OpenCAPI 3.1 Transaction Layer TX (TLXT) Workbook

Version 1.0

TLXT Reference Design for the OpenCAPI 3.1 Transaction Layer Specification from OpenCAPI Consortium:
https://opencapi.org/technical/specifications/

Used in Open Memory Interface (OMI) ASIC Devices

1 OpenCAPI 3.1 Transaction Layer TX (TLXT) Workbook

1.1 Functional Description

TLX TX component handles the encoding of command responses and data into OpenCAPI frames in accordance with the OpenCAPI v3.1 spec Memory interface class profile. This logic sits between the memory sequencer and read data flow, and the Data link Layer (DLX). The TLX TX acts as the OpenCAPI TLX framer, packing up control flits and controlling the flow of data to the DLX from the read data flow. This component also contains the TLX credit management.

1.2 VHDL Diagram

1.2.1 Logic Block Descriptions

cb_tlxt_mac – Top level tlxt macro.

cb_tlxt_flit_arb_rlm – Contains read, write, and mmio response fifos as well as a meta bit fifo and bad data bit fifo. Controls packing control flits and calculating data run length, optimal template selection, and when to send control flits.

cb_tlxt_trans_arb_rlm – Arbiter in control of transmitting control and data flits to the DLX. Contains a 32B wide data buffer for splitting octword read responses to reduce latency.

cb_tlxt_crd_mgmt_rlm – OpenCAPI credit management for the TLX layer. Contains 8 Credit counters.

cb_tlxt_regs – Contains all config registers and error checking registers for the TLXR and TLXT block.

cb_tlxt_dbg - Contains trace (debug) bus multiplexing and output register.

cb_gp_fifo - Fifo (size set by generics), used multiple times by cb_tlxt_flit_arb_rlm.

1.3 Interfaces

Signal Name	I/O	Size	Description	Clock Domain	Endian
Interface with DLX					
tlxt_dlx_flit_vld	O	1	Indicates this cycle has valid **tlxt_dlx_flit_data** and **tlxt_dlx_flit_ecc**	gckn	
tlxt_dlx_flit_data	O	128	Partial flit data (16 bytes of 64 bytes flit).	gckn	Little
tlxt_dlx_flit_ecc	O	16	ECC protecting **tlxt_dlx_flit_data** (8 bits per 64 bits of flit data).	gckn	Little

Signal Name	I/O	Size	Description	Clock Domain	Endian
tlxt_dlx_flit_lbip_vld	O	1	Indicates this cycle has valid last beat in parallel data and ecc.	gckn	
tlxt_dlx_flit_lbip_data	O	82	Last beat in parallel data (16 bytes of the 64 bytes flit) last beat of a control flit.	gckn	Little
tlxt_dlx_flit_lbip_ecc	O	16	ECC protecting ***tlxt_dlx_flit_lbip_data*** (8 bits per 64 bits of flit data)	gckn	Little
tlxt_dlx_flit_early_vld	O	1	One cycle early valid indication	gckn	
tlxt_dlx_tl_error	O	1		gckn	
tlxt_dlx_tl_event	O	1		gckn	
dlx_tlxr_link_up	I	1	Indicates that the HSS link is trained and ready to transmit and receive packets. Triggers the initial credit return to the host.	gckn	
dlx_tlxt_flit_credit	I	1	Local interface credit returned to the TLX. Each credit is for 1 16B flit partial. Used for back-pressure during replay or error cases.	gckn	
Interface with SRQ					
srq_tlxt_cmdq_release	I	1	Indicates that the sequencer has pulled a response from the new cmd fifo and a credit can be released back to the TL on the host.	gckn	
tlxt_srq_rdbuf_pop	O	1	Pulsed when one read response of data has been sent to the DLX.	gckn	
srq_tlxt_padmem_done_val	I	1	Padmem response is valid this cycle. Stays 1 until *tlxt_srq_padmem_done_ack* is asserted.	gckn	
srq_tlxt_padmem_done_tag	I	16	Tag associated with a padmem done or fail response. Valid when *srq_tlxt_padmem_done_val* or *srq_tlxt_failresp_val* is asserted.	gckn	big
srq_tlxt_padmem_done_tag_p	I	2	Even Parity on each byte of ***s****rq_tlxt_padmem_done_tag****. V****alid when *srq_tlxt_padmem_done_val* or *srq_tlxt_failresp_val* is asserted.	gckn	big
tlxt_srq_padmem_done_ack	O	1	Ack to SRQ to indicate that a padmem or fail response tag has been pushed into it's respective FIFO. TLXT can now accept another response	gckn	
srq_tlxt_failresp_val	I	1	Fail response is valid on this cycle. TLXT uses *srq_tlxt_padmem_done_tag* to get the tag for this response	gckn	
srq_tlxt_failresp_dlen	I	2	Data length associated with the fail response. Valid when *srq_tlxt_failresp_val* is asserted.	gckn	big
srq_tlxt_failresp_code	I	4	Response code associated with the fail response. Valid when *srq_tlxt_failresp_val* is asserted.	gckn	big
Interface with MMIO/CFG					

Signal Name	I/O	Size	Description	Clock Domain	Endian
mmio_tlxt_resp_valid	I	1	Response valid. This must be held active until ack is received. Then must be de-asserted for at least one cycle.	gckn	
mmio_tlxt_resp_opcode	I	8	Response opcode. This will either be x01 (mem_rd_response) for a successful read or x02 (mem_rd_fail) for a failed read. Valid when mmio_tlxt_resp_valid=1b.	gckn	Little
mmio_tlxt_resp_dl	I	2	Response Data Length. Always 0b01 to indicate a 64 byte response Valid when mmio_tlxt_resp_valid=1b.	gckn	Little
mmio_tlxt_resp_capptag	I	16	Response tag. Valid when mmio_tlxt_resp_valid=1b.	gckn	Little
mmio_tlxt_resp_dp	I	2	Response Data Part. Always 0b00 to indicate 0 byte offset. Valid when mmio_tlxt_resp_valid=1b.	gckn	Little
mmio_tlxt_resp_code	I	4	Response code. In the event of a command fail, this bus indicates the cause of the failure. MMIO returns 0b0000 for successful operation. Valid when mmio_tlxt_resp_valid=1b.	gckn	Little
tlxt_mmio_resp_ack	O	1	TLXT signal to the MMIO engine that the response has been pushed into queue and another can be received	gckn	
mmio_tlxt_rdata_offset	I	4	Response valid. This must be held active until ack is received. Then must be de-asserted for at least one cycle.	gckn	Little
mmio_tlxt_rdata_bus	I	32	Read data. 255:0 – read data 256 – even parity over bits 7:0 257 – even parity over bits 15:8 ... 287 – even parity over bits 255:248	gckn	Little
mmio_tlxt_rdata_bdi	I	1	Bad Data Indication. Valid when mmio_tlxt_resp_valid=1b.	gckn	
mmio_tlxt_resp_par	I	1	Even parity over mmio_tlxt_resp_valid, mmio_tlxt_resp_opcode, mmio_tlxt_resp_dl, mmio_tlxt_resp_capptag, mmio_tlxt_resp_dp, mmio_tlxt_resp_code, mmio_tlxt_rdata_offset & mmio_tlxt_rdata_bdi	gckn	
mmio_tlxt_busnum	I	9	Bus number discovered when Host sends config_write to the OMI ASIC Bit 8 is even parity over bits 7:0.	gckn	Little
OpenCAPI Configuration					
cfg_otl0_tl_minor_vers_config	I	1	OpenCAPI Minor version.		
cfg_otl0_tl_minor_vers_config_p	I	1	Parity on cfg_otl0_tl_minor_vers_config		

