125
- MBOX_FN_STATUS, 123
- MBOX_FN_TARGET, 126
- MBOX_IN_MSG_BASE, 127
- MBOX_ISR_EN, 127
- MBOX_ISR_VEC, 126
- MBOX_MSG_OP_PF_MASK, 128
- MBOX_MSG_REG_MAX, 128
- MBOX_MSG_STEP, 127
- MBOX_OP_BYE, 128
- MBOX_OP_CSR_RESP, 128
- MBOX_OP_CSR, 128
- MBOX_OP_FMAP_RESP, 128
- MBOX_OP_FMAP, 128
- MBOX_OP_HELLO_RESP, 128
- MBOX_OP_HELLO, 128
- MBOX_OP_INTR_CTXT_RESP, 128
- MBOX_OP_INTR_CTXT, 128
- MBOX_OP_MAX, 128
- MBOX_OP_QCONF_RESP, 128
- MBOX_OP_QCONF, 128
- MBOX_OP_QCTXT_CLR_RESP, 128
- MBOX_OP_QCTXT_CLR, 128
- MBOX_OP_QCTXT_INV_RESP, 128
- MBOX_OP_QCTXT_INV, 128
- MBOX_OP_QCTXT_RD_RESP, 128
- MBOX_OP_QCTXT_RD, 128
- MBOX_OP_QCTXT_WRT_RESP, 128
- MBOX_OP_QCTXT_WRT, 128
- MBOX_OUT_MSG_BASE, 127
- MBOX_PF_ACK_BASE, 127
- MBOX_PF_ACK_COUNT, 127
- MBOX_PF_ACK_STEP, 127
- mbox_msg_op, 128
- qdma_mbox_cleanup, 129
- qdma_mbox_init, 128
- qdma_mbox_msg_alloc, 129
- qdma_mbox_msg_send, 129
- S_MBOX_FN_CMD_RCV, 125
- S_MBOX_FN_CMD_SND, 125
- S_MBOX_FN_CMD_VF_RESET, 126
- S_MBOX_FN_STATUS_ACK, 124
- S_MBOX_FN_STATUS_IN_MSG, 123
- S_MBOX_FN_STATUS_OUT_MSG, 124
- S_MBOX_FN_STATUS_SRC, 124
- S_MBOX_FN_TARGET_ID, 126
- S_MBOX_ISR_EN, 127
- S_MBOX_ISR_VEC, 126
- V_MBOX_FN_TARGET_ID, 126
- V_MBOX_ISR_VEC, 126
- qdma_mbox_cleanup
 - qdma_mbox.h, 129
- qdma_mbox_init
 - qdma_mbox.h, 128
- qdma_mbox_msg_alloc
 - qdma_mbox.h, 129
- qdma_mbox_msg_send
 - qdma_mbox.h, 129
- qdma_mm_desc, 47
 - dst_addr, 47
 - flag_len, 47
 - rsvd0, 47
 - rsvd1, 47
 - src_addr, 47
- qdma_mod.h, 130
 - qdma_device_read_bypass_register, 133
 - qdma_device_read_user_register, 132
 - qdma_device_write_bypass_register, 133
 - qdma_device_write_user_register, 133
 - xpdev_find_by_idx, 131
 - xpdev_list_dump, 131
 - xpdev_queue_add, 132
 - xpdev_queue_delete, 132
 - xpdev_queue_get, 131
- qdma_nl.h, 133
 - MAX_QFLAGS, 140
 - QDMA_MAX_INT_RING_ENTRIES, 140
 - qdma_err_idx, 142
 - XNL_ATTR_C2H_BUFSZ_IDX, 141
 - XNL_ATTR_CMPT_CNTR_IDX, 141
 - XNL_ATTR_CMPT_DESC_SIZE, 141
 - XNL_ATTR_CMPT_TIMER_IDX, 141
 - XNL_ATTR_CMPT_TRIG_MODE, 141
 - XNL_ATTR_DEV_CFG_BAR, 141
 - XNL_ATTR_DEV_IDX, 141
 - XNL_ATTR_DEV_QSET_MAX, 141
 - XNL_ATTR_DEV_QSET_QBASE, 141
 - XNL_ATTR_DEV_STAT_MMC2H_PKTS1, 141
 - XNL_ATTR_DEV_STAT_MMC2H_PKTS2, 141
 - XNL_ATTR_DEV_STAT_MMH2C_PKTS1, 141
 - XNL_ATTR_DEV_STAT_MMH2C_PKTS2, 141
 - XNL_ATTR_DEV_STAT_STC2H_PKTS1, 141
 - XNL_ATTR_DEV_STAT_STC2H_PKTS2, 141
 - XNL_ATTR_DEV_STAT_STH2C_PKTS1, 141