Signal Name	I/O	Size	Description	Clock Domain	Endian
cfg_otl0_tl_xmt_tmpl_config	I	13	OpenCAPI Transmit Template Configuration. Bit index corresponds to template number. Bit 12 is even parity on 11:0		
cfg_otl0_tl_xmt_rate_tmpl_config	I	49	OpenCAPI Transmit Rate Configuration. 4 bits for each template number. Th rate corresponds to how many idle flit cycles or data flits there can be between control flits with the specified tempalte. 3:0 → rate for template 0 7:4 → rate for template 1 11:8 → rate for template 2 15:12 → rate for template 3 19:16 → rate for template 4 23:20 → rate for template 5 27:24 → rate for template 6 31:28 → rate for template 7 35:32 → rate for template 8 39:36 → rate for template 9 43:40 → rate for template A 47:44 → rate for template B 48 → even parity for 47:0	gckn	Little
cfg_f1_octrl00_enable_afu	I	1	AFU enabled	gckn	
cfg_f1_octrl00_metadata_enabled	I	1	Metadata enabled	gckn	
cfg_f1_octrl00_p	I	1	Even Parity over cfg1_f1_octrl00_enable_afu, cfg_f1_octrl00_metadata_enabled	gckn	
cfg_f1_ofunc_func_actag_base	I	13	Base value for OpenCAPI AcTag. Bit 12 is even parity over 11:0	gckn	Little
cfg_f1_ofunc_func_actag_len_enab	I	13	Length for AC Tag. Bit 12 is even parity over 11:0.	gckn	Little
cfg_f1_octrl00_pasid_length_enabled	I	6	Length for PASID enabled. Bit 5 is parity over 4:0.	gckn	Little
cfg_f1_octrl00_pasid_base	I	21	Base value for OpenCAPI PASID. Bit 20 is even parity over 19:0.	gckn	Little
Interface with RDF					
rdf_tlxt_resp_valid	I	1	**rdf_tlxt_resp_dpart** and **rdf_tlxt_resp_otag** valid this cycle	gckn	
rdf_tlxt_resp_dPart	I	2	Indicates byte offset of 64B read data sent back to the host. Populates dPart section of OpenCAPI read response 00 → 0B offset 01 → 64B offset	gckn	Big

Signal Name	I/O	Size	Description	Clock Domain	Endian
rdf_tlxt_resp_Otag	I	16	OpenCAPI Tag associated with the 64B read response. Valid when rdf_tlxt_resp_valid is asserted.	gckn	Big
rdf_tlxt_data_valid	I	1	**rdf_tlxt_data** and **rdf_tlxt_dat_ecc** are valid this cycle.	gckn	
tlxt_rdf_data_taken	O	1	Data in rdf_tlxt_data has been taken. TLXT ready for another 16B of data. Can only be asserted if rdf_tlxt_data_valid is asserted.	gckn	
rdf_tlxt_data	I	128	16B read data from RDF. Valid when rdf_tlxt_data_valid is asserted.	gckn	Big
rdf_tlxt_dat_ecc	I	16	ECC for data present on rdf_tlxt_data bus. Valid when rdf_tlxt_data_valid is asserted.	gckn	Big
rdf_tlxt_bad_data_valid	I	1	**rdf_tlxt_bad_data** and **rdf_tlxt_bad_data_first_32B** are valid this cycle. Write bad data into fifo or pass into flit	gckn	Big
rdf_tlxt_bad_data	I	1	Indicates that data in the read buffer has an uncorrectable error and should be thrown out by the host	gckn	Big
rdf_tlxt_bad_data_1st32B	I	1	Bad data indication for the first half of a read, used for bypass read cases.	gckn	
rdf_tlxt_bad_data_p	I	1	Parity for **rdf_tlxt_bad_data_valid**, **rdf_tlxt_bad_data**, and **rdf_tlxt_bad_data_first_32B**	gckn	
rdf_tlxt_meta_valid	I	2	Bit 1 means ***rdf_tlxt_meta***(5:3) is valid, bit 0 means ***rdf_tlxt_meta***(2:0) is valid	gckn	Big
rdf_tlxt_meta	I	6	6 bits(meta for 64 bytes of data, 64 bytes aligned) 5 qw0 tag, 4 qw1 tag, 3 MDI 2 qw0 tag, 1 qw1 tag, 0 MDI	gckn	Big
rdf_tlxt_meta_p	1	1	Meta Bit Parity Even parity over rdf_tlxt_meta.	gckn	
Interface with TLXR					
tlxr_tlxt_write_resp	I	22	Write done response from tlxr. 21:6 → CAPP Tag for response 5:2 → error response code 1:0 → Data length for response. Valid when 01b or 10b. 00b indicates mem_cntl response	gckn	Little
tlxr_tlxt_write_resp_p	I	3	Even Parity on the response and valid right justified. Bit 2 covers tlxr_tlxt_write_resp(21:14) Bit 1 covers tlxr_tlxt_write_resp(13:6) Bit 0 covers tlxr_tlxt_write_resp(5:0) and tlxr_tlxt_write_resp_val.	gckn	Little
tlxr_tlxt_write_resp_val	I	1	tlxr_tlxt_write_resp is valid on this cycle.	gckn	

Signal Name	I/O	Size	Description	Clock Domain	Endian
tlxt_tlxr_write_resp_full	I	1	Write response queue is full or a padmem response is being pushed into the queue.	gckn	
tlxr_tlxt_intrp_resp	I	8	2 bits for each type of expected interrupt response. 7:6 → Application Interrupt (tag 4) 5:4 → Special Attention Interrupt (tag 3) 3:2 → Recoverable Attention Interrupt (tag 2) 1:0 → Channel Checkstop Interrupt (tag 1) 1:0 is for tag 1 ……. 7:6 is for tag 4 Each two bit field is coded: “01” - good completion “10” - retry “11” - fail “00” - inactive	gckn	
tlxr_tlxt_vc0_release	I	2	Pulse. Release credit for mem_cntl, intrp_rdy and intrp_resp.	gckn	
tlxr_tlxt_vc1_release	I	1	Pulse Release credit for pr_wr_mem into padmem data buffer.	gckn	
tlxt_tlxr_dcp1_release	I	3	DCP1 credit release pulse.	gckn	
tlxr_tlxt_consume_vc0	I	1	Error Checking	gckn	
tlxr_tlxt_consume_vc1	I	1	Error Checking	gckn	
tlxr_tlxt_consume_dcp1	I	3	Error Checking	gckn	
tlxr_tlxt_return_val	I	1	Indicates that credits have been returned from the TL. Add number of credits indicated by ***tlxr_tlxc_credits_vc**** and ***tlxr_tlxc_credits_dcp**** signals to respective counters.	gckn	
tlxr_tlxt_return_vc0	I	4	Number of credits returned to virtual channel 0 on receipt of a ***return_tlx_credits*** comand from the host.	gckn	Little
tlxr_tlxt_return_vc3	I	4	Number of credits returned to virtual channel 3 on receipt of a ***return_tlx_credits*** comand from the host.	gckn	Little
tlxr_tlxt_return_dcp0	I	6	Number of credits returned to Data credit Pool 0 on receipt of a ***return_tlx_credits*** comand from the host.	gckn	Little
tlxt_tlxr_early_wdone_disable	O	2	These bits disable the early write done mechanism so that write responses are only sent once the write has completed at the dram.	gckn	Little
tlxt_tlxr_ctrl	O	16	These register bits control the debug bus and chicken switches in tlxr. See TLXR chapter for more details	gckn	Little
tlxr_tlxt_errors	I	64	Error pulses for tlxr internal errors. See TLXR chapter for more details.	gckn	Little
tlxr_tlxt_signature_dat	I	64	OpenCAPI error signature. Walid when tlxr_tlxt_signature_strobe=1b.	gckn	Little