 - XNL_ATTR_DEV_STAT_STH2C_PKTS2, 141
 - XNL_ATTR_DEV_STM_BAR, 141
 - XNL_ATTR_DEV_USR_BAR, 141
 - XNL_ATTR_DRV_INFO, 141

XNL_ATTR_GENMSG, 141
 XNL_ATTR_GLOBAL_CSR, 141
 XNL_ATTR_INTR_VECTOR_END_IDX, 141
 XNL_ATTR_INTR_VECTOR_IDX, 141
 XNL_ATTR_INTR_VECTOR_START_IDX, 141
 XNL_ATTR_NUM_Q, 141
 XNL_ATTR_PCI_BUS, 141
 XNL_ATTR_PCI_DEV, 141
 XNL_ATTR_PCI_FUNC, 141
 XNL_ATTR_PIPE_FLOW_ID, 141
 XNL_ATTR_PIPE_GL_MAX, 141
 XNL_ATTR_PIPE_SLR_ID, 141
 XNL_ATTR_PIPE_TDEST, 141
 XNL_ATTR_QFLAG, 141
 XNL_ATTR_QIDX, 141
 XNL_ATTR_QRNGSZ_IDX, 141
 XNL_ATTR_RANGE_END, 141
 XNL_ATTR_RANGE_START, 141
 XNL_ATTR_REG_ADDR, 141
 XNL_ATTR_REG_BAR_NUM, 141
 XNL_ATTR_REG_VAL, 141
 XNL_ATTR_RSP_BUF_LEN, 141
 XNL_ATTR_SW_DESC_SIZE, 141
 XNL_CMD_DEV_INFO, 142
 XNL_CMD_DEV_LIST, 142
 XNL_CMD_DEV_STAT_CLEAR, 142
 XNL_CMD_DEV_STAT, 142
 XNL_CMD_GLOBAL_CSR, 142
 XNL_CMD_INTR_RING_DUMP, 142
 XNL_CMD_MAX, 142
 XNL_CMD_Q_ADD, 142
 XNL_CMD_Q_CMPT, 142
 XNL_CMD_Q_DESC, 142
 XNL_CMD_Q_DEL, 142
 XNL_CMD_Q_DUMP, 142
 XNL_CMD_Q_LIST, 142
 XNL_CMD_Q_RX_PKT, 142
 XNL_CMD_Q_START, 142
 XNL_CMD_Q_STOP, 142
 XNL_CMD_Q_UDD, 142
 XNL_CMD_REG_DUMP, 142
 XNL_CMD_REG_RD, 142
 XNL_CMD_REG_WRT, 142
 XNL_CMD_VERSION, 142
 XNL_ERR_BUFLLEN, 138
 XNL_F_C2H_CMPL_INTR_EN, 140
 XNL_F_CMPL_STATUS_ACC_EN, 139
 XNL_F_CMPL_STATUS_DESC_EN, 140
 XNL_F_CMPL_STATUS_EN, 139
 XNL_F_CMPL_STATUS_PEND_CHK, 139
 XNL_F_CMPL_UDD_EN, 140
 XNL_F_CMPT_OVF_CHK_DIS, 140
 XNL_F_DESC_BYPASS_EN, 139
 XNL_F_FETCH_CREDIT, 139
 XNL_F_PFETCH_BYPASS_EN, 140
 XNL_F_PFETCH_EN, 139
 XNL_F_QDIR_BOTH, 139
 XNL_F_QDIR_C2H, 139
 XNL_F_QDIR_H2C, 139
 XNL_F_QMODE_MM, 139
 XNL_F_QMODE_ST, 138
 XNL_NAME_PF, 138
 XNL_NAME_VF, 138
 XNL_QIDX_INVALID, 138
 XNL_RESP_BUFLLEN_MAX, 138
 XNL_RESP_BUFLLEN_MIN, 138
 XNL_ST_C2H_CMPT_DESC_SIZE_16B, 142
 XNL_ST_C2H_CMPT_DESC_SIZE_32B, 142
 XNL_ST_C2H_CMPT_DESC_SIZE_64B, 142
 XNL_ST_C2H_CMPT_DESC_SIZE_8B, 142
 XNL_ST_C2H_NUM_CMPT_DESC_SIZES, 142
 XNL_STR_LEN_MAX, 138
 XNL_VERSION, 138
 xnl_attr_t, 141
 xnl_op_t, 142
 xnl_st_c2h_cmpt_desc_size, 141
 qdma_prog_intr_context
 qdma_context.h, 99
 qdma_queue_add
 libqdma_export.h, 89
 qdma_queue_avail_desc
 libqdma_export.h, 94
 qdma_queue_c2h_peek
 libqdma_export.h, 94
 qdma_queue_cmpl_ctrl
 libqdma_export.h, 94
 qdma_queue_conf, 48
 at, 51
 c2h, 49
 c2h_buf_sz_idx, 50
 c2h_bufsz, 53
 c2h_use_fl, 50
 cdh_max, 51
 cmpl_cnt_th_idx, 51
 cmpl_desc_sz, 50
 cmpl_en_intr, 51
 cmpl_ovf_chk_dis, 51
 cmpl_rng_sz_idx, 50