Signal Name	I/O	Size	Description	Clock Domain	Endian
tlxr_tlxt_signature_strobe	I	1	Strobe to load error signature into SCOM register.	gckn	
tlxr_tlxt_return_dcp3	I	6	Number of credits returned to Data credit Pool 3 on receipt of a ***return_tlx_credits*** comand from the host.	gckn	Little
Interface with SCOM, Trace and Global FIR					
tcm_tlxt_scom_cch	I	1	SCOM ring in	gckn	
tcm_tlxt_scom_dch	I	1	SCOM ring in	gckn	
tlxt_tcm_scom_cch	O	1	SCOM ring out	gckn	
tlxt_tcm_scom_dch	O	1	SCOM ring out	gckn	
tlxt_dbg_debug_bus	O	88	Trace bus to trace multiplexer	gckn	Big
tlx_xstop_err	O	1	TLXT/TLXR fir checkstop	gckn	
tlx_recov_err	O	1	TLXR/TLXT fir recoverable error	gckn	
tlx_recov_int	O	1	TLXR/TLXT fir recoverable interrupt	gckn	
tlx_mchk_out	O	1	TLXR/TLXT fir application interrupt	gckn	
global_fir_chan_xstop	I	1	Global fir checkstop	gckn	
global_fir_rec_attn	I	1	Global fir recoverable attention	gckn	
global_fir_sp_attn	I	1	Global fir special attention	gckn	
global_fir_mchk	I	1	Global fir application interrupt	gckn	

1.4 Design Restrictions

1.4.1 Bit Endianness

All TLXT logic is little endian. The TLXT switches bit and byte endianness in the upstream path of the OMI ASIC. All interfaces busses incoming and outgoing to the Sequencer and Read Data Flow are big endian. All interface busses incoming and outgoing to the DLX layer and TLXR are little endian. The endianness switch occurs as close to the boundary to the Read data flow and sequencer as necessary to ensure that TL response packets and data have the correct endianness in accordance with the OpenCAPI TL spec.

1.4.2 Read Response Length

The assumption of data length for a read response received from the RDF changes depending on the mode that the TLXT is operating in. If in half-dimm mode the assumption is that every read response received from the RDF represents 40B of data to be returned to the host. How this data is returned is described in section 1.5.5.3 below.

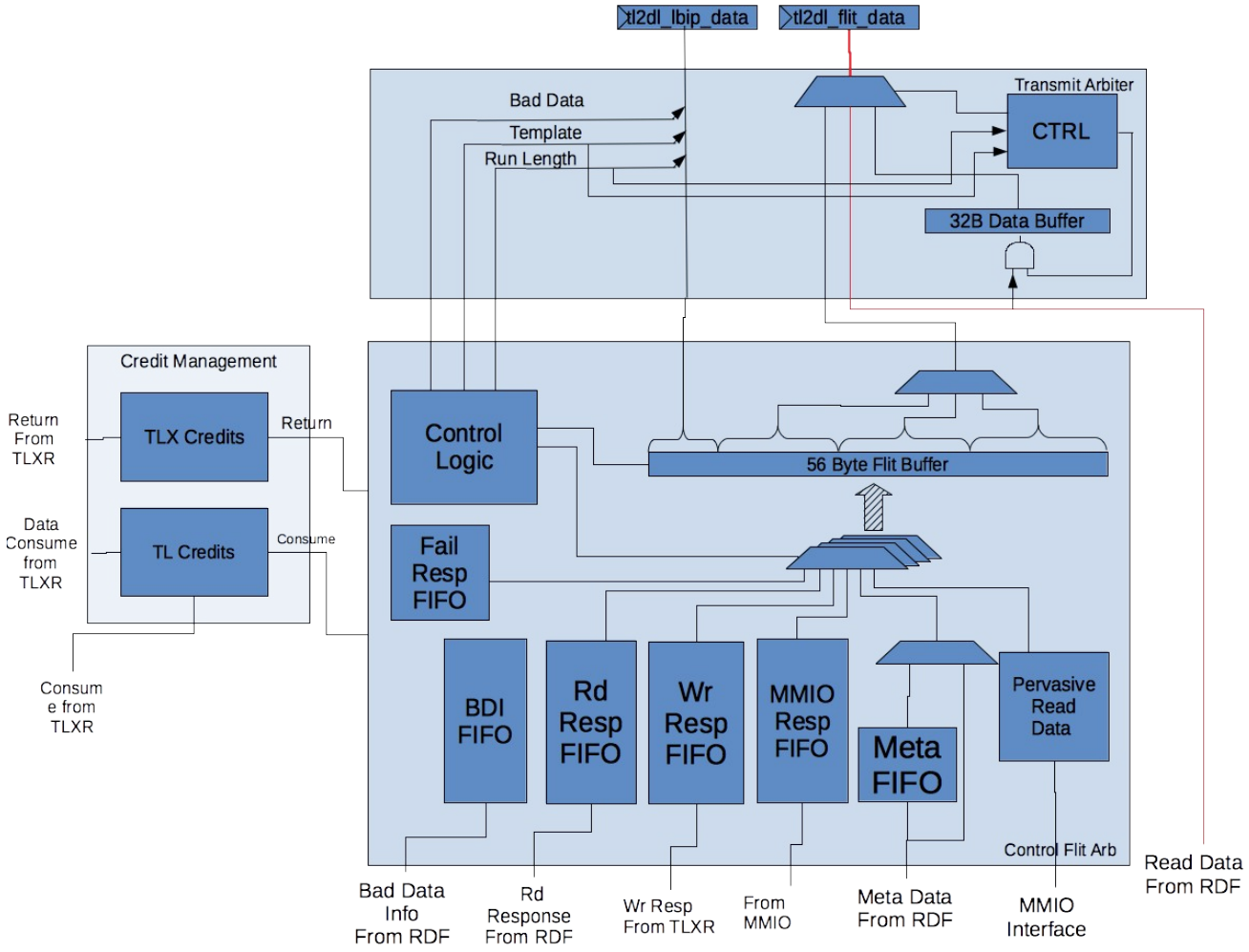
In all other modes each read response received from the RDF is assumed to represent 64B of data to be returned to the host. How this data is returned is described in sections 1.5.5.1 and 1.5.5.2 below.

1.4.3 Parity Protection

Partity on all Interfaces listed in Section 1.3 is even.

1.5 Design Details

1.5.1 Unit Data Flow



1.5.2 Choosing a Template

The OMI ASIC supports templates in the OpenCAPI 3.1 Memory interface class of the TL spec. The OMI ASIC supports transmitting control flits with template 0x00, 0x01, 0x05, 0x09, and 0x0B. Templates are selected based on the number and types of responses that are queued in the TLXT, pending data from the RDF, and enabled templates.

1.5.3 Credit Management

The TLXT contains a TL credit management block which is compliant with the OpenCAPI 3.1 TL specification.

There are a total of eight 16b counters within the credit management block, four each for TL and TLX credit count. The maximum number of TLX credits is set by the host during config time and TL credits are set by the OMI ASIC during config time. This value is also sent to the host TL. TLX virtual channel and data credits are consumed by the OMI ASIC when sending responses and response data back to the host and those credits are returned via the TLXR downstream interface. TL credits are consumed by the host with downstream commands and returned to the host via a command through the TLXT.

1.5.3.1 Credit Consumption by Host TL

The OMI ASIC maintains three TL credit pools, TL.VC.0, TL.VC.1 and TL.DCP.1. They represent the OMI ASIC resources that are consumed by the host with downstream commands and released back to their corresponding credit pool once the OMI ASIC resource free up. TLXT periodically returns credits to host from the credit pools by a return_tl_credits command. TLXT can return up-to 15 vc.0/vc.1 and 63 dcp.1 credits at a time from each credit pool to host.

TL.VC.0 : The OMI ASIC initializes TL.VC.0 credit pool to 4 and TLXT returns 4 VC.0 credits from the credit pool through return_tl_credits command to host after DL link up and periodically returns the TL.VC.0 credits to host thereafter as long as there is available credit in the TL.VC.0 credit pool. One TL.VC.0 credit is consumed by host with each **intrp_resp or mem_cntl or intrp_rdy** command packets to the OMI ASIC. TLXR releases TL.VC.0 credit for intrp_resp or intrp_rdy command back to TL.VC.0 credit pool as soon as the command flit CRC check is good. TLXR releases TL.VC.0 credit for mem_cntrl when the response tag is passed to TLXT.