 cmpl_stat_en, 50
 cmpl_status_acc_en, 49
 cmpl_status_en, 49
 cmpl_status_pend_chk, 49
 cmpl_timer_idx, 51
 cmpl_trig_mode, 51
 cmpl_udd_en, 50
 desc_bypass, 50
 desc_rng_sz_idx, 49
 fetch_credit, 50
 fp_descq_c2h_packet, 52
 fp_descq_isr_top, 52
 irq_en, 49
 name, 52
 pfetch_bypass, 51
 pfetch_en, 50
 pipe, 49
 pipe_flow_id, 52

- pipe_gl_max, [52](#)
- pipe_slr_id, [52](#)
- pipe_tdest, [52](#)
- port_id, [51](#)
- qidx, [49](#)
- quld, [52](#)
- rngsz, [53](#)
- rngsz_cmpt, [53](#)
- st, [49](#)
- st_pkt_mode, [50](#)
- sw_desc_sz, [51](#)
- qdma_queue_dump
 - libqdma_export.h, [92](#)
- qdma_queue_dump_cmpt
 - libqdma_export.h, [93](#)
- qdma_queue_dump_desc
 - libqdma_export.h, [92](#)
- qdma_queue_get_config
 - libqdma_export.h, [91](#)
- qdma_queue_list
 - libqdma_export.h, [92](#)
- qdma_queue_packet_read
 - libqdma_export.h, [95](#)
- qdma_queue_packet_write
 - libqdma_export.h, [95](#)
- qdma_queue_prog_stm
 - libqdma_export.h, [91](#)
- qdma_queue_reconfig
 - libqdma_export.h, [89](#)
- qdma_queue_remove
 - libqdma_export.h, [91](#)
- qdma_queue_service
 - libqdma_export.h, [95](#)
- qdma_queue_start
 - libqdma_export.h, [90](#)
- qdma_queue_stop
 - libqdma_export.h, [90](#)
- qdma_req_cb_get
 - qdma_descq.h, [105](#)
- qdma_request, [53](#)
 - count, [54](#)
 - dma_mapped, [54](#)
 - ep_addr, [54](#)
 - fp_done, [54](#)
 - h2c_eot, [54](#)
 - opaque, [54](#)
 - sgcnt, [54](#)
 - sgl, [55](#)
 - timeout_ms, [54](#)
 - udd, [55](#)
 - udd_len, [54](#)
 - uld_data, [54](#)
 - write, [54](#)
- qdma_request_submit
 - libqdma_export.h, [93](#)
- qdma_sdesc_info, [55](#)
 - cidx, [56](#)
 - eop, [56](#)
 - fbits, [55](#)
 - filler, [56](#)
 - next, [55](#)
 - rsvd, [56](#)
 - sop, [56](#)
 - valid, [56](#)
- qdma_set_buf_sz
 - libqdma_config.h, [74](#)
- qdma_set_cmpl_status_acc
 - libqdma_config.h, [73](#)
- qdma_set_cnt_thresh
 - libqdma_config.h, [76](#)
- qdma_set_glbl_rng_sz
 - libqdma_config.h, [75](#)
- qdma_set_intr_rngsz
 - libqdma_config.h, [73](#)
- qdma_set_qmax
 - libqdma_config.h, [72](#)
- qdma_set_timer_cnt
 - libqdma_config.h, [75](#)
- qdma_sgt_req_cb, [56](#)
 - desc_nr, [57](#)
 - done, [58](#)
 - left, [57](#)
 - list, [57](#)
 - offset, [57](#)
 - sg_idx, [57](#)
 - sg_offset, [57](#)
 - status, [58](#)
 - unmap_needed, [58](#)
 - wq, [57](#)
- qdma_sgt_req_done
 - qdma_descq.h, [111](#)
- qdma_software_version_info
 - libqdma_export.h, [88](#)