TL.VC.1 : The OMI ASIC initializes TL.VC.1 credit pool to 30 and TLXT returns 30 VC.1 credits from the credit pool through return_tl_credits command to host after DL link up and periodically returns the TL.VC.1 credits to host thereafter as long as there is available credit in the TL.VC.1 credit pool. TL.VC.1 credit is consumed by host with **config_read, config_write, pad_mem, pr_rd_mem, pr_wr_mem, rd_mem or write_mem or write_mem.be** command packets to the OMI ASIC. TLXR releases TL.VC.1 credit for pr_wr_mem(pad mem pattern) command back to TL.VC.1 credit pool as soon as the command flit CRC check is good. TLXR also releases TL.VC.1 credits back to TL.VC.1 credit pool for any unsupported write commands except pad_mem. For all other supported writes, reads, pad_mem commands and unsupported reads, pad_mem commands, TLXR pushes command to New Command FIFO (NCF). NCF releases TL.VC.1 credits back to TL.VC.1 credit pool as soon as the command leaves NCF. NCF is 32-entry deep. (note: Two entries are reserved for MCBIST. Refer to NCF section of SRQ document for detail)

TL.DCP.1 : The OMI ASIC initializes TL.DCP.1 credit pool to 63 in cfg_half_dimm_mode and initializes TL.DCP.1 credit pool to 64 in not cfg_half_dimm_mode the cycle after reset or switching events of cfg_half_dimm_mode (note: cfg_half_dimm_mode switch event shall only happen before DL link up). TLXT returns up to max of 63 DCP.1 credit from the credit pool at a time through multiple return_tl_credits commands to host after DL link up until DCP.1 credit pool reaches 0 and periodically returns the TL.DCP.1 credits to host thereafter as long as there is available credit in the TL.DCP.1 credit pool. One or two TL.DCP.1 credits are consumed by host with each **config_write, pr_wr_mem, write_mem or write_mem.be** command packets to the OMI ASIC. TLXR releases TL.DCP.1 credit back to TL.DCP.1 credit pool after the write done response of the request has been accepted by TLXT and SRQ has issued DRAM write done of this request to TXLR except following two cases. 1. For a 128B OP that occupied 2 DCP.1 credits, TLXR releases the first DCP.1 credit to DCP.1 credit pool as soon as SRQ issued the first write done of the OP to TLXR regardless whether the write done response has been accepted by TLXT or not. 2. For pr_wr_mem(pad mem pattern), TLXR releases DCP.1 credit to DCP.1 credit pool as soon as the write done response is accepted by TLXT.

In early_write_done mode, the write done response of the request is sent to TLXT as soon as data CRC check is good. It is most likely that by the time when SRQ returned the DRAM write done for this request, the write done response has been accepted by TLXT long ago. TLXR would be able to releases TL.DCP.1 credit back to TL.DCP.1 credit pool right away upon receiving the DRAM write done from SRQ. Please note that both events have to be met before TLXR triggers release of DCP.1, the order of the events is irrelevant.

In late_write_done mode, the write done response can only be sent after receiving the DRAM write done from SRQ. TLXR can only release the TL.DCP.1 credit after the write done response has been accepted by TLXT.

COMMAND	TO	Consumed VC0	VC1	DCP1	VCn_RETURN Who	When	DCP1_RETURN Who	When
rd_mem	DRAM		1		SRQ	NCF->RRQ		
rd_mem	bad		1		SRQ	NCF->TLXT(fail)		
pr_rd_mem	DRAM		1		SRQ	NCF->RRQ		
pr_rd_mem	MMIO		1		SRQ	NCF->MMIO		
pr_rd_mem	bad		1		SRQ	NCF->TLXT(fail)		
write_mem	DRAM		1	1/2	SRQ	NCF->WRQ	TLXR	[Note 1] [Note2]
write_mem	bad		1	1/2/4	TLXR	CrcOK	TLXR	TLXR->TLXT(resp) all together
write_mem.be	DRAM		1	1	SRQ	NCF->WRQ	TLXR	[Note 1]
write_mem.be	bad		1	1	TLXR	CrcOK	TLXR	TLXR->TLXT(resp)
pr_wr_mem	MMIO		1	1	SRQ	NCF->MMIO	TLXR	TLXR->TLXT(resp)
pr_wr_mem	DRAM		1	1	SRQ	NCF->WRQ	TLXR	[Note 1]
pr_wr_mem	PADPATT		1	1	TLXR	CrcOK	TLXR	TLXR->TLXT(resp)
pr_wr_mem	bad		1	1	TLXR	CrcOK	TLXR	TLXR->TLXT(resp)
pad_mem	DRAM		1		SRQ	NCF->WRQ		
pad_mem	bad		1		SRQ	NCF->TLXT		
config_read			1		SRQ	NCF->MMIO		
config_write			1	1	SRQ	NCF->MMIO	TLXR	TLXR->TLXT(resp)
mem_cntl		1			TLXR	TLXR->TLXT(resp)		
intrp_resp		1			TLXR	CrcOK		
intrp_rdy		1			TLXR	CrcOK		

[Note 1]: Final dcp1 is sent when the later of { TLXR->TLXT(resp), SRQ->TLXR(done) } occurs.
Late write done will always use the TLXR->TLXT(resp).
Early write done can use either, depending on the order.

[Note 2]: If it's a 128Byte command, first dcp1 is sent when first SRQ->TLXR(done)

1.5.3.2 Credit Consumption by the OMI ASIC TLX

Before packing a response into a control flit, credit counts for tlx virtual channels and data credit pools are checked. If there are no more credits available then the response is not pulled from the corresponding queue. When a response is packed into a control flit the corresponding virtual channel credit counter is decremented by one. For reads, the data credit pool is decremented by either 1 or 2, depending on which data carrier the response uses (32B or 64B) if at all. The TLXR returns TLX credits from the host after receiving a ***return_tlx_credits*** command. The TLXR asserts the ***return_val*** signal and the number of credits returned in each of the return busses. These values are added to the tlx credit pools.

1.5.4 Transmitting Control and Data Flits

After a control flit is packed in the control flit arbiter, it is pushed into the transmit arbiter, provided there is not currently a control flit waiting to be sent upstream. The transmit arbiter asserts a valid to the control flit arbiter in this case to avoid a control flit being overwritten in the buffer. The Transmit arbiter is provided with the data length associated with the control flit as well as the template which will be used.

1.5.5 Modes of Operation

The TLXT can be configured to operate in one of 3 different modes which determine response format and scheduling. These modes are reffered to below as Low latency mode, high latency mode, and half-dimm mode. In all modes data refers to DRAM data received from the Read Data flow as well as register data received by the mmio/cfg macro.

1.5.5.1 Low Latency Mode

When in low latency mode the TLXT will send back read responses 3 flits prior to the corresponding data, this is referred to as the "3-flit rule". Within this mode of operation there are 3 sub-modes which determine the maximum read bandwidth which can be achieved. Flits are sent upstream in sets of 3, referred to as a triplet, and each bandwidth mode has it's own unique triplet. The 3 bandwidth modes are as follows:

- Low Bandwidth: Max 50% read bandwidth
- Mid Bandwidth: Max 66% read bandwidth
- High Bandwidth: Max 83% read bandwidth

In low bandwidth mode, data will be always be returned in a 32B data carrier as part of a template x'09' control flit in a triplet comprised of only control flits. Mid bandwidth mode will return data in 32B data carriers with a single 64B data flit being sent after every other control flit. High bandwidth mode returns data in 32B data carriers with each control flit using the maximum of two 64B data flits. By default the mid bandwidth triplet is disabled, and instead the TLXT will transition directly from the lowest bandwidth triplet to the highest. This transition occurs when there is a backlog of 4 read responses in the read response FIFO and takes 3 flits to fully transition. This along with timing diagrams of all other triplets and triplet transitions is shown in section 1.5.8 below.