- qdma_st_c2h.h, [143](#)
 - descq_flq_alloc_resource, [144](#)
 - descq_flq_free_resource, [144](#)
 - descq_process_completion_st_c2h, [145](#)
 - descq_st_c2h_read, [145](#)
 - incr_cmpl_desc_cnt, [144](#)
 - qdma_descq_dump_cmpt, [143](#)
 - qdma_descq_rxq_read, [143](#)
- qdma_sw_sg, [58](#)
 - dma_addr, [59](#)
 - len, [59](#)
 - next, [58](#)
 - offset, [59](#)
 - pg, [58](#)
- qdma_thread.h, [145](#)
 - qdma_thread_add_work, [146](#)
 - qdma_thread_remove_work, [146](#)
 - qdma_threads_create, [146](#)
 - qdma_threads_destroy, [146](#)
- qdma_thread_add_work
 - qdma_thread.h, [146](#)
- qdma_thread_remove_work
 - qdma_thread.h, [146](#)

- qdma_threads_create
 - qdma_thread.h, [146](#)
- qdma_threads_destroy
 - qdma_thread.h, [146](#)
- qdma_version_info, [59](#)
- qdma_wait_queue
 - qdma_compat.h, [98](#)
- qhndl
 - xlnx_qdata, [66](#)
- qid
 - mbox_msg_qctxt, [17](#)
 - qdma_intr_ring, [41](#)
- qidx
 - qdma_queue_conf, [49](#)
- qidx_hw
 - qdma_descq, [28](#)
- qmax
 - mbox_msg_fmap, [13](#)
 - qconf_entry, [18](#)
 - qdma_dev, [33](#)
 - xlnx_pci_dev, [65](#)
- qsets_base
 - qdma_dev_conf, [36](#)
- qsets_max
 - qdma_dev_conf, [34](#)
- quld
 - qdma_queue_conf, [52](#)
- RTL1_VERSION
 - xdev.h, [151](#)
- raw
 - mbox_msg, [12](#)
- req
 - qdma_io_cb, [43](#)
- ring_index_list
 - mbox_msg_intr_ctxt, [15](#)
- ring_sz
 - global_csr_conf, [7](#)
- rngsz
 - qdma_queue_conf, [53](#)
- rngsz_cmpt
 - qdma_queue_conf, [53](#)
- rsvd
 - qdma_desc_cmpl_status, [26](#)
 - qdma_descq, [31](#)
 - qdma_intr_ring, [41](#)
 - qdma_sdesc_info, [56](#)
- rsvd0
 - qdma_mm_desc, [47](#)
- rsvd1
 - qdma_dev_conf, [35](#)
 - qdma_mm_desc, [47](#)
- rsvd2
 - qdma_dev_conf, [34](#)
- rx
 - qdma_mbox, [46](#)
- rx_pend_list
 - qdma_mbox, [46](#)
- rx_work
 - qdma_mbox, [46](#)
- S_MBOX_FN_CMD_RCV
 - qdma_mbox.h, [125](#)
- S_MBOX_FN_CMD_SND
 - qdma_mbox.h, [125](#)
- S_MBOX_FN_CMD_VF_RESET
 - qdma_mbox.h, [126](#)
- S_MBOX_FN_STATUS_ACK
 - qdma_mbox.h, [124](#)
- S_MBOX_FN_STATUS_IN_MSG
 - qdma_mbox.h, [123](#)
- S_MBOX_FN_STATUS_OUT_MSG
 - qdma_mbox.h, [124](#)
- S_MBOX_FN_STATUS_SRC
 - qdma_mbox.h, [124](#)
- S_MBOX_FN_TARGET_ID
 - qdma_mbox.h, [126](#)
- S_MBOX_ISR_EN
 - qdma_mbox.h, [127](#)
- S_MBOX_ISR_VEC
 - qdma_mbox.h, [126](#)
- s_color
 - qdma_intr_ring, [41](#)
- SHIFT_DEC_PCI_BUS
 - libqdma_config.h, [72](#)
- SHIFT_DEC_PCI_DEV
 - libqdma_config.h, [72](#)
- STM_BAR
 - libqdma_config.h, [70](#)
- sdesc
 - qdma_flq, [39](#)
- sdesc_info
 - qdma_flq, [39](#)
- sg_idx
 - qdma_sgt_req_cb, [57](#)
- sg_offset
 - qdma_sgt_req_cb, [57](#)
- sgcnt
 - qdma_request, [54](#)
- sgl
 - qdma_io_cb, [42](#)
 - qdma_request, [55](#)
- sgl_map
 - qdma_descq.h, [111](#)
- sgl_unmap
 - qdma_descq.h, [112](#)
- size
 - qdma_flq, [38](#)
- sop
 - qdma_sdesc_info, [56](#)
- src
 - mbox_msg_hdr, [14](#)
- src_addr
 - qdma_h2c_desc, [40](#)
 - qdma_mm_desc, [47](#)
- st
 - mbox_msg_qctxt, [16](#)
 - qdma_queue_conf, [49](#)

- st_mode_en
 - xlnx_dma_dev, 61
- st_pkt_mode
 - qdma_queue_conf, 50
- status
 - mbox_msg_hdr, 14
 - qdma_sgt_req_cb, 58
- stm
 - stm_descq_context, 60
- stm_descq_context, 59
 - stm, 60
- stm_en
 - xlnx_dma_dev, 61
- stm_regs
 - xlnx_dma_dev, 62
- stm_rev
 - xlnx_dma_dev, 62
- sw
 - hw_descq_context, 8
- sw_desc_sz
 - qdma_queue_conf, 51
- sys_device
 - qdma_cdev, 22
- TOTAL_PF_QS
 - libqdma_config.h, 71
- TOTAL_QDMA_QS
 - libqdma_config.h, 71
- TOTAL_VF_QS
 - libqdma_config.h, 71
- TRIG_MODE_ANY
 - libqdma_export.h, 82
- TRIG_MODE_COMBO
 - libqdma_export.h, 82
- TRIG_MODE_COUNTER
 - libqdma_export.h, 82
- TRIG_MODE_DISABLE
 - libqdma_export.h, 82
- TRIG_MODE_TIMER
 - libqdma_export.h, 82
- TRIG_MODE_USER
 - libqdma_export.h, 82
- task
 - qdma_kthread, 44
- thread.h, 147
 - lock_thread, 147
 - pr_debug_thread, 148
 - qdma_kthread_dump, 148
 - qdma_kthread_start, 148
 - qdma_kthread_stop, 149
 - qdma_kthread_wakeup, 148
 - unlock_thread, 147
- tigger_mode_t
 - libqdma_export.h, 82
- timeout
 - qdma_kthread, 44
- timeout_ms
 - qdma_request, 54
- timer
 - qdma_mbox, 46
- timer_idx
 - qdma_cmpl_ctrl, 24
- tm_mode_en
 - qdma_dev_conf, 37
- tm_one_cdh_en
 - qdma_dev_conf, 37
- total_cmpl_descs
 - qdma_descq, 31
- total_st_c2h_pkts
 - xlnx_dma_dev, 63
- trigger_mode
 - qdma_cmpl_ctrl, 24
- tx_todo_list
 - qdma_mbox, 46
- tx_work
 - qdma_mbox, 46
- type
 - qconf_entry, 18
- udd
 - qdma_request, 55
- udd_cnt
 - qdma_descq, 31
 - qdma_flq, 38
- udd_len
 - qdma_request, 54
- uld
 - qdma_dev_conf, 35
- uld_data
 - qdma_request, 54
- unlock_descq
 - qdma_descq.h, 105
- unlock_thread
 - thread.h, 147
- unmap_needed
 - qdma_sgt_req_cb, 58
- user_bar_regs
 - xlnx_pci_dev, 65
- V_MBOX_FN_TARGET_ID
 - qdma_mbox.h, 126
- V_MBOX_ISR_VEC
 - qdma_mbox.h, 126
- valid
 - qdma_sdesc_info, 56
- vec_base
 - mbox_msg_intr_ctxt, 15
- vec_id
 - intr_coal_conf, 9
- verify