In all cases a template x'05' control flit can be inserted into the triplet flow to return an interrupt request without breaking the 3-flit rule or the triplet flow. If template x'05' is disabled then the read flow is halted to return the interrupt request in a template x'00' flit and prevent a breakage of the 3 flit rule. Fail responses and write responses are returned on cycles when there are no read responses can be sent back to the host, either do to restrictions from the triplet flow or a lack of data credits. Restrictions from the triplet flow can include; having already packed an octword response in low bandwidth mode, transitioning to a new triplet type, incurring a one-flit stall due to a response timing issue on the RDF to TLXT interface, or a pending interrupt that can not be optimized into the flow due to template restrictions.

1.5.5.2 High Latency Mode

High latency mode opts for returning data only in 64B data flits. Because metadata is placed in the flit which has the associated data attached as a data run length, it is necessary for metadata, or bad data information if metadata is disabled, to be present prior to any responses being packed into a control flit. This takes a big hit on the latency because the data can not be sent out as soon as it is ready, however if the system is backed up with read data a higher bandwidth than in the high bandwidth low latency mode can be achieved.

1.5.5.3 Half-Dimm Mode

Half-dimm mode is controlled by the global enterprise and half-dimm config bits described in section 2.2.1. The

assumption of This mode works similarly to the Low Latency mode described in section 1.5.5.1 above,in that read responses are send a fixed number of flits prior to the data that is being returned. Unlike low latency mode, the lead time of read responses to data in half-dimm mode is only 2 control flits instead of 3, and there are no variants of the flow which include data return in 64B data flits. Data is instead always returned in a template x'0B' control flit, utilizing the 32B data carrier plus 8B of the xmeta field to return 40B of data to the host.

1.5.6 Control Flit Arbiter

Internal to the Control flit arbiter there are several response FIFO's used to optimize response order on the OpenCAPI link. Responses and register data are packed into a 56B buffer in the control flit arbiter until a valid flit transmission start condition is triggered or if the flit has no more open slots for response. During the transmission time no responses are packed into the buffer.

1.5.6.1 Response Packing Priority

The priority with which responses are pulled from the various queues is as follows.

1. Interrupt Responses
2. MMIO Read Responses
3. DRAM Read Responses
4. MMIO & DRAM Write Responses

1.5.7 Arrays

There are 6 arrays in the TLXT, each acting as a FIFO to track responses and data linked to response and data: the write response queue, read response queue, mmio response queue, fail response queue, meta data queue, and bad data bit queue.

1.5.8 Timing Diagrams

Figure 1: Triplet Types and Transitions

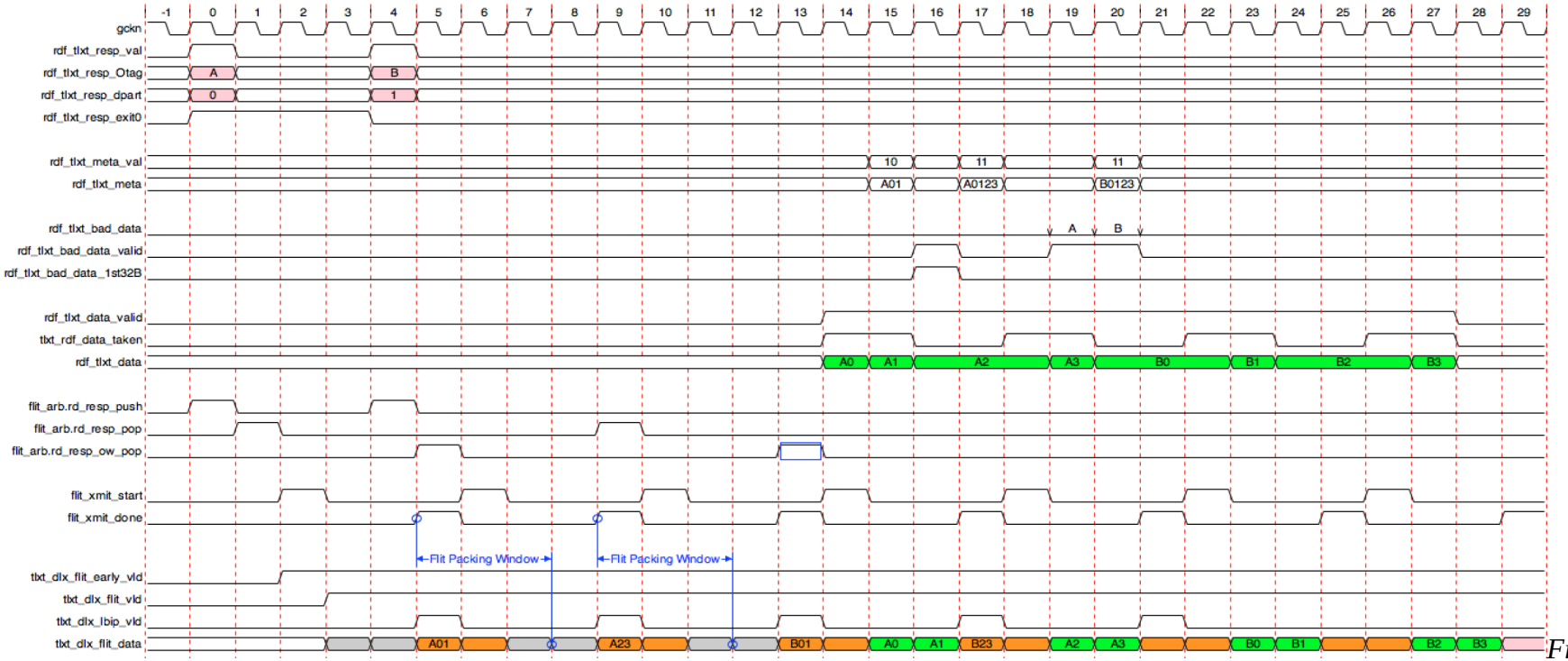


Figure 2: Low Bandwidth Triplet From Idle

1.5.9 Reset Requirements

TBD

1.6 Error Reporting and Handling

1.6.1 Bad Data Bit FIFO Error

The bad data bit indicator is held in FIFO order as received from the RDF and pulled out as data flits are transmitted from the TLXT. To protect this data the Bad Data bit is duplicated when pushed into the fifo and checked on output. If the two data bits do not match when read out from the FIFO the bad data is set to 1 indicating to the host that the corresponding data was bad, even if it wasn't. This is a non-fatal error. There is no way to detect, or recover from, a 2 bit error in this FIFO.

1.6.2 ECC

ECC is generated on the CAPP Tag and associated response information in each of the response queues, as well as metadata information in the metadata queue. When a response is pulled from the queue the ECC is checked and then the response flows into the control flit buffer. In the case of an uncorrectable error the response is dropped and a reported to the global fir, and an interrupt will be triggered.

1.6.3 Hardware Error Checkers

TLXT errors are reported to MB_SIM.TLXT.TLXT_REGS.TLX_ERR1_REPORTQ which has that corresponding mask register MB_SIM.TLXT.TLXT_REGS.TLX_ERR1_REPORTQ_MASK. The column ERR1 Bit in the TLXT error table below corresponds to these registers. These errors are reported to MB_SIM.TLXT.TLXT_REGS.TLXFIRQ. The Fir Bit specified in the error table below correspond to the following FIR bit specification.

TLXC errors are reported to MB_SIM.TLXT.TLXT_REGS.TLX_ERR2_REPORTQ which has that corresponding mask register MB_SIM.TLXT.TLXT_REGS.TLX_ERR2_REPORTQ_MASK. The column ERR21 Bit in the TLXC error table below corresponds to these registers. These errors are reported to MB_SIM.TLXT.TLXT_REGS.TLXFIRQ. The Fir Bit specified in the error table below correspond to the following FIR bit specification.

1.6.3.1 Fir Bit Specification

FIR Bit	Description
0	Info Reg Parity Error
1	Control Reg Parity Error
2	TLX.VC0 Return Credit Overflow
3	TLX.VC1 Return Credit Overflow
4	TLX.DCP0 Return Credit Overflow
5	TLX.DCP3 Return Credit Overflow
6	TLXC Error
7	TLXC Parity Error
8	TLXT Fatal Parity Error

FIR Bit	Description
9	TLXT Recoverable Error
10	TLXT Configuration Error
11	TLXT Unrecoverable Error
12	TLXT Informational Parity Error

1.6.3.2 Error Table

Name	Description	ERR1 bit	FIR bit	Expected Action	Sim Injection
cfg_vers_par_err	Parity error detected on config version	0	128	InformationalFatal. Reset required	Corrupt cfg_otl0_tl_minor_vers_config_p
cfg_tmpl_par_err	Parity error detected on template enables	1	8	Fatal. Reset required	Corrupt cfg_otl0_tl_xmt_template_config(12)
cfg_rate_par_err	Parity error detected on template rate fields	2	8	Fatal. Reset required	Corrupt cfg_otl0_tl_xmt_rate_tmpl_config(48)
cfg_enab_par_err	Parity error detected on afu or metadata enable bits	3	8	Fatal. Reset required	Corrupt cfg_f1_octrl00_p
cfg_actag_par_err	Parity error detected on actag base field	4	8	Fatal. Reset required	Corrupt cfg_f1_ofunc_func_actag_base(12)
cfg_actagl_par_err	Parity error detected on actag length field	5	8	Fatal. Reset required	Corrupt cfg_f1_ofunc_func_actag_len_enab(12)
cfg_pasidl_par_err	Parity error detected on pasid length field	6	8	Fatal. Reset required	Corrupt cfg_f1_octrl00_pasid_length_enabled(5)
cfg_pasid_par_err	Parity error detected on pasid base field	7	8	Fatal. Reset required	Corrupt cfg_g1_octrl00_pasid_base(20)
wr_resp_par_err	Parity error detected on boundary write response	8:10	8	Fatal. Reset required	Corrupt tlxr_tlxt_write_resp_p
mmio_resp_par_err	Parity error detected on mmio response interface	11	8	Fatal Reset required	Corrupt mmio_tlxt_write_resp_p

Name	Description	ERR1 bit	FIR bit	Expected Action	Sim Injection
bad_data_par_err	Parity error detected on bad data interface	12	8	Fatal. Reset required	Corrupt rdf_tlxt_bad_data_p
metadata_par_err	Parity detected on metadata interface	13	8	Fatal. Reset required	Corrupt rdf_tlxt_meta_p
srq_resp_par_err	Parity error detected on response interface from SRQ (padmem and readfail ops)	14:15	8	Fatal. Reset required	Corrupt srq_tlxt_padmem_done_tag_p
arb_resp_par_err	Parity error detected on flit_rd_resp_q or flit_wr_resp_q	16	8	Fatal. Reset required	Corrupt flit_rd_resp_p_q or flit_wr_resp_p_q
tl_sl_vld_par_err	Parity error detected on 1-slot packet or 4-slot packet valid state machines	17	8	Fatal. Reset required	Corrupt tl_1sl_vld_p_q or tl_4sl_vld_p_q
arb_meta_bdi_vld_par_err	Parity Error Detected on Meta Field Valid or Bad Data Field Valid state machines	18	8	Fatal. Reset required.	Corrupt bdi_vld_p_q or mf_vld_p_q
arb_drl_par_err	Parity error detected on data run length counters	19	8	Fatal. Reset required	Corrupt drl_sub0_p_q
arb_pktcnt_par_err	Parity error detected on flit packet counter	20	8	Fatal. Reset required	Corrupt flit_pkt_cnt_p_q
arb_tmpl_par_err	Parity error on tmpl_current_q.	21	8	Fatal. Reset required	Flip a bit in tmpl_current_q.
idlcrdtmr_par_err	Parity error detected on Idle credit return timer	2221	128	Fatal. Reset requiredInformational	Corrupt idl_crd_ret_tmr_p_q
triplet_par err	Parity error detected in triplet return state machine	23	8	Fatal. Reset required	Corrupt triplet_state_q (more than one bit on)
rd_resp_par_err	Parity error detected on read response interface	24	8	Fatal Reset required	Corrupt rdf_tlxt_resp_p
mmio_data_par_err	Parity error detected on mmio data interface	25	12	Data discarded, mem_rd_fail response sent with code=8.	Corrupt mmio_tlxt_rdata_bus parity
eccgen_par_err	Parity error on generated ECC bits	27	8	Fatal. Reset required	Corrupt 1 bit in tlxt_dlx_flit_ecc_egen
flit_credits_par_err	Parity error detected in flit credit counter	28	8	Fatal. Reset required	Corrupt flit_crd_str_p_q
mstr_cnt_par_err	Parity error detected on mstr_cnt in transmit arbiter	29	8	Fatal. Reset required	Corrupt mstr_cnt_p_q

Name	Description	ERR1 bit	FIR bit	Expected Action	Sim Injection
push_pull_par_err	Parity error detected on data push/pull counters	30	8	Fatal. Reset required	Corrupt push_data_cntr_p_q or pull_data_cntr_p_q
flit_data_par_err	Parity error on control flit content where ECC gets added.	31	8	Fatal. Reset required	Corrupt sl_*_p, note that eccgen_par_err may also get set.
tlxt_rd_resp_fifo_ctrl_par_err	Pointer parity error in the read response FIFO.	32	8	Fatal. Halt further read responses. Reset Required	
tlxt_wr_resp_ctrl_par_err	Pointer parity error in the write response fifo	33	8	Fatal. Halt further write responses. Reset required.	
tlxt_meta_fifo_ctrl_par_err	Pointer Parity error in the meta fifo.	34	8	Fatal. Poison all future bad data. Reset required.	
tlxt_fail_resp_fifo_ctrl_par_err	Pointer parity error on the fail response fifo	35	8	Fatal. Halt further fail responses. Reset required.	
tlxt_bdi_fifo_ctrl_par_err	Pointer parity error on the fail response fifo.	36	8	Fatal. Poison all future Bad data bits. Reset required.	
tlxt_fifo_ce	Correctable error on one or more response FIFO.	37	9	Correct response packet. Set informational FIR. NOTE: Currently Set as Fatal In Logic	
intrp_req_failed	Host has responded to an interrupt request with a failing response code.	38	9	Informational. Corresponding interrupt request state machine is reset, as if a good completion was received, and a FIR is set.	
unexpeted_intrp_resp	TLXT has received an interrupt response but no interrupt is awaiting a response.	39	9	Informational FIR Set. Suspected error on the host.	
bdi_poisoned	Bits stored in bad data fifo do not match. Bad data has been indicated to the host.	40	9	Informational FIR. Bad data indicated to host. No further action required. Host will handle as if data has bad ECC or other issue from DRAM.	Flip one bit in an entry of the bad data fifo. Or write in mismatching bits.
tlxt_metadata_ue	Uncorrectable ECC Error on metadata.	41	9	Poison corresponding bad data bit.	
tlxt_invalid_meta_config	Read attempted with metadata enabled, but no template with metadata configured.	42	10	Informational. Poison Bad data on reads until template enabled	Configure metadata but disable template 5 or 9 and issue a read to DRAM.