 - mbox_msg_qctxt, 16
- vf_cnt
 - qconf_entry_head, 19
- vf_count
 - xlnx_dma_dev, 62
- vf_free_list
 - qconf_entry_head, 19
- vf_info

- xlnx_dma_dev, [61](#)
- vf_list
 - qconf_entry_head, [19](#)
- vf_max
 - qdma_dev_conf, [35](#)
- vf_qbase
 - qconf_entry_head, [19](#)
- vf_qmax
 - qconf_entry_head, [19](#)
- w
 - mbox_msg_intr_ctxt, [15](#)
- waitq
 - mbox_msg, [11](#)
 - qdma_kthread, [44](#)
- work
 - qdma_descq, [29](#)
- work_cnt
 - qdma_kthread, [44](#)
- work_list
 - qdma_descq, [29](#)
 - qdma_kthread, [44](#)
- workq
 - qdma_mbox, [46](#)
- wq
 - qdma_sgt_req_cb, [57](#)
- write
 - qdma_request, [54](#)
- XDEV_FLAG_IRQ
 - xdev.h, [151](#)
- XDEV_FLAG_OFFLINE
 - xdev.h, [151](#)
- XDEV_NUM_IRQ_MAX
 - xdev.h, [151](#)
- XNL_ATTR_C2H_BUFSZ_IDX
 - qdma_nl.h, [141](#)
- XNL_ATTR_CMPT_CNTR_IDX
 - qdma_nl.h, [141](#)
- XNL_ATTR_CMPT_DESC_SIZE
 - qdma_nl.h, [141](#)
- XNL_ATTR_CMPT_TIMER_IDX
 - qdma_nl.h, [141](#)
- XNL_ATTR_CMPT_TRIG_MODE
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_CFG_BAR
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_IDX
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_QSET_MAX
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_QSET_QBASE
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_STAT_MMC2H_PKTS1
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_STAT_MMC2H_PKTS2
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_STAT_MMH2C_PKTS1
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_STAT_MMH2C_PKTS2
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_STAT_STC2H_PKTS1
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_STAT_STC2H_PKTS2
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_STAT_STH2C_PKTS1
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_STAT_STH2C_PKTS2
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_STM_BAR
 - qdma_nl.h, [141](#)
- XNL_ATTR_DEV_USR_BAR
 - qdma_nl.h, [141](#)
- XNL_ATTR_DRV_INFO
 - qdma_nl.h, [141](#)
- XNL_ATTR_GENMSG
 - qdma_nl.h, [141](#)
- XNL_ATTR_GLOBAL_CSR
 - qdma_nl.h, [141](#)
- XNL_ATTR_INTR_VECTOR_END_IDX
 - qdma_nl.h, [141](#)
- XNL_ATTR_INTR_VECTOR_IDX
 - qdma_nl.h, [141](#)
- XNL_ATTR_INTR_VECTOR_START_IDX
 - qdma_nl.h, [141](#)
- XNL_ATTR_NUM_Q
 - qdma_nl.h, [141](#)
- XNL_ATTR_PCI_BUS
 - qdma_nl.h, [141](#)
- XNL_ATTR_PCI_DEV
 - qdma_nl.h, [141](#)
- XNL_ATTR_PCI_FUNC
 - qdma_nl.h, [141](#)
- XNL_ATTR_PIPE_FLOW_ID
 - qdma_nl.h, [141](#)
- XNL_ATTR_PIPE_GL_MAX
 - qdma_nl.h, [141](#)
- XNL_ATTR_PIPE_SLR_ID
 - qdma_nl.h, [141](#)
- XNL_ATTR_PIPE_TDEST
 - qdma_nl.h, [141](#)
- XNL_ATTR_QFLAG
 - qdma_nl.h, [141](#)
- XNL_ATTR_QIDX
 - qdma_nl.h, [141](#)
- XNL_ATTR_QRNGSZ_IDX
 - qdma_nl.h, [141](#)
- XNL_ATTR_RANGE_END
 - qdma_nl.h, [141](#)
- XNL_ATTR_RANGE_START
 - qdma_nl.h, [141](#)
- XNL_ATTR_REG_ADDR
 - qdma_nl.h, [141](#)
- XNL_ATTR_REG_BAR_NUM
 - qdma_nl.h, [141](#)
- XNL_ATTR_REG_VAL
 - qdma_nl.h, [141](#)

XNL_ATTR_RSP_BUF_LEN
 qdma_nl.h, [141](#)
 XNL_ATTR_SW_DESC_SIZE
 qdma_nl.h, [141](#)
 XNL_CMD_DEV_INFO
 qdma_nl.h, [142](#)
 XNL_CMD_DEV_LIST
 qdma_nl.h, [142](#)
 XNL_CMD_DEV_STAT_CLEAR
 qdma_nl.h, [142](#)
 XNL_CMD_DEV_STAT
 qdma_nl.h, [142](#)
 XNL_CMD_GLOBAL_CSR
 qdma_nl.h, [142](#)
 XNL_CMD_INTR_RING_DUMP
 qdma_nl.h, [142](#)
 XNL_CMD_MAX
 qdma_nl.h, [142](#)
 XNL_CMD_Q_ADD
 qdma_nl.h, [142](#)
 XNL_CMD_Q_CMPT
 qdma_nl.h, [142](#)
 XNL_CMD_Q_DESC
 qdma_nl.h, [142](#)
 XNL_CMD_Q_DEL
 qdma_nl.h, [142](#)
 XNL_CMD_Q_DUMP
 qdma_nl.h, [142](#)
 XNL_CMD_Q_LIST
 qdma_nl.h, [142](#)
 XNL_CMD_Q_RX_PKT
 qdma_nl.h, [142](#)
 XNL_CMD_Q_START
 qdma_nl.h, [142](#)
 XNL_CMD_Q_STOP
 qdma_nl.h, [142](#)
 XNL_CMD_Q_UDD
 qdma_nl.h, [142](#)
 XNL_CMD_REG_DUMP
 qdma_nl.h, [142](#)
 XNL_CMD_REG_RD
 qdma_nl.h, [142](#)
 XNL_CMD_REG_WRT
 qdma_nl.h, [142](#)
 XNL_CMD_VERSION
 qdma_nl.h, [142](#)
 XNL_ERR_BUFLLEN
 qdma_nl.h, [138](#)
 XNL_F_C2H_CMPL_INTR_EN
 qdma_nl.h, [140](#)
 XNL_F_CMPL_STATUS_ACC_EN
 qdma_nl.h, [139](#)
 XNL_F_CMPL_STATUS_DESC_EN
 qdma_nl.h, [140](#)
 XNL_F_CMPL_STATUS_EN
 qdma_nl.h, [139](#)
 XNL_F_CMPL_STATUS_PEND_CHK
 qdma_nl.h, [139](#)
 XNL_F_CMPL_UDD_EN
 qdma_nl.h, [140](#)
 XNL_F_CMPT_OVF_CHK_DIS
 qdma_nl.h, [140](#)
 XNL_F_DESC_BYPASS_EN
 qdma_nl.h, [139](#)
 XNL_F_FETCH_CREDIT
 qdma_nl.h, [139](#)
 XNL_F_PFETCH_BYPASS_EN
 qdma_nl.h, [140](#)
 XNL_F_PFETCH_EN
 qdma_nl.h, [139](#)
 XNL_F_QDIR_BOTH
 qdma_nl.h, [139](#)
 XNL_F_QDIR_C2H
 qdma_nl.h, [139](#)
 XNL_F_QDIR_H2C
 qdma_nl.h, [139](#)
 XNL_F_QMODE_MM
 qdma_nl.h, [139](#)
 XNL_F_QMODE_ST
 qdma_nl.h, [138](#)
 XNL_NAME_PF
 qdma_nl.h, [138](#)
 XNL_NAME_VF
 qdma_nl.h, [138](#)
 XNL_QIDX_INVALID
 qdma_nl.h, [138](#)
 XNL_RESP_BUFLLEN_MAX
 qdma_nl.h, [138](#)
 XNL_RESP_BUFLLEN_MIN
 qdma_nl.h, [138](#)
 XNL_ST_C2H_CMPT_DESC_SIZE_16B
 qdma_nl.h, [142](#)
 XNL_ST_C2H_CMPT_DESC_SIZE_32B
 qdma_nl.h, [142](#)
 XNL_ST_C2H_CMPT_DESC_SIZE_64B
 qdma_nl.h, [142](#)