Name	Description	ERR1 bit	FIR bit	Expected Action	Sim Injection
tlxt_invalid_config	*tlxcfg1_mid_bw_enab* and *tlxcfg1_hi_bw_enab_rd_thresh* are both set and a read has been attempted in low latency mode.	43	10	Back down to lowest bandwidth mode until conflicting configuration is resolved.	Set both specified bits and send read traffic to the OMI ASIC .
tlxt_rd_resp_ue	Uncorrectable ECC error on read response.	44	11	Drop response in error, set FIR. Continue sending further responses.	
tlxt_wr_resp_ue	Uncorrectable ECC error on write response.	45	11	Drop response in error, set FIR. Continue sending further responses.	
tlxt_mmio_resp_ue	Uncorrectable ECC error on mmio response.	46	11	Drop response in error, set FIR. Continue sending further responses.	
tlxt_fail_resp_ue	Uncorrectable ECC error on fail response .	47	11	Drop response in error, set FIR. Continue sending further responses.	
invalid_crd_ret	Credit return has been sent out to DLX but not indicated to credit management logic.	48	11	Too many credits returned to host. Fatal. Reset Required.	
dropped_crd_ret	Credit Return Packet has been dropped before transmission to DLX.	49	11	Too few credits returned to host. Fatal. Reset Required.	
unexpected_val	Unexpected valid on from SRQ. Padmem_Done_Val and fail_resp_val on same cycle.	50	11	Control Error. Fatal Reset required.	
tlxt_rd_resp_fifo_overflow	Read Response FIFO overflow	51	11	Stop sending read responses. Fatal. Reset Required.	
tlxt_wr_resp_fifo_overflow	Write Response FIFO overflow	52	11	Stop sending Write responses. Fatal. Reset Required.	
tlxt_meta_fifo_overflow	Meta FIFO overflow	53	11	Poison Bad all future Bad data. Fatal. Reset Required.	
tlxt_fail_resp_fifo_overflow	Fail Response FIFO overflow	54	11	Stop sending read responses. Fatal. Reset Required.	
tlxt_bdi_fifo_overflow	Bad Data FIFO overflow	55	11	Poison all future Bad data. Fatal. Reset Required.	
tlxt_rd_resp_fifo_underflow	Read Response FIFO underflow	56	11	Stop sending read responses. Fatal. Reset Required.	
tlxt_wr_resp_fifo_underlow	Write Response FIFO underlow	57	11	Stop sending Write responses. Fatal. Reset Required.	
tlxt_meta_fifo_underflow	Meta FIFO underflow	58	11	Poison Bad all future Bad data. Fatal. Reset Required.	
tlxt_fail_resp_fifo_underlow	Fail Response FIFO underflow	59	11	Stop sending read responses. Fatal. Reset Required.	

Name	Description	ERR1 bit	FIR bit	Expected Action	Sim Injection
tlxt_bdi_fifo_underlow	Bad Data FIFO underflow	60	11	Poison all future Bad data. Fatal. Reset Required.	
flit_credit_underflow	Flit Credit Counter underflow	61	11	Fatal, reset required	
flit_credit_overflow	Flit Credit Counter overflow	62	11	Fatal. Reset Required	
invalid_template_config	Invalid transmit template configuration. Interrupt required without template with 4 slot packet.	63	10	Non-fatal, edit transmit template configuration to include a template with a 4 slot command.	Disable templates 0,1 and 5 and inject an error that will trigger a fir.

1.6.3.3 TLXC Error Table

Name	Description	ERR2 bit	FIR bit	Expected Action	Sim Injection
spare	***spare***	0		spare	
tlxc_tlxt_avail_credit_count_vc0_perror	***tlxc_tlxt_avail_credit_count_vc0_perror***	1	7	Set FIR. Fatal	
tlxc_tlxt_avail_credit_count_vc3_perror	***tlxc_tlxt_avail_credit_count_vc3_perror***	2	7	Set FIR. Fatal	
tlxc_tlxt_avail_credit_count_dcp0_perror	***tlxc_tlxt_avail_credit_count_dcp0_perror***	3	7	Set FIR. Fatal	
tlxc_tlxt_avail_credit_count_dcp3_perror	***tlxc_tlxt_avail_credit_count_dcp3_perror***	4	7	Set FIR. Fatal	
tlxc_tlxt_vc0_credits_perror	***tlxc_tlxt_vc0_credits_perror***	5	7	Set FIR. Fatal	
tlxc_tlxt_vc1_credits_perror	***tlxc_tlxt_vc1_credits_perror***	6	7	Set FIR. Fatal	
tlxc_tlxt_dcp1_credits_perror	***tlxc_tlxt_dcp1_credits_perror***	7	7	Set FIR. Fatal	
vc0_cnt_underflow_error	***vc0_cnt_underflow_error***	8	6	Set FIR. Fatal	
vc1_cnt_underflow_error	***vc1_cnt_underflow_error***	9	6	Set FIR. Fatal	
dcp1_cnt_underflow_error	***dcp1_cnt_underflow_error***	10	6	Set FIR. Fatal	
vc0_max_crd_error	***vc0_max_crd_error***	11	6	Set FIR. Fatal	

Name	Description	ERR2 bit	FIR bit	Expected Action	Sim Injection
vc1_max_crd_error	***vc1_max_crd_error***	12	6	Set FIR. Fatal	
dcp1_max_crd_error	***dcp1_max_crd_error***	13	6	Set FIR. Fatal	
inthld_0_reg_perr	inthld_0_reg_perr	14	1	Set FIR. Fatal	
inthld_1_reg_perr	inthld_1_reg_perr	15	1	Set FIR. Fatal	
inthld_2_reg_perr	inthld_2_reg_perr	16	1	Set FIR. Fatal	
inthld_3_reg_perr	inthld_3_reg_perr	17	1	Set FIR. Fatal	

1.6.4 Error Injection Capabilities

The TLXT can inject a single or double bit error into it's 8B ECC generation on control flits sent to the DLX. The table below shows the dials associated with this functionality.

Name	Description	Expected Action
TLXCFG1_CFEI_ENAB	Conrtol Flit Error Inject enable.	Enable Error Inject capability
TLXCFG1_CFEI_PERSIST	Persistent error inject enable.	When 0 a single error injection will occur. When 1 an error is injected on every cycle of control flit transmitted.
TLXCFG1_CFEI_BIT0	Enable for error injection on bit0 of 16B control flit.	Inject an error into tlxt_dlx_flit_data(0) Can be set with CFEI_BIT1 for a double bit error.
TLXCFG1_CFEI_BIT1	Enable for error injection on bit1 of 16B control flit.	Inject an error into tlx_dlx_flit_data(1). Can be set with CFEI_BIT0 for a double bit error.

There are 2 ways to inject an error, either a single error or a persistent error, which are controlled by the *CFEI_PERSIST* dial. When this bit is set an error will be injected on every valid cycle of control flit that is transmitted from the TLXT. When the bit is not set a single error will be injected on the first cycle of control flit after the rising edge of the CFEI_BIT0/1 bits. For either injection mode the type of error injected is determined by the two bit enables, TLXCFG1_BIT0 and TLXCFG1_BIT1. Either bit can be set for a single bit error, or both bits can be set for a 2-bit error.

1.7 Debug Capabilities

1.7.1 Trace

1.7.1.1 Trace Control

The TLXT trace bus is controlled by the tlxt debug dial (TLXCFG1_tlxt_dbg_dial) which is summarized in the table below.

TLXCFG1_TLXT_DBG_DIAL	
Bit Range	**Description**
0:1	Selection for Debug Bus bit 0:21 '00' -> select Debug Bus bit0:21 from tlxc_dbg_a_bus(0 to 21) '01' -> select Debug Bus bit0:21 from tlxc_dbg_b_bus(0 to 21) '10' -> select Debug Bus bit0:21 from tlxt_dbg_a_bus(0 to 21) ‘11' -> select Debug Bus bit0:21 from tlxr_dbg_bus(0 to 21)
2:3	Selection for Debug Bus bit 22:43 '00' -> select Debug Bus bit22:43 from tlxc_dbg_a_bus(22 to 43) ‘01' -> select Debug Bus bit22:43 from tlxc_dbg_b_bus(22 to 43) '10' -> select Debug Bus bit22:43 from tlxt_dbg_a_bus(22 to 43) '11' -> select Debug Bus bit22:43 from tlxr_dbg_bus(22 to 43)
4:5	Selection for Debug Bus bit 44:65 '00' -> select Debug Bus bit44:65 from tlxc_dbg_a_bus(44 to 65) '01' -> select Debug Bus bit44:65 from tlxc_dbg_b_bus(44 to 65) '10' -> select Debug Bus bit44:65 from tlxt_dbg_a_bus(44 to 65) '11' -> select Debug Bus bit44:65 from tlxr_dbg_bus(44 to 65)
6:7	Selection for Debug Bus bit 66:87 '00' -> select Debug Bus bit66:87 from tlxc_dbg_a_bus(66 to 87) '01' -> select Debug Bus bit66:87 from tlxc_dbg_b_bus(66 to 87) '10' -> select Debug Bus bit66:87 from tlxt_dbg_a_bus(66 to 87) '11' -> select Debug Bus bit66:87 from tlxr_dbg_bus(66 to 87)
8	Flip tlxc_dbg_a_bus bit 0:43 with bit 44:87
9	Flip tlxc_dbg_b_bus bit 0:43 with bit 44:87
10	Flip tlxt_dbg_a_bus bit 0:43 with bit 44:87
11	Flip tlxr_dbg_bus bit 0:43 with bit 44:87
12	Enable TLXT Debug Bus
13:15	RSVD

1.7.1.2 Trace Bus Descriptions

	Trace Bus A	Trace Bus B	Trace Bus C	Trace Bus D
Bit	**Signal Name**	**Signal Name**	**Signal Name**	**Signal Name**
0	.rd_idle	rsvd	srq_tlxt_padmem_done_val	flit_xmit_done
1	wr_resp_empty	rsvd	ssrq_tlxt_failresp_val	data_xmit
2	wr_resp_pop	rsvd	srq_tlxt_failresp_code(1:3)	data_flit_xmit
3	rd_resp_pop	rsvd		data_valid
4	rd_resp_ow_pop	rsvd		wr_resp_count(3:0)
5	mmio_resp_pop	rsvd	rdf_tlxt_resp_val	
6	fail_resp_pop	rsvd	rdf_tlxt_resp_dpart(0)	
7	meta_fifo_pop	rsvd	rdf_tlxt_data_valid	
8	00 → tmpl0 01 → tmpl1 10 → tmpl5 11 → tmpl9	rsvd	rdf_tlxt_meta_valid(0:1)	rsvd
9		rsvd		rsvd
10	triplet_state_q(2)	rsvd	rdf_tlxt_bad_data_valid	tmpl_current_q(4:0)
11	triplet_state_q(1)	rsvd	rdf_tlxt_bad_data_1st32B	
12	flit_credit_avail	tlxc_tlxt_vc0_credits_q(15:0)	rdf_tlxt_bad_data	
13	flit_hold_1st		tlxr_tlxt_write_resp_valid	
14	flit_hold_2nd		tlxr_tlxt_write_resp(0)	
15	mmio_data_pending		tlxr_tlxt_wr_resp(5:2)	rd_resp_count(5:0)
16	tl_1sl_vld_q(3:0)			
17				
18				
19			mmio_tlxt_resp_code(3:0)	
20	tl_4sl_vld(1:0)			
21				data_rnu_length(3:0)
22	flit_buffer_reset			
23	flit_xmit_start_q		tlxr_tlxt_intrp_resp(7:0)	
24	tlx_vc0_avail			
25	tlx_dcp0_avail			flit_pkt_cnt_q(3:0)
26	wr_resp_flush_q			
27	data_valid			
28	rdf_tlxt_resp_valid	tlxr_tlxt_vc0_release_q(1:0)		
29	rdf_tlxt_resp_dPart(0)			flit_rd_resp_q(3:0)
30	rdf_tlxt_data_valid	vc0_credit_update_val		

	Trace Bus A	Trace Bus B	Trace Bus C	Trace Bus D
31	rdf_tlxt_meta_valid(0)	vc0_credit_return_pause		
32	tlxr_tlxt_write_resp_val	tlxt_tlxc_crd_ret_taken	srq_tlxt_padmem_done_tag(0:15)	
33	tlxr_tlxt_write_resp(0)	cfg_half_dimm_mode		flit_wr_resp_q(3:0)
34	srq_tlxt_padmem_done_val	tlxr_tlxt_return_val_q		
35	srq_tlxt_failresp_val	rsvd		
36	mmio_tlxt_resp_valid	tlxr_tlxt_return_vc0_gt(3:0)		
37	tlxr_tlxt_return_valid			flit_mmio_resp_q(3:0)
38	or_reduce(tlxr_tlxt_dcp1_release(1: 0))			
39	or_reduce(tlxr_tlxt_vc0_release(1; 0))			
40	or_reduce(tlxr_tlxt_vc1_release & srq_tlxt_cmdq_release)	tlxr_tlxt_return_vc3(3:0)		
41	and_reduce(tlxr_tlxt_dcp1_release(1 downto 0))			flit_xmit_start_q
42	tlxr_tlxt_vc1_release			tl_2sl_pkt_full_q
43	srq_tlxt_cmdq_release			tl_4sl_vld_q(1:0)
44	tlxc_tlxt_dcp1_credits_q(15:0)	tlxr_tlxt_return_dcp0(5:0)		
45				drl_current(3:0)
46				
47				
48			rdf_tlxt_resp_otag(0:15)	
49				mmio_data_current
50		tlxr_tlxt_return_dcp3(5:0)		rdf_1st_32B_current
51				rdf_2nd_32B_current
52				rdf_data_current
53				bdi_vld_q(3:0)
54				
55				
56				
57		tlxt_tlxc_consume_vc0(0)		mmio_data_flit_q
58		tlxt_tlxc_consume_vc3(0)		mmio_data_val_q
59		tlxt_tlxc_consume_dcp0(0)		tmpl9_data_valid_q
60	tlxr_tlxt_dcp1_release_q(2:0)	tlxt_tlxc_consume_dcp3(0)		data_flit_xmit_q

	Trace Bus A	Trace Bus B	Trace Bus C	Trace Bus D
61		rsvd		triplet_999_q
62		rsvd		triplet_9d9_q
63	dcp1_credit_update_val	rsvd		triplet_9dd_q
64	dcp1_credit_return_pause	rsvd	tlxr_tlxt_write_resp(21:6)	triplet_state_q(2:0)
65	tlxc_tlxt_vc1_credits_q(15:0)	rsvd		
66		rsvd		
67		rsvd		rd_resp_ow_pend_q
68		rsvd		flit_credit_avail
69		rsvd		ow_meta_pend_q
70		rsvd		ow_meta_packed_q
71		rsvd		ow_meta_bypass_taken_q
72		tlxc_tlxt_avail_vc0_q		ow_bdi_pend_q
73		tlxc_tlxt_avail_vc3_q		ow_bdi_packed_q
74		tlxc_tlxt_dcp0_avail_q		bdi_bypass_taken_q
75		tlxc_tlxt_dcp3_avail_q		wr_resp_flush_q
76		rsvd		tl_4sl_pkt_full_q
77		rsvd		rsvd
78		rsvd		rsvd
79		rsvd		rsvd
80		rsvd	srq_tlxt_failresp_dlen	rsvd
81	srq_tlxt_cmdq_release_q	rsvd		rsvd
82	tlxr_tlxt_vc1_release_q	rsvd	rdf_tlxt_meta(0:5)	rsvd
83	vc1_credit_update_val	rsvd		rsvd
84	vc1_credit_return_pause	rsvd		rsvd
85	tlxt_tlxc_crd_ret_taken	rsvd		rsvd
86	syncr_falling_q	rsvd		rsvd
87	cfg_half_dimm_mode	rsvd		rsvd