 XNL_ST_C2H_CMPT_DESC_SIZE_8B
 qdma_nl.h, [142](#)
 XNL_ST_C2H_NUM_CMPT_DESC_SIZES
 qdma_nl.h, [142](#)
 XNL_STR_LEN_MAX
 qdma_nl.h, [138](#)
 XNL_VERSION
 qdma_nl.h, [138](#)
 xcb
 qdma_cdev, [22](#)
 xcdev
 xlrx_qdata, [66](#)
 xdev
 qdma_descq, [28](#)
 qdma_mbox, [46](#)
 xdev.h, [149](#)
 f_intr_handler, [152](#)
 INTR_TYPE_DATA, [152](#)
 INTR_TYPE_ERROR, [152](#)
 INTR_TYPE_MAX, [152](#)

- INTR_TYPE_USER, 152
- intr_type_list, 152
- PCI_DMA_H, 151
- PCI_DMA_L, 151
- QDMA_BAR_NUM, 151
- QDMA_DESC_BLEN_BITS, 151
- QDMA_DESC_BLEN_MAX, 151
- QDMA_MAX_BAR_LEN_MAPPED, 151
- RTL1_VERSION, 151
- XDEV_FLAG_IRQ, 151
- XDEV_FLAG_OFFLINE, 151
- XDEV_NUM_IRQ_MAX, 151
- xdev_check_hndl, 154
- xdev_find_by_idx, 153
- xdev_find_by_pdev, 152
- xdev_list_dump, 153
- xdev_list_first, 153
- xdev_list_next, 153
- xdev_sriov_disable, 151
- xdev_sriov_enable, 152
- xdev_sriov_vf_offline, 152
- xdev_sriov_vf_online, 152
- xdev_2_qdev
 - qdma_device.h, 113
- xdev_check_hndl
 - xdev.h, 154
- xdev_find_by_idx
 - xdev.h, 153
- xdev_find_by_pdev
 - xdev.h, 152
- xdev_list_dump
 - xdev.h, 153
- xdev_list_first
 - xdev.h, 153
- xdev_list_next
 - xdev.h, 153
- xdev_sriov_disable
 - xdev.h, 151
- xdev_sriov_enable
 - xdev.h, 152
- xdev_sriov_vf_offline
 - xdev.h, 152
- xdev_sriov_vf_online
 - xdev.h, 152
- xlrx_dma_dev, 60
 - conf, 61
 - dev_intr_info_list, 62
 - dev_priv, 62
 - dvec_start_idx, 62
 - flr_prsnt, 61
 - func_id, 62
 - hw_prg_lock, 61
 - intr_coal_list, 63
 - list_head, 61
 - lock, 61
 - mbox, 63
 - mm_mode_en, 61
 - mod_name, 61
 - msix, 62
 - num_vecs, 62
 - pf_count, 62
 - pipe_stm_max_pkt_size, 63
 - st_mode_en, 61
 - stm_en, 61
 - stm_regs, 62
 - stm_rev, 62
 - total_st_c2h_pkts, 63
 - vf_count, 62
 - vf_info, 61
- xlrx_nl_work, 63
- xlrx_nl_work_q_ctrl, 64
- xlrx_pci_dev, 64
 - bypass_bar_regs, 65
 - cdev_cb, 65
 - cdev_lock, 65
 - dev_hndl, 64
 - idx, 65
 - list_head, 64
 - nl_task_wq, 65
 - pdev, 64
 - qdata, 65
 - qmax, 65
 - user_bar_regs, 65
- xlrx_qdata, 66
 - qhndl, 66
 - xcdev, 66
- xnl_attr_t
 - qdma_nl.h, 141
- xnl_op_t
 - qdma_nl.h, 142
- xnl_respond_buffer
 - nl.h, 97
- xnl_st_c2h_cmpt_desc_size
 - qdma_nl.h, 141
- xpdev
 - qdma_cdev_cb, 23
- xpdev_find_by_idx
 - qdma_mod.h, 131
- xpdev_list_dump
 - qdma_mod.h, 131
- xpdev_queue_add
 - qdma_mod.h, 132
- xpdev_queue_delete
 - qdma_mod.h, 132
- xpdev_queue_get
 - qdma_mod.h, 131
- xreg_info, 66
- zerolen_dma
 - qdma_dev_conf, 